Mémoire

Auteur : Renard, Antoine
Promoteur(s) : Rigo, Michel
Faculté : Faculté des Sciences
Diplôme : Master en sciences mathématiques, à finalité didactique
Année académique : 2021-2022
URI/URL : http://hdl.handle.net/2268.2/14701

Faculté des Sciences
Département de Mathématique

La cryptographie sur les courbes elliptiques, et son application à la blockchain du Bitcoin

Mémoire de fin d'études présenté en vue de l'obtention du titre de
Master en Sciences Mathématiques, à finalité didactique

Année académique 2021 – 2022

Auteur :
Antoine RENARD

Promoteur :
Michel RIGO

Remerciements

Tout d'abord, je tiens à remercier chaleureusement mon promoteur, Michel Rigo, de m'avoir permis d'explorer un sujet qui m'était totalement inconnu mais pour lequel j'ai pu développer un grand intérêt. Je le remercie également pour sa disponibilité, ses nombreuses relectures ainsi que ses conseils et remarques qui m'ont permis d'améliorer continuellement la qualité de mon travail.

Je pense également à mes proches, qui m'ont toujours soutenu dans mes choix et encouragé quoi qu'il arrive. Je tiens notamment à remercier mes parents et ma sœur d'avoir toujours cru en moi et d'avoir été présents, autant dans les moments de bonheur que dans ceux de doute et de stress. Un merci tout particulier à ma maman, qui a toujours pris le temps de relire mes travaux, même si les sujets lui semblaient parfois totalement farfelus. Sans oublier Damien, à qui j'ai sûrement communiqué beaucoup de stress parfois, mais qui a toujours trouvé le moyen de me réconforter. Merci à eux d'avoir toujours su trouver les mots justes pour me rassurer et me faire aller de l'avant.

Enfin, je remercie toutes les personnes dont j'ai croisé la route durant ces cinq années d'études. Je pense tout particulièrement à Emeline, qui aura été bien plus qu'une simple marraine à mes yeux, mais aussi à Ophélie et Elodie, qui ont rendu ces deux années de master inoubliables. Merci pour votre soutien sans faille et tous ces moments partagés ensemble. Nous avons créé des liens et des souvenirs qui, j'en suis sûr, ne s'effaceront jamais.

Table des matières

Introduction

Depuis l'apparition du Bitcoin en 2009, les cryptomonnaies n'ont cessé de gagner en importance. A l'heure actuelle, elles interviennent dans une part significative des transactions économiques effectuées de façon électronique. En outre, différentes entreprises tentent de plus en plus de démocratiser les cryptomonnaies, et de rendre leur utilisation accessible au grand public. Citons notamment le groupe `Crypto.com` qui, ces dernières années, s'est largement développé grâce à ses campagnes publicitaires et ses sponsors stratégiques, d'abord aux États-Unis, puis en Europe (notamment en Belgique ; certain.e.s auront d'ailleurs peut-être déjà aperçu le spot publicitaire télévisuel de `Crypto.com`, qui n'est diffusé sur nos ondes que depuis quelques mois au moment de la rédaction de ce mémoire).

De façon plus générale, la cryptographie est omniprésente dans le monde moderne, même si elle n'est pas toujours visible. Elle intervient dans nos transactions bancaires, nos achats en ligne, ou encore nos messages sur des applications telles que WhatsApp ou Messenger.

Partant de ce constat, il n'est pas insensé de s'intéresser aux systèmes mis en place pour sécuriser toutes ces transactions impliquant des cryptomonnaies. En particulier, l'objectif principal de ce travail sera de comprendre les fondements mathématiques et le fonctionnement de la ***blockchain***, méthode utilisée pour sécuriser la toute première cryptomonnaie mise en circulation, le Bitcoin.

Ce travail s'articule autour de trois grands chapitres. Le premier a pour objectif d'introduire la notion de ***courbe elliptique***, intervenant dans de nombreux domaines en cryptographie (dont le Bitcoin). Il s'appuie notamment sur l'ouvrage de Joseph H. SILVERMAN, *"The arithmetic of elliptic curves"* [47] ainsi que sur le mémoire de fin d'étude [13].

En guise de préambule, quelques prérequis nécessaires à la compréhension des différents concepts liés aux courbes elliptiques sont tout d'abord présentés. En particulier, nous rappelons la notion d'espace projectif avant de l'illustrer à travers les exemples de la droite et du plan projectifs réels. La définition de courbe projective est alors présentée, en s'arrêtant un instant sur le lien existant entre une telle courbe et son analogue dans l'espace affine correspondant, ce qui permet notamment d'introduire la notion de ***point à l'infini***.

Les ***courbes elliptiques*** sont ainsi naturellement présentées comme étant des courbes projectives planes, non-singulières et irréductibles d'équation

$$y^2z + a_1xyz + a_3yz^2 - x^3 - a_2x^2z - a_4xz^2 - a_6z^3 = 0,$$

où les coefficients a_1, a_2, a_3, a_4, a_6 appartiennent au champ K sur lequel la courbe est définie. Nous illustrons ensuite ce "nouvel" objet mathématique par divers exemples.

S'en suivent quelques propriétés fondamentales de ces courbes. Nous investiguons notamment les simplifications pouvant être opérées sur l'équation associée, qui dépendent de la caractéristique du champ sur lequel la courbe est définie. Nous établissons également une caractérisation des courbes elliptiques d'un point de vue algébrique : après avoir défini la notion de ***discriminant*** d'une telle courbe, nous montrons en effet qu'une courbe projective plane est une courbe elliptique, si et seulement si son discriminant est nul.

Finalement, la structure de groupe formée par l'ensemble des points d'une courbe elliptique est progressivement développée. Nous nous intéressons tout d'abord au cas des courbes elliptiques réelles, qui permet notamment une interprétation géométrique de la loi de composition introduite. Celle-ci est ensuite définie dans un cadre général par l'intermédiaire d'expressions algébriques. Une attention particulière est finalement accordée au cas des courbes définies sur un champ fini, notamment en décrivant la structure du groupe formé par les points de la courbe.

Dans un second temps, nous abordons l'utilisation qui est faite des courbes elliptiques dans le domaine de la cryptographie, en nous basant principalement sur les découvertes de NEAL KOBLITZ reprises notamment dans son article *"Elliptic Curve Cryptosystems"* [23] ainsi que dans son ouvrage *"A Course in Number Theory and Cryptography"* [24], le livre *"Modern Cryptography and Elliptic Curves, A Beginner's Guide"* rédigé par THOMAS R. SHEMANSKE [46], et les notes du cours de Mathématiques Discrètes de MICHEL RIGO [43].

Le début du chapitre est consacré à une brève introduction historique et biographique. Pour pallier l'absence d'un cours dédié à la cryptographie dans le cursus en Sciences Mathématiques, nous exposons ensuite quelques notions de base de ce domaine. Entre autres, nous explicitons les concepts de ***cryptosystèmes***, ***codage***, ***clés secrète*** et ***privée***. Ceux-ci sont alors illustrés sur un exemple de cryptosystème à clé publique bien connu, le ***RSA***, que nous décrivons explicitement théoriquement mais aussi en pratique.

Ensuite, nous nous intéressons à l'intérêt de l'utilisation des courbes elliptiques en cryptographie, en abordant notamment le ***problème du logarithme discret***. Celui-ci concerne la recherche, étant donnés un groupe cyclique G d'ordre n, un de ses générateur y et un élément $x \in G$, d'un entier $k \in \{0, \dots, n-1\}$ tel que $x = y^k$. Bien que relativement simple en apparence, ce problème est supposé insoluble (moyennant l'utilisation de paramètres adéquats), ce qui assure la sécurité des cryptosystèmes reposant sur sa résolution. En particulier, nous expliquons en quoi son analogue dans le cas d'un groupe formé par les points d'une courbe elliptique permet la construction de cryptosystèmes davantage sécurisés.

Nous clôturons le chapitre par la description du cryptosystème de Massey-Omura. Créé en 1982, ce système à clé publique repose sur le caractère insoluble du problème du logarithme discret, ce qui permet de faire un lien avec la section précédente. Après avoir présenté l'algorithme originel étape par étape, nous nous intéressons à son analogue en termes de courbes elliptiques, avec un point d'attention accordé à la façon d'associer un message clair à un point de la courbe considérée. Un petit exemple est alors réalisé pour illustrer le cryptosystème dans un cadre concret.

Enfin, le dernier chapitre de ce mémoire est consacré au Bitcoin proprement dit et à l'explication de son principe de fonctionnement. Nous prenons notamment appui sur l'ouvrage *"Bitcoin and Cryptocurrency Technologies, A Comprehensive Introduction"* de NARAYANAN ET. AL [33], ainsi que sur l'article de SATOSHI NAKAMOTO, *"Bitcoin : A Peer-to-Peer Electronic Cash System"* [32], à l'origine de la création de la célèbre cryptomonnaie.

A nouveau, nous consacrons le début du chapitre à un petit historique de la genèse du Bitcoin. Nous introduisons ensuite divers outils mathématiques et cryptographiques intervenant dans le protocole de la cryptomonnaie et nécessaires à sa compréhension. En particulier, nous mettons en avant les notions de ***fonctions de hachage***, qui assurent l'intégrité d'un message, et de ***signatures digitales***, qui attestent quant à elles la provenance d'un message chiffré. La ***blockchain***, structure de données utilisée dans le protocole de plusieurs cryptomonnaies (dont le Bitcoin), est également définie.

Subséquemment, nous abordons le fonctionnement du Bitcoin progressivement. Plus précisément, nous partons d'une situation de départ relativement simple, qui consiste à trouver un moyen de gérer les dépenses effectuées par un groupe d'amis, et nous la complexifions petit à petit en ajoutant les différents éléments intervenant dans le protocole. Par ailleurs, un point

d'attention est accordé à la structure et la construction des blocs de la chaîne, en particulier la notion de ***minage***, ainsi qu'à la forme des transactions.

Pour terminer, nous établissons un lien entre les courbes elliptiques et le Bitcoin en introduisant l'***Elliptic Curve Digital Signature Algorithm***, algorithme de signature digitale dans lequel elles interviennent. Plus précisément, nous présentons d'abord son analogue classique, étape par étape, avant de le décrire dans le cas des courbes elliptiques. Finalement, nous introduisons les paramètres utilisés par le protocole du Bitcoin, en nous arrêtant un instant sur la simplicité apparente de la courbe secp256k1, dont l'équation est donnée par

$$y^2 = x^3 + 7.$$

Chapitre 1

Les courbes elliptiques

Dans ce premier chapitre, nous allons poser les bases mathématiques nécessaires à la compréhension du concept de courbe elliptique. Nous commencerons par quelques rappels concernant les espaces projectifs. La notion de courbe elliptique sera alors naturellement introduite, et les principales propriétés de ces "nouveaux" objets mathématiques seront investiguées, notamment la structure de groupe que possède l'ensemble des points d'une telle courbe.

1.1 Quelques rappels et notations

Avant de nous lancer dans le vif du sujet, il semble opportun de rappeler ici quelques notions et notations d'algèbre (tirées de [16] et [17]), afin de bien comprendre les lignes qui suivent.

Définition 1.1.1. Soit K un champ.

1. Une ***fonction polynôme***[1] f sur K est une fonction

$$K \to K : x \mapsto a_n x^n + \cdots + a_1 x + a_0,$$

avec $a_i \in K$, pour tout $i \in \{0, 1, \ldots, n\}$. Si $a_n \neq 0$, alors le ***degré*** du polynôme vaut n. Un élément $\alpha \in K$ est une ***racine*** de f si $f(\alpha) = 0$.

L'ensemble des polynômes sur K en la variable x forme un anneau, et est noté $K[x]$. Par extension, nous désignerons par $K[x_1, \ldots, x_n]$ l'anneau des polynômes à n variables, notées $x_1, \ldots, x_n$, à coefficients dans K.

2. Le ***corps de rupture*** d'un polynôme f dans K, noté $\mathrm{rupt}_K f$, est une extension de K telle que f y possède toutes ses racines, et qui est minimale pour cette propriété.
3. La ***clôture algébrique*** de K, notée $\overline{K}$, est une extension de K dans laquelle tout polynôme $f \in K[x]$ se factorise complètement, et qui est minimale pour cette propriété. Un champ égal à sa clôture algébrique est dit ***algébriquement clos***.

Remarque 1.1.2. Pour rappel, nous nous permettons de parler **du** corps de rupture d'un polynôme et de **la** clôture algébrique d'un champ, car étant donnés un champ et un polynôme sur celui-ci, ces deux structures sont uniques à un isomorphisme près. Le lecteur intéressé peut consulter [17] pour plus d'informations.

Dans les pages qui suivent, nous désignerons par K un champ quelconque, et $\overline{K}$ sa clôture algébrique.

1. Par la suite, nous utiliserons simplement le terme de ***polynôme***, le caractère fonctionnel de cet objet étant sous-entendu.

De manière plus générale, des notions liées à la théorie des groupes interviendront dans la suite de ce travail. Il semble donc judicieux de rappeler ici quelques définitions et résultats utiles.

Définition 1.1.3. Soit 1_K le neutre du champ K pour la multiplication. La ***caractéristique*** du champ K est le plus petit entier p tel que

$$\underbrace{1_K + \ldots + 1_K}_{p \text{ fois}} = 0$$

si un tel nombre existe, et 0 sinon.

Remarque 1.1.4. Rappelons ici que, si la caractéristique p d'un champ K est non nulle, alors il s'agit nécessairement d'un nombre premier (d'où la notation utilisée).

Définition 1.1.5. Soit G un groupe et soit $X \subseteq G$. Le ***sous-groupe engendré par*** X, noté $\langle X \rangle$, est le plus petit sous-groupe de G contenant X. Si $X = \{x\}$, alors on le note simplement $\langle x \rangle$.
Un groupe G est dit ***cyclique*** s'il existe $x \in G$ tel que $G = \langle x \rangle$, *i.e.* tel que G est le groupe engendré par x.

Exemple 1.1.6. Le groupe $(\mathbb{Z}_n, +, 0)$ est cyclique pour tout naturel n non nul.

Rappelons également ici la notion d'ordre, et qui interviendra plus tard lorsque nous nous intéresserons aux courbes elliptiques définies sur des champs d'ordre fini.

Définition 1.1.7. L'***ordre*** d'un groupe G est le nombre d'éléments qui le composent. On le note $|G|$.
L'***ordre d'un élément*** x d'un groupe G, noté $o(x)$, est l'ordre du sous-groupe $\langle x \rangle$. En particulier, en notation additive et si e est le neutre de G, on a $o(x) \cdot x = e$.

Quelques résultats relatifs à ces notions semblent également importants à mentionner ici. Ils ne seront pas démontrés, la réalisation de leurs preuves sortant du cadre fixé pour ce mémoire.

Proposition 1.1.8. *Si G est un groupe fini et si $x \in G$, alors l'ordre de x divise celui de G.*

Théorème 1.1.9. *Si p est un nombre premier et n un entier positif, alors il existe, à un isomorphisme près, un et un seul champ d'ordre p^n.*

Le théorème précédent nous permet d'introduire une notation que nous utiliserons abondamment par la suite : $\mathbb{F}_{p^n}$ désignera l'unique champ d'ordre p^n, à isomorphisme près, p étant la caractéristique de ce champ. Par la suite, nous noterons simplement $\mathbb{F}_q$, étant sous-entendu que $q = p^n$ pour un certain entier p premier et un certain naturel n non nul.

Enfin, il n'est pas inutile de rappeler ce que l'on entend par groupe multiplicatif d'un champ fini.

Définition 1.1.10. Le ***groupe multiplicatif*** du champ $\mathbb{F}_q$, parfois également appelé ***groupes des unités*** et noté $\mathbb{F}_q^*$, est le groupe $(\mathbb{F}_q \setminus \{0\}, \cdot, 1)$, où 0 et 1 sont respectivement les neutres pour l'addition et la multiplication.

Proposition 1.1.11. *Le groupe multiplicatif d'un champ fini $\mathbb{F}_q$ est cyclique d'ordre $q - 1$.*

1.2 Espace projectif

Bien qu'elles soient généralement représentées dans un espace affine à deux dimensions, les courbes elliptiques sont en réalité des courbes projectives, *i.e.* définies sur un espace projectif. Rappelons donc les différentes définitions des notions dont nous aurons besoin par la suite. La plupart des informations présentées dans cette section sont tirées de [10] et [45].

1.2.1 Premières définitions et exemples

L'espace projectif étant un espace quotient, nous devons d'abord définir la relation d'équivalence qui interviendra dans celui-ci.

Définition 1.2.1. Soit V un espace vectoriel de dimension $n+1$ sur le champ K. Nous désignerons par ***relation projective*** la relation $\sim$ définie sur l'espace vectoriel $V \setminus \{0\}$ par

$$u \sim v \Leftrightarrow \exists \lambda \in K \setminus \{0\} \text{ tel que } u = \lambda v,$$

pour tous $u, v \in V \setminus \{0\}$.

Il est aisé de montrer que la relation projective définie ci-dessus est bien une relation d'équivalence. Nous pouvons dès lors donner une définition de l'espace projectif :

Définition 1.2.2. Soit V un espace vectoriel de dimension $n+1$ sur le champ K. L'***espace projectif***[2] ***de dimension*** n ***sur*** V, noté $\mathbb{P}(V)$, est l'ensemble quotient $V \setminus \{0\}/\sim$. En d'autres termes, l'espace projectif $\mathbb{P}(V)$ est donc l'ensemble des ***droites vectorielles*** de V, *i.e.* l'ensemble des sous-espaces vectoriels de dimension 1 de V.
Étant donnée une base $(e_0, \ldots, e_n)$ de V, la classe d'équivalence d'un vecteur non nul u de composantes $(x_0, \ldots, x_n)$ sera notée $[x_0, \ldots, x_n]$. Les éléments d'un espace projectif seront naturellement appelés ***points projectifs***, et les scalaires $x_0, \ldots, x_n$ sont appelés ***coordonnées homogènes*** ou ***projectives*** du point.

Remarque 1.2.3. Si V est un espace vectoriel de dimension $n+1$ sur le champ K, alors il est isomorphe à l'ensemble K^{n+1} vu comme K-vectoriel. Par la suite, nous supposerons donc souvent travailler avec cet espace vectoriel. Le cas échéant, nous noterons alors l'espace projectif $\mathbb{P}^n(K) := \mathbb{P}(K^{n+1})$.

Enfin, introduisons une dernière définition, qui nous sera utile par la suite :

Définition 1.2.4. Un ***espace affine***[3] ***de dimension*** n ***sur*** K est la donnée d'un triplet (E, V, t), où

- E est un ensemble non vide,
- V est un espace vectoriel de dimension n sur K,
- $t : E \times E \to V : (A, B) \mapsto \overrightarrow{AB}$ est une application telle que
  (1) $\overrightarrow{AB} + \overrightarrow{BC} = \overrightarrow{AC}$, $\forall A, B, C \in E$,
  (2) $\forall A \in E$, $\forall u \in V$, $\exists! B \in E$ tel que $\overrightarrow{AB} = u$.

Ici également, les éléments de E seront appelés ***points affines***.

2. On parle également de K-***espace projectif***.
3. On parle également de K-***espace affine***.

Remarque 1.2.5. A nouveau, nous pouvons considérer le cas où $E = V = K^n$, ainsi que l'application

$$t : K^n \times K^n \to K^n : ((x_1, \ldots, x_n), (y_1, \ldots, y_n)) \mapsto (y_1 - x_1, \ldots, y_n - x_n),$$

et alors assimiler l'espace affine de dimension n sur K à l'ensemble des n-uples

$$\{(x_1, \ldots, x_n) \mid x_i \in K, \ \forall 1 \leq i \leq n\}.$$

Cet ensemble est noté $\mathbb{A}^n(K)$.

Exemple 1.2.6. Un exemple fort connu d'espace projectif est celui de la ***droite projective réelle***. Nous entendons par cette appellation l'espace projectif réel de dimension 1, $\mathbb{P}^1(\mathbb{R})$, autrement dit l'ensemble des droites vectorielles de $\mathbb{R}^2$ (*i.e.* l'ensemble des droites de $\mathbb{R}^2$ passant par l'origine du repère), vu comme espace vectoriel.

Cependant, nous aimerions décrire la droite projective en termes de points et non en termes de droites. L'idée est donc d'assimiler chaque droite vectorielle à un point. Il faut néanmoins faire attention au choix de ce dernier. Entre autres, il ne faut pas que deux droites vectorielles distinctes soient associées à un même point, et inversement.

Une solution possible est de caractériser chaque droite vectorielle par son intersection avec une partie de $\mathbb{R}^2$. Dans cette optique, considérons le cercle unité

$$S^1 = \{(x, y) \in \mathbb{R}^2 \mid x^2 + y^2 = 1\}.$$

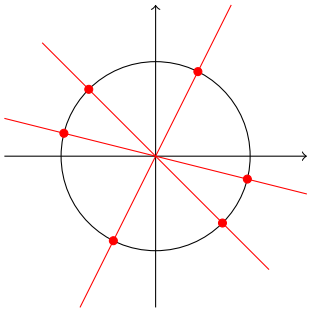

FIGURE 1.1 – S_1

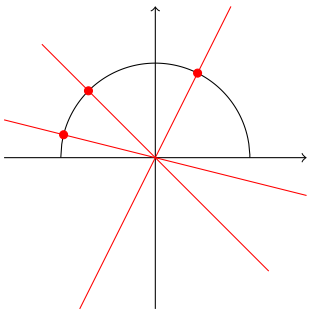

FIGURE 1.2 – S_1^+

Chaque droite vectorielle l'intersecte en exactement deux points diamétralement opposés, univoquement déterminés par celle-ci (voir Figure 1.1). Dès lors, nous pouvons nous contenter d'un demi-cercle pour décrire les droites (comme sur la Figure 1.2). Considérons l'ensemble

$$\begin{aligned} S_+^1 &= \{(x, y) \in S^1 \mid y > 0\} \cup \{(1, 0)\} \\ &= \{(\cos\theta, \sin\theta), \theta \in [0, \pi[\}. \end{aligned}$$

Chaque droite vectorielle de $\mathbb{R}^2$ intersecte S_+^1 en un unique point univoquement déterminé par celle-ci. Ainsi, les points projectifs de $\mathbb{P}^1(\mathbb{R})$ sont exactement les points de S_+^1. La droite projective réelle coïncide donc avec l'ensemble S_+^1.

Notons également que, pour avoir une représentation topologiquement correcte de la droite projective réelle, le point $(-1, 0)$, qui n'appartient pas à l'ensemble S_+^1 défini ci-dessus, peut être assimilé au point $(1, 0)$, de sorte que la droite projective réelle soit vue comme une courbe fermée.

Exemple 1.2.7. Intéressons-nous à présent au ***plan projectif réel***, *i.e.* à l'espace projectif $\mathbb{P}^2(\mathbb{R})$. A nouveau, nous regardons ici l'ensemble des droites vectorielles de $\mathbb{R}^3$ (*i.e.* l'ensemble des droites de $\mathbb{R}^3$ passant par l'origine du repère), vu comme espace vectoriel.

Considérons le plan π d'équation $z = 1$. Comme illustré sur la Figure 1.3, il est clair que chaque droite vectorielle de $\mathbb{R}^3$ n'appartenant pas au plan α d'équation $z = 0$ intersecte le plan π en exactement un et un seul point.

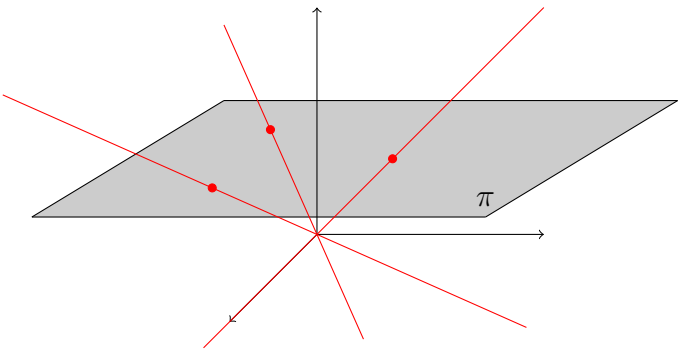

FIGURE 1.3 – Intersections des droites avec π

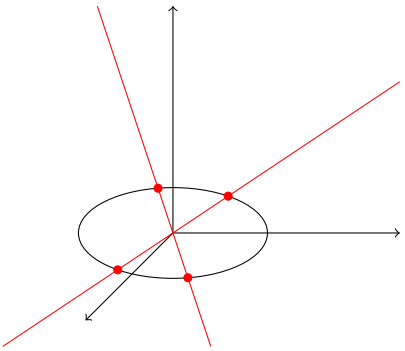

FIGURE 1.4 – Cas des droites du plan $z = 0$

Ainsi, chaque droite vectorielle n'appartenant pas au plan α est caractérisée par un unique point de la forme $(x, y, 1)$. Quant aux droites du plan α (représentées sur la Figure 1.4), on peut procéder comme dans l'Exemple 1.2.6 et associer à chaque droite vectorielle un point de l'ensemble S^1_+ vu ici comme sous-ensemble de $\mathbb{R}^3$, *i.e.* $S^1_+ = \{(\cos\theta, \sin\theta, 0)\,, \theta \in [0, \pi[\}$ dans ce cas. Au final, on peut donc assimiler le plan projectif réel $\mathbb{P}^2(\mathbb{R})$ à l'ensemble

$$\{(x, y, 1) \,|\, x, y \in \mathbb{R}\} \cup S^1_+.$$

1.2.2 Droites et courbes projectives

Comme mentionné précédemment, les courbes elliptiques sont des courbes projectives. Cette section a donc pour but d'introduire cette notion et d'en donner quelques exemples et propriétés.

Définition 1.2.8. Un polynôme $f \in K[x_1, \ldots, x_n]$ est ***homogène de degré*** d si, pour tout $\lambda \in K \setminus \{0\}$,

$$f(\lambda x_1, \ldots, \lambda x_n) = \lambda^d f(x_1, \ldots, x_n).$$

Ainsi, dans un tel polynôme, le degré total de chaque monôme non nul est égal à d.
De plus, le polynôme f est dit ***irréductible*** s'il ne peut s'écrire sous la forme d'un produit non trivial[4] de deux polynômes de $K[x_1, \ldots, x_n]$.

Exemple 1.2.9. Si on considère l'anneau $\mathbb{R}[x, y, z]$ des polynômes à trois variables et à coefficients réels, alors les polynômes f et g définis par

$$f(x, y, z) = x^3 - 2yz^2 + 6x^2y \quad \text{et} \quad g(x, y, z) = 2x^2yz^2 + 7y^4z - xyz^3$$

sont homogènes de degrés 3 et 5 respectivement. Le polynôme f est irréductible, ce qui n'est pas le cas de g. En effet,

$$g(x, y, z) = yz(2x^2z + 7y^3 - xz^2).$$

4. Par produit non-trivial de polynôme, nous entendons un produit de deux polynômes de degré strictement inférieur à celui de f.

Remarque 1.2.10. En particulier, on peut considérer les polynômes homogènes de degré d à n variables comme des fonctions définies sur l'espace projectif $\mathbb{P}^n(K)$. En effet, si f désigne un tel polynôme et si $(x_0, \ldots, x_n) \in K^{n+1}$, alors

$$\begin{aligned} f(x_0, \ldots, x_n) = 0 &\Leftrightarrow \lambda^d f(x_0, \ldots, x_n) = 0, \forall \lambda \in K \setminus \{0\} \\ &\Leftrightarrow f(\lambda x_0, \ldots, \lambda x_n) = 0, \forall \lambda \in K \setminus \{0\}. \end{aligned}$$

Autrement dit, si $(x_0, \ldots, x_n) \in K^{n+1}$ annule le polynôme f, alors tout multiple non nul de ce point l'annule également. Il semble alors naturel d'affirmer que le point projectif de $\mathbb{P}^n(K)$ de coordonnées homogènes $[x_0, \ldots, x_n]$ annule le polynôme f, étant donné que l'annulation d'un polynôme homogène en un point dépend uniquement de la classe à laquelle appartient ce point.

Avant de définir la notion de courbe projective, il convient de rappeler ce que nous entendons lorsque nous parlons de la ***dérivée formelle*** d'un polynôme $f \in K[x_1, \ldots, x_n]$.

Définition 1.2.11. Soit $f \in K[x]$ un polynôme de degré d. Notons

$$f(x) = \sum_{k=0}^{d} a_k x^k, \quad \text{avec } a_d \neq 0.$$

La ***dérivée formelle de f par rapport à la variable*** x, notée f' ou $\dfrac{df}{dx}$, est le polynôme de $K[x]$ défini par

$$\frac{df}{dx}(x) = \sum_{k=1}^{d} k a_k x^{k-1}. \tag{1.1}$$

Si $f \in K[x_1, \ldots, x_n]$ est un polynôme à n variables, alors la ***dérivée formelle de f par rapport à la variable*** x_i $(1 \leq i \leq n)$, notée $\dfrac{df}{dx_i}$, est le polynôme de $K[x_1, \ldots, x_n]$ obtenu selon la formule (1.1) en considérant f comme un polynôme de $K[x_1, \ldots, \widehat{x_i}, \ldots, x_n][x_i]$.

Exemple 1.2.12. Soit $f \in \mathbb{R}[x, y, z]$ le polynôme à trois variables et à coefficients réels défini par

$$f(x, y, z) = x^3 - 2yz^2 + 6x^2y.$$

On a, par définition,

$$\frac{df}{dx}(x, y, z) = 3x^2 + 12xy, \quad \frac{df}{dy}(x, y, z) = -2z^2 + 6x^2 \quad \text{et} \quad \frac{df}{dz}(x, y, z) = -4yz.$$

Bien que, jusqu'ici, nous ayons défini les différentes notions nécessaires dans un cadre général (à savoir, dans des espaces ayant une dimension finie quelconque), nous allons à présent nous concentrer sur l'espace projectif de dimension 2 (*i.e.* $\mathbb{P}^2(K)$), également appelé, de façon assez naturelle, ***plan projectif***.

Commençons tout d'abord par définir les deux principaux objets de cette section, à savoir les droites et les courbes projectives.

Définition 1.2.13. Une ***droite projective*** est un ensemble de points du plan projectif $\mathbb{P}^2(K)$ dont les coordonnées homogènes satisfont une équation de la forme

$$ax + by + cz = 0,$$

avec $a, b, c \in K$ non tous nuls. Autrement dit, une droite projective est un ensemble de la forme

$$\mathscr{D} = \{[x, y, z] \in \mathbb{P}^2(K) \mid ax + by + cz = 0, (a, b, c) \neq (0, 0, 0)\}.$$

Exemple 1.2.14. Les équations

$$2x + 3y = 0 \quad \text{et} \quad x - 6z = 0$$

définissent respectivement deux droites projectives de $\mathbb{P}^2(\mathbb{R})$. Celles-ci s'intersectent en un point de coordonnées homogènes $[6, -4, 1]$.

Définition 1.2.15. Une ***courbe projective plane*** est un ensemble de points du plan projectif $\mathbb{P}^2(K)$ dont les coordonnées homogènes satisfont une équation de la forme

$$f(x, y, z) = 0,$$

avec $f \in K[x, y, z]$ un polynôme homogène de degré $d \geq 1$. Autrement dit, une courbe projective plane est un ensemble de la forme

$$\mathscr{C} = \{[x, y, z] \in \mathbb{P}^2(K) \mid f(x, y, z) = 0, f \in K[x, y, z] \text{ polynôme homogène de degré } d \geq 1\}.$$

Remarque 1.2.16. Dans la définition précédente, si $d = 1$, alors la courbe projective plane est en réalité une droite projective, et on retombe alors sur la Définition 1.2.13.
Si $d = 2$, alors la courbe projective plane est appelée ***conique***, et si $d = 3$, on parlera de ***cubique***.

Il nous reste finalement à introduire deux dernières notions relatives aux courbes projectives avant d'arriver, progressivement, à la définition de courbe elliptique.

Définition 1.2.17. Une courbe projective plane d'équation $f(x, y, z) = 0$ est dite ***irréductible*** si le polynôme homogène f l'est.

Définition 1.2.18. Un point $P \in \mathbb{P}^2(K)$ de coordonnées homogènes $[x_0, y_0, z_0]$ d'une courbe projective plane d'équation $f(x, y, z) = 0$ est dit ***singulier*** si

$$\frac{df}{dx}(x_0, y_0, z_0) = \frac{df}{dy}(x_0, y_0, z_0) = \frac{df}{dz}(x_0, y_0, z_0) = 0.$$

Sinon, P est dit ***non-singulier*** ou ***simple***.
Par extension, une courbe projective plane $\mathscr{C}$ est dite ***non-singulière*** ou ***lisse*** si tous ses points sont simples.

1.2.3 Lien entre courbes affines et projectives, notion de point à l'infini

Dans la pratique, il est peu commode de travailler avec des courbes projectives, notamment car elles sont relativement complexes à représenter. Dès lors, il semble intéressant, étant donnée une courbe projective plane (*i.e.*, de $\mathbb{P}^2(K)$), de trouver son "équivalent" dans l'espace affine à deux dimensions $\mathbb{A}^2(K)$.

Pour ce faire, introduisons quelques résultats utiles.

Théorème 1.2.19. *Pour tout $n \in \mathbb{N}_0$, l'espace affine $\mathbb{A}^n(K)$ peut être plongé dans l'espace projectif $\mathbb{P}^n(K)$.*

Démonstration. Il suffit de considérer l'application

$$\varphi : \mathbb{A}^n(K) \to \mathbb{P}^n(K) : (x_1, \ldots, x_n) \mapsto [x_1, \ldots, x_n, 1].$$

Il est alors aisé de vérifier que cette application est bien injective. □

Exemple 1.2.20. Reprenons l'exemple du plan projectif réel $\mathbb{P}^2(\mathbb{R})$ et de sa description réalisée dans l'Exemple 1.2.7. Comme nous l'avons vu, le plan projectif peut être assimilé à l'ensemble

$$\{(x, y, 1) \,|\, x, y \in \mathbb{R}\} \cup S^1_+.$$

Or, vu la Remarque 1.2.5, l'espace affine $\mathbb{A}^2(\mathbb{R})$ correspond à l'ensemble $\{(x, y) \,|\, x, y \in \mathbb{R}\}$ et est donc en bijection avec l'ensemble $\{(x, y, 1) \,|\, x, y \in \mathbb{R}\}$. Une telle description du plan projectif réel nous montre donc que l'espace $\mathbb{A}^2(\mathbb{R})$ peut être trivialement plongé dans l'espace $\mathbb{P}^2(\mathbb{R})$, ce qui illustre le résultat précédent.

Définition 1.2.21. L'application φ introduite dans la démonstration précédente sera appelée ***plongement canonique*** de $\mathbb{A}^n(K)$ dans $\mathbb{P}^n(K)$.

Remarque 1.2.22. Le Théorème 1.2.19 peut prendre une forme beaucoup plus générale étant donné qu'en réalité, il peut être démontré que tout K-espace affine peut être plongé dans un K-espace projectif de même dimension. Néanmoins, la démonstration de ce résultat ne sera pas faite ici, celle-ci sortant du cadre fixé pour ce travail. Le lecteur intéressé peut consulter [10] pour plus d'informations.

Le théorème précédent nous apprend en particulier que toute courbe affine plane de $\mathbb{A}^2(K)$ peut donc être associée à une courbe projective plane de $\mathbb{P}^2(K)$. L'inverse est-il possible ?

Il est clair que le plongement canonique $\varphi : \mathbb{A}^2(K) \to \mathbb{P}^2(K)$ défini ci-dessus n'est pas une bijection, l'application n'étant pas surjective. Considérons donc l'ensemble

$$U := \left\{[x, y, z] \in \mathbb{P}^2(K) : z \neq 0\right\}.$$

Si on restreint l'ensemble d'arrivée $\mathbb{P}^2(K)$ de l'application φ à l'ensemble U, on obtient une "nouvelle" application $\varphi|_U : \mathbb{A}^2(K) \to U$ qui, elle, est bien bijective ! Ceci nous amène alors au résultat suivant.

Proposition 1.2.23. *L'application*

$$\Phi := \varphi|_U : \mathbb{A}^2(K) \to U : (x, y) \mapsto [x, y, 1]$$

est une bijection dont l'inverse est donnée par

$$\Phi^{-1} : U \to \mathbb{A}^2(K) : [x, y, z] \mapsto \left(\frac{x}{z}, \frac{y}{z}\right).$$

Démonstration. Il suffit de remarquer que, pour tous $x, y \in K$,

$$\Phi^{-1}(\Phi(x, y)) = \Phi^{-1}([x, y, 1]) = \left(\frac{x}{1}, \frac{y}{1}\right) = (x, y).$$

De même, pour tous $x, y, z \in K$,

$$\Phi\left(\Phi^{-1}([x, y, z])\right) = \Phi\left(\frac{x}{z}, \frac{y}{z}\right) = \left[\frac{x}{z}, \frac{y}{z}, 1\right] = [x, y, z].$$

□

Ainsi, étant donnée une courbe projective plane de U, il est possible de trouver une courbe affine plane qui lui est associée dans l'espace affine correspondant $\mathbb{A}^2(K)$.

Néanmoins, il serait illusoire de penser que toutes les courbes projectives planes de $\mathbb{P}^2(K)$ appartiennent à l'ensemble U. En effet, si l'on considère la courbe d'équation

$$x^2 - xz - y^2 - z^2 = 0$$

on peut facilement vérifier que le point $[1, 1, 0]$ n'appartient pas à U, mais bien à la courbe.

Qu'en est-il donc de ces points projectifs dont la troisième coordonnée homogène est nulle ? Ces points, qui ne peuvent être associés à des points affines correspondant dans $\mathbb{A}^2(K)$, ont un statut particulier, qui mérite bien une définition.

Définition 1.2.24. Soit $\mathscr{C}$ une courbe projective plane d'équation $f(x,y,z)=0$. Un point P de $\mathscr{C}$ de coordonnées homogènes $[x_0,y_0,z_0]$ est qualifié de ***point à l'infini*** si $z_0=0$.

Autrement dit, les points à l'infini d'une courbe projective plane n'ont pas d'équivalent dans l'espace affine $\mathbb{A}^2(K)$. Cela permet alors de distinguer deux types de points projectifs sur une courbe de $\mathbb{P}^2(K)$ d'équation $f(x,y,z)=0$:

- les ***points affines***, qui sont des points de la forme $[x,y,z]$ tels que $f(x,y,z)=0$ et $z\neq 0$,
- les ***points à l'infini***, qui sont des points de la forme $[x,y,0]$ tels que $f(x,y,0)=0$.

Ainsi, là où un point affine peut être associé à un réel point de l'espace $\mathbb{A}^2(K)$, un point à l'infini ne peut pas l'être. C'est donc pour éviter cette distinction que certaines courbes sont vues et étudiées comme des courbes projectives et non comme des courbes affines : l'avantage de l'espace projectif $\mathbb{P}^n(K)$ est que, dans cet espace, tous les points ont le même statut !

Afin de bien comprendre cette notion de point à l'infini, et de percevoir le lien entre les espaces projectif et affine, considérons quelques exemples.

Exemple 1.2.25. Considérons la conique projective

$$\mathscr{C}_P \equiv x^2 - xz - y^2 - z^2 = 0.$$

Dans l'espace affine $\mathbb{A}^2(\mathbb{R})$, elle correspond à la courbe

$$\mathscr{C}_A \equiv x^2 - x - y^2 - 1 = 0,$$

qui n'est autre que l'équation d'une hyperbole ! Les points affines vérifiant cette équation forment alors ce que l'on appellera la ***partie affine*** de la courbe $\mathscr{C}_P$.
Pour trouver les éventuels points à l'infini, il faut regarder les points de la courbe projective tels que $z=0$. Dans notre cas, nous cherchons donc les points projectifs $[x,y,0]$ tels que

$$x^2 - y^2 = 0 \Leftrightarrow (x+y)(x-y)=0 \Leftrightarrow x=y \text{ ou } x=-y.$$

Les solutions de cette équation sont donc les triplets de la forme $(x,x,0)$ et $(x,-x,0)$. Par définition des points projectifs et de la relation projective, on en tire que les points à l'infini de $\mathscr{C}_P$ sont les points $[1,1,0]$ et $[1,-1,0]$.

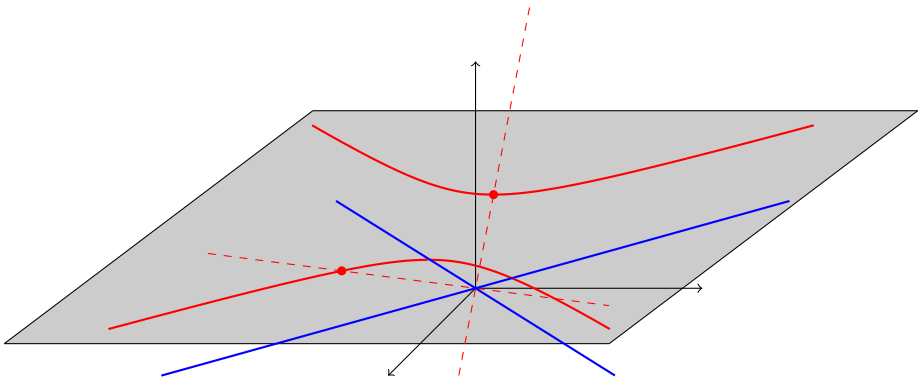

FIGURE 1.5 – Projection de $\mathscr{C}_A$ dans l'espace $\mathbb{A}^2(\mathbb{R})$

Sur la Figure 1.5, la courbe rouge correspond à la conique affine $\mathscr{C}_A$ associée à la courbe projective $\mathscr{C}_P$, ici représentée dans le plan d'équation $z = 1$, assimilé à l'espace affine $\mathbb{A}^2(\mathbb{R})$. Les droites pointillées, elles, sont les droites vectorielles de $\mathbb{R}^3$ représentant deux points projectifs de $\mathbb{P}^2(\mathbb{R})$ satisfaisant l'équation de la conique projective $\mathscr{C}_P$, à savoir les points de coordonnées homogènes

$$[2, -1, 1] \quad \text{et} \quad \left[\frac{1-\sqrt{5}}{2}, 0, 1\right].$$

Enfin, les droites bleues sont les droites vectorielles de $\mathbb{R}^3$ correspondant aux points à l'infini $[1, 1, 0]$ et $[1, -1, 0]$ de la courbe, et qui n'intersectent donc pas le plan $z = 1$. Ainsi, si on ne considère la courbe que dans l'espace affine $\mathbb{A}^2(\mathbb{R})$, ces points seront invisibles. Là est donc tout l'intérêt, dans certains cas, de travailler directement dans l'espace projectif.

Exemple 1.2.26. Considérons à présent une courbe affine

$$\mathscr{C}'_A \equiv x^2 + xy + y^3 - 2 = 0.$$

Pour trouver la courbe projective correspondante, il convient d'***homogénéiser*** l'équation affine [5]. On obtient alors la courbe projective

$$\mathscr{C}'_P \equiv x^2 z + xyz + y^3 - 2z^3 = 0.$$

Ici aussi, cherchons les coordonnées homogènes des points à l'infini de $\mathscr{C}'_P$. Si on annule la variable z, on trouve l'équation

$$y^3 = 0 \Leftrightarrow y = 0.$$

Cette équation est donc satisfaite par les points de la forme $(x, 0, 0)$. Ainsi, le point à l'infini de la courbe projective $\mathscr{C}'_P$ est $[1, 0, 0]$.

Dans le premier exemple, nous avons introduit la notion de ***partie affine*** d'une courbe projective, que nous définissons proprement ci-dessous.

Définition 1.2.27. Étant donnée une courbe projective plane $\mathscr{C}$, la ***partie affine*** de la courbe $\mathscr{C}$ est l'ensemble des points affines de la courbe. Autrement dit, il s'agit de l'ensemble des points de la courbe qui ne sont pas des points à l'infini.

1.3 Les courbes elliptiques

A présent que les bases sont posées, nous pouvons nous permettre de rentrer dans le vif du sujet de ce chapitre : les courbes elliptiques. Nous commencerons par donner la forme la plus générale de l'équation d'une courbe elliptique, mais verrons par la suite qu'il est possible de simplifier celle-ci dans certains cas, ce qui permettra notamment de faciliter considérablement les calculs.

5. Pour rappel, ***homogénéiser*** un polynôme f à n variables consiste à définir un nouveau polynôme g à $n+1$ variables, homogène, en complétant chaque monôme non nul de f par une puissance adéquate de la nouvelle variable. Concrètement, si f est fonction de $(x_1, \ldots, x_n)$, alors g est fonction de $(x_0, x_1, \ldots, x_n)$ avec

$$g(x_0, x_1, \ldots, x_n) = x_0^d \cdot f\left(\frac{x_1}{x_0}, \ldots, \frac{x_n}{x_0}\right),$$

où d est le degré de f.

1.3.1 Définition générale

Définition 1.3.1. Une ***courbe elliptique*** définie sur un champ K, généralement notée [6] E_K, est une cubique projective plane, lisse et irréductible de $\mathbb{P}^2\left(\overline{K}\right)$ dont l'équation est donnée par

$$F(x,y,z) = y^2z + a_1xyz + a_3yz^2 - x^3 - a_2x^2z - a_4xz^2 - a_6z^3 = 0, \tag{1.2}$$

avec $a_1, a_2, a_3, a_4, a_6 \in K$. Une telle équation est appelée ***équation de Weierstrass***.

Remarque 1.3.2. Initialement, une courbe elliptique est simplement définie comme une cubique projective plane, lisse et irréductible, sans restriction sur son équation. Autrement dit, une courbe elliptique est à la base décrite par une équation très générale, de la forme

$$\alpha_1x^3 + \alpha_2x^2y + \alpha_3x^2z + \alpha_4xy^2 + \alpha_5xz^2 + \alpha_6y^3 + \alpha_7y^2z + \alpha_8yz^2 + \alpha_9xyz + \alpha_{10}z^3 = 0, \tag{1.3}$$

avec $\alpha_1, \ldots, \alpha_{10} \in K$. Néanmoins, il est clair qu'il est peu commode de manipuler des équations aussi lourdes.

C'est ici qu'un théorème de géométrie algébrique bien utile intervient : le théorème de Riemann-Roch. Pour être plus précis, ce n'est pas directement ce résultat qui est utilisé, mais une proposition qui en découle [7]. Celle-ci permet en outre d'affirmer que toute courbe elliptique définie par une équation de la forme (1.3) peut être mise sous forme de Weierstrass, *i.e.* décrite par une équation de la forme (1.2). Réciproquement, toute courbe lisse et irréductible définie par une équation de Weierstrass est une courbe elliptique.

Avant d'aller plus loin dans l'étude des courbes elliptiques, arrêtons-nous un instant sur sa partie affine. En utilisant la bijection définie dans la Proposition 1.2.23, nous savons qu'il est possible d'associer à tout point projectif $[x, y, 1]$ un point de l'espace affine correspondant. Ainsi, une courbe elliptique E_K dont l'équation est de la forme (1.2) définit dans l'espace $\mathbb{A}^2(K)$ une courbe affine ayant pour équation

$$f(x,y) = y^2 + a_1xy + a_3y - x^3 - a_2x^2 - a_4x - a_6 = 0. \tag{1.4}$$

L'ensemble des points affines satisfaisant cette équation constitueront alors la partie affine de la courbe elliptique E_K.

Pour déterminer les éventuels points à l'infini de E_K, reprenons l'équation projective (1.2) et déterminons les points projectifs de la forme $[x, y, 0]$ la satisfaisant. Si on annule la variable z, on trouve l'équation

$$-x^3 = 0 \Leftrightarrow x = 0.$$

Comme les points qui satisfont cette équation sont de la forme $(0, y, 0)$, on en déduit que le seul point à l'infini de la courbe elliptique E_K est $[0, 1, 0]$. Nous verrons dans les pages qui suivent que ce point à l'infini joue un rôle essentiel au sein de l'ensemble des points d'une courbe elliptique. Vu son importance et son unicité, nous lui attribuerons la notation $\mathscr{O}$.

Dans la suite, nous définirons principalement une courbe elliptique par son équation affine correspondante, *i.e.* par une équation de la forme (1.4), sans oublier que derrière celle-ci se cache la présence du point à l'infini $\mathscr{O}$. Nous reviendrons à l'équation projective en temps voulu.

6. Dans la suite, nous utiliserons souvent la notation E_K pour désigner tantôt une courbe elliptique, tantôt un ensemble de solutions, en fonction du contexte dans lequel elle est utilisée.

7. Ces résultats ne seront pas détaillés dans le cadre de ce mémoire, mais le lecteur intéressé peut consulter [18] et [47] pour plus d'informations. Le travail [27] décrit également une autre façon d'arriver à la forme de Weierstrass à partir de l'équation générale.

Exemples 1.3.3. Afin d'illustrer la Définition 1.3.1, considérons les courbes elliptiques E_1 et E_2 définies sur le champ $K = \mathbb{R}$ par les équations

$$y^2 = x^3 - x \quad \text{et} \quad y^2 = x^3 - x + 2$$

respectivement. Ces courbes sont représentées ci-dessous dans l'espace affine $\mathbb{A}^2(\mathbb{R})$.

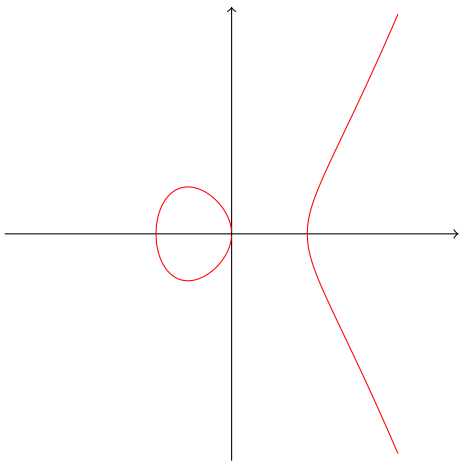

FIGURE 1.6 – Courbe E_1

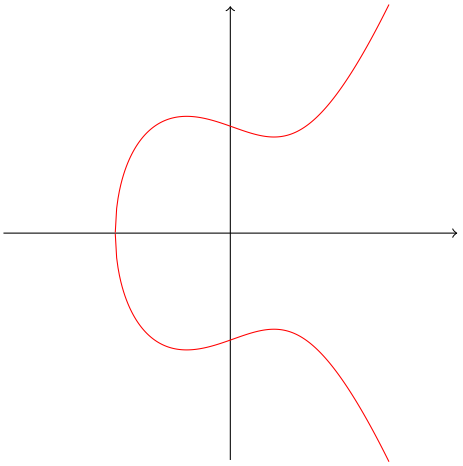

FIGURE 1.7 – Courbe E_2

Exemple 1.3.4. Les courbes elliptiques peuvent également être définies sur des champs d'ordre fini ! En particulier, par la suite, nous nous intéresserons aux champs $\mathbb{F}_{p^n}$, qui, pour rappel, sont les champs finis d'ordre p^n, avec p premier et n un naturel non nul.

Pour l'exemple, considérons la courbe elliptique E définie sur le champ $\mathbb{F}_5$ par l'équation

$$y^2 = x^3 + 3x.$$

Comme le champ considéré est fini, la représentation de la courbe elliptique ne sera pas "continue", comme elle l'était dans les Exemples 1.3.3. Néanmoins, il reste possible de représenter la courbe, comme illustré à la Figure 1.8, où les éléments de $\mathbb{F}_5$ (ici assimilé à $\mathbb{Z}_5$) sont énumérés sur chacun des axes.

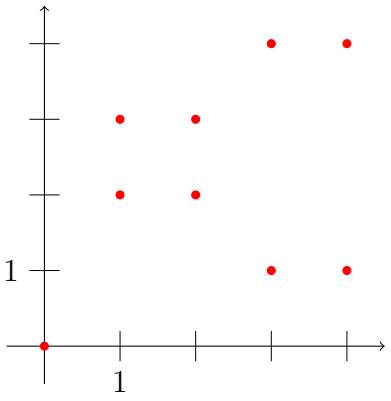

FIGURE 1.8 – Représentation de la courbe E dans $\mathbb{F}_5$.

Le lecteur intéressé par la représentation de courbes elliptiques sur des champs finis peut notamment consulter

`<https://graui.de/code/elliptic2/>`.

1.3.2 Quelques simplifications

Comme nous avons pu le voir dans la section précédente, l'équation décrivant une courbe elliptique comporte beaucoup de paramètres, ce qui a pour conséquence d'alourdir considérablement certains calculs. Néanmoins, si le champ K sur lequel elle est définie possède de "bonnes" propriétés (nous allons voir ce qui est entendu par "bonnes"), celle-ci peut se ramener à une équation parfois beaucoup plus simple. En particulier, nous allons voir que les simplifications possibles de l'équation (1.4) sont fonctions de la caractéristique p du champ sur lequel la courbe elliptique est définie.

Proposition 1.3.5. *Si K est un champ de caractéristique $p \neq 2$, alors l'équation définissant une courbe elliptique E_K sur K peut se mettre sous la forme*

$$y^2 = 4x^3 + b_2x^2 + 2b_4x + b_6. \tag{1.5}$$

Démonstration. Comme E_K est une courbe elliptique sur K, elle est définie par une équation de Weierstrass

$$y^2 + a_1xy + a_3y - x^3 - a_2x^2 - a_4x - a_6 = 0,$$

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

Par complétion du carré sur y, il vient

$$\begin{aligned}
& y^2 + a_1xy + a_3y - x^3 - a_2x^2 - a_4x - a_6 = 0 \\
\Leftrightarrow\ & y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6 \\
\Leftrightarrow\ & \left(y + \frac{(a_1x + a_3)}{2}\right)^2 - \frac{(a_1x + a_3)^2}{4} = x^3 + a_2x^2 + a_4x + a_6 \\
\Leftrightarrow\ & \left(y + \frac{a_1x + a_3}{2}\right)^2 = x^3 + \left(a_2 + \frac{a_1^2}{4}\right)x^2 + \left(a_4 + \frac{2a_1a_3}{4}\right)x + \left(a_6 + \frac{a_3^2}{4}\right) \\
\Leftrightarrow\ & (2y + a_1x + a_3)^2 = 4x^3 + \left(a_1^2 + 4a_2\right)x^2 + 2\left(2a_4 + a_1a_3\right)x + \left(a_3^2 + 4a_6\right).
\end{aligned}$$

Notons que les opérations effectuées ci-dessus ont bien un sens, étant donné que le champ K considéré est de caractéristique différente de 2, et que l'on peut donc sans problème diviser par des puissances de 2.

Ainsi, en effectuant le changement de variables

$$\psi_1 : \mathbb{A}^2(K) \to \mathbb{A}^2(K) : (x, y) \mapsto \left(x, \frac{1}{2}(y - a_1x - a_3)\right),$$

et en posant

$$b_2 = a_1^2 + 4a_2, \tag{1.6}$$
$$b_4 = 2a_4 + a_1a_3, \tag{1.7}$$
$$b_6 = a_3^2 + 4a_6, \tag{1.8}$$

on arrive bien au résultat attendu. □

Avant de poursuivre, nous introduisons une nouvelle quantité qui, de prime abord, peut sembler arbitraire, mais qui permettra d'alléger les notations par la suite. Posons donc

$$b_8 = a_1^2a_6 + 4a_2a_6 - a_1a_3a_4 + a_2a_3^2 - a_4^2. \tag{1.9}$$

Proposition 1.3.6. *Si K est un champ de caractéristique $p \notin \{2, 3\}$, alors l'équation définissant une courbe elliptique E_K sur K peut se mettre sous la forme*

$$y^2 = x^3 - 27c_4x - 54c_6. \tag{1.10}$$

Démonstration. Par la Proposition 1.3.5, on sait que l'équation décrivant E_K est de la forme

$$y^2 = 4x^3 + b_2x^2 + 2b_4x + b_6,$$

avec $b_2, b_4, b_6 \in K$ donnés respectivement en (1.6), (1.7) et (1.8). Considérons le changement de variables

$$\psi_2 : \mathbb{A}^2(K) \to \mathbb{A}^2(K) : (x, y) \mapsto \left(\frac{x - 3b_2}{36}, \frac{y}{108}\right).$$

A nouveau, cette application est bien définie étant donné que 36 et 108 s'écrivent comme des produits de puissances de 2 et 3, qui ne correspondent aucun à la caractéristique du champ K considéré.

L'équation devient alors

$$\begin{aligned}
&\left(\frac{y}{108}\right)^2 = 4\left(\frac{x - 3b_2}{36}\right)^3 + b_2\left(\frac{x - 3b_2}{36}\right)^2 + 2b_4\left(\frac{x - 3b_2}{36}\right) + b_6 \\
\Leftrightarrow\ &\frac{y^2}{11664} = \frac{x^3 - 9b_2x^2 + 27b_2^2x - 27b_2^3}{11664} + \frac{b_2x^2 - 6b_2^2x + 9b_2^3}{1296} + \frac{2b_4x - 6b_2b_4}{36} + b_6 \\
\Leftrightarrow\ &y^2 = x^3 - 9b_2x^2 + 27b_2^2x - 27b_2^3 + 9b_2x^2 - 54b_2^2x + 81b_2^3 + 648b_4x - 1944b_2b_4 + 11664b_6 \\
\Leftrightarrow\ &y^2 = x^3 - 27\left(b_2^2 - 24b_4\right)x - 54\left(-b_2^3 + 36b_2b_4 - 216b_6\right).
\end{aligned}$$

Il suffit alors de poser

$$c_4 = b_2^2 - 24b_4, \tag{1.11}$$
$$c_6 = -b_2^3 + 36b_2b_4 - 216b_6, \tag{1.12}$$

pour conclure. □

Avant de poursuivre notre analyse des courbes elliptiques, introduisons deux notions qui nous permettrons, par la suite, de déterminer facilement si une courbe régie par une équation de Weierstrass est une courbe elliptique ou non.

Définition 1.3.7. Soit $\mathscr{C}$ une courbe affine définie par une équation de Weierstrass, *i.e.* une équation de la forme (1.4). Le ***discriminant*** de $\mathscr{C}$ est la quantité

$$\Delta(\mathscr{C}) = -b_2^2b_8 - 8b_4^3 - 27b_6^2 + 9b_2b_4b_6,$$

où b_2,...,b_8 sont donnés en (1.6)–(1.9). Si celui-ci est non nul, alors on définit le ***j-invariant*** de $\mathscr{C}$ comme étant la quantité

$$j(\mathscr{C}) = \frac{c_4^3}{\Delta},$$

où c_4 est donné en (1.11).

Remarque 1.3.8. Il est également possible de démontrer l'égalité suivante

$$\Delta(\mathscr{C}) = \frac{c_4^3 - c_6^2}{1728}.$$

La preuve se résumant à des calculs "bêtes et méchants", elle ne sera pas détaillée ici. Notons cependant que, même si cette formule semble beaucoup plus légère que celle donnée dans la Définition 1.3.7, elle n'a pas été choisie pour définir le discriminant, en raison du fait que son sens dépend fortement de la caractéristique du champ considéré (en effet, la division par 1728 implique que cette formule n'est valable que pour des champs dont la caractéristique ne divise pas ce nombre, en l'occurrence des champs de caractéristique différente de 2 et 3).

Remarque 1.3.9. Les nombres définis dans la définition précédente peuvent sembler sortis de nulle part, mais ce n'est pas le cas. Arrêtons-nous un instant sur la notion de discriminant. En algèbre, étant donné un polynôme $f \in A[x]$ de degré $n \geq 1$ et dont le coefficient dominant a_n est inversible (ce qui est le cas si A est un anneau intègre [8]), le ***discriminant*** de f est défini par la formule

$$\Delta(f) = \frac{(-1)^{\frac{n(n-1)}{2}}}{a_n} R(f, f'),$$

où $R(f, f')$ est le ***résultant*** de f et f'. Pour information, le ***résultant*** de deux polynômes $f, g \in A[x]$, avec

$$f(x) = \sum_{i=0}^{n} a_i x^i \quad \text{et} \quad g(x) = \sum_{i=0}^{m} b_i x^i,$$

est le déterminant de leur ***matrice de Sylvester***, *i.e.* $R(f, g) = \det M$, avec M la matrice de dimensions $(n+m) \times (n+m)$ suivante :

$$M = \begin{pmatrix} a_n & 0 & \cdots & 0 & b_m & 0 & \cdots & 0 \\ a_{n-1} & a_n & \ddots & \vdots & \vdots & b_m & \ddots & \vdots \\ \vdots & a_{n-1} & \ddots & 0 & \vdots & \vdots & \ddots & 0 \\ \vdots & \vdots & \ddots & a_n & b_1 & & & b_m \\ a_0 & & & a_{n-1} & b_0 & \ddots & \vdots & \vdots \\ 0 & \ddots & & \vdots & 0 & \ddots & b_1 & \vdots \\ \vdots & \ddots & a_0 & \vdots & \vdots & \ddots & b_0 & b_1 \\ 0 & \cdots & 0 & a_0 & 0 & \cdots & 0 & b_0 \end{pmatrix}.$$

Pour contourner la contrainte d'avoir un coefficient dominant inversible, il existe d'autres formules pour le discriminant d'un polynôme. Ainsi, on peut également le définir en termes des racines de f. En effet, si $\alpha_1, \ldots, \alpha_n$ désignent les racines de f, comptées avec leur multiplicité, alors

$$\Delta(f) = a_n^{2n-2} \prod_{i<j} (\alpha_i - \alpha_j)^2.$$

En particulier, cette formule met en lumière une des propriétés majeures du discriminant, que l'on retrouvera d'ailleurs pour les courbes elliptiques : il est nul si et seulement si le polynôme f admet une racine multiple. Par contre, là où il permet de déterminer les racines d'un polynôme du second degré, cela n'est plus le cas pour des polynômes de degré supérieur.

L'appellation "discriminant" utilisée dans la Définition 1.3.7 n'est donc pas innocente. En effet, dans le résultat suivant, nous allons voir que, sous certaines conditions, l'équation de Weierstrass d'une courbe peut se mettre sous la forme

$$y^2 = x^3 + a_4 x + a_6.$$

Dans ce cas particulier, on peut montrer le discriminant de la courbe affine $\mathscr{C}$ définie par cette équation de Weierstrass correspond en réalité, à un facteur multiplicatif près, au discriminant du polynôme f défini par $f(x) = x^3 + a_4 x + a_6$. Ainsi, on comprend aisément pourquoi la même appellation a été assignée à ces deux valeurs a priori différentes.

8. Pour rappel, un anneau $(A, +, \cdot)$ est ***intègre*** si $1_A \neq 0_A$ et s'il ne contient pas de diviseur de 0, *i.e.* $\forall (a, b) \in A^2$, $a \cdot b = 0 \Rightarrow a = 0$ ou $b = 0$.

A travers le résultat suivant, nous allons voir que les deux quantités définies ci-dessus peuvent prendre des formes beaucoup plus simples en fonction de la caractéristique du champ K sur lequel on travaille. Par la même occasion, nous généralisons aussi les Propositions 1.3.5 et 1.3.6 en déterminant les simplifications qui peuvent être effectuées sur une équation de la forme (1.4) dans les différents cas envisagés.

Théorème 1.3.10. *Soit K un champ de caractéristique p. Une courbe $\mathscr{C}$ définie sur K par une équation de Weierstrass de la forme (1.4) peut prendre la forme simplifiée suivante :*

1. *si $p \notin \{2,3\}$,*

$$y^2 = x^3 + a_4x + a_6, \text{ avec } \Delta(\mathscr{C}) = -16\left(4a_4^3 + 27a_6^2\right) \text{ et } j(\mathscr{C}) = 1728\frac{4a_4^3}{4a_4^3 + 27a_6^2},$$

2. *si $p = 2$ et $j(\mathscr{C}) \neq 0$,*

$$y^2 + xy = x^3 + a_2x^2 + a_6, \text{ avec } \Delta(\mathscr{C}) = a_6 \text{ et } j(\mathscr{C}) = \frac{1}{a_6},$$

si $p = 2$ et $j(\mathscr{C}) = 0$,

$$y^2 + a_3y = x^3 + a_4x + a_6, \text{ avec } \Delta(\mathscr{C}) = a_4^3 \text{ et } j(\mathscr{C}) = 0,$$

3. *si $p = 3$ et $j(\mathscr{C}) \neq 0$,*

$$y^2 = x^3 + a_2x^2 + a_6, \text{ avec } \Delta(\mathscr{C}) = -a_2^3a_6 \text{ et } j(\mathscr{C}) = -\frac{a_2^3}{a_6},$$

si $p = 3$ et $j(\mathscr{C}) = 0$,

$$y^2 = x^3 + a_4x + a_6, \text{ avec } \Delta(\mathscr{C}) = -a_4^3 \text{ et } j(\mathscr{C}) = 0.$$

Démonstration.

1. L'expression de la forme simplifiée découle directement de la Proposition 1.3.6, en assimilant respectivement $-27c_4$ et $-54c_6$ aux nouveaux coefficients a_4 et a_6. On se retrouve alors avec une équation de Weierstrass de la forme (1.4), avec $a_1 = a_2 = a_3 = 0$. Dès lors,

$$b_2 = 0, \quad b_4 = 2a_4 \quad \text{et} \quad b_6 = 4a_6,$$

et donc

$$c_4 = -48a_4 \quad \text{et} \quad c_6 = -864a_6.$$

Ceci étant, il vient

$$\Delta(\mathscr{C}) = \frac{c_4^3 - c_6^2}{1728} = \frac{(-48a_4)^3 - (-864a_6)^2}{1728} = -64a_4^3 - 432a_6^2 = -16\left(4a_4^3 + 27a_6^2\right).$$

Aussi,

$$j(\mathscr{C}) = \frac{c_4^3}{\Delta} = \frac{(-48a_4)^3}{-16\left(4a_4^3 + 27a_6^2\right)} = 1728\frac{4a_4^3}{4a_4^3 + 27a_6^2}.$$

2. Comme nous sommes en caractéristique 2, la courbe $\mathscr{C}$ est définie par l'équation de Weierstrass générale

$$y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6.$$

Par définition des coefficients c_4 et b_2, et à nouveau en considérant le fait que le champ K est de caractéristique 2, on a

$$c_4 = b_2^2 - 24b_4 = b_2^2 \quad \text{et} \quad b_2 = a_1^2 + 4a_2 = a_1^2.$$

De là, on a directement

$$j(\mathscr{C}) = \frac{c_4^3}{\Delta} = \frac{\left(b_2^2\right)^3}{\Delta} = \frac{\left(a_1^2\right)^6}{\Delta} = \frac{a_1^{12}}{\Delta}.$$

Notons également que, en caractéristique 2,

$$\Delta(\mathscr{C}) = -b_2^2 b_8 - 8b_4^3 - 27b_6^2 + 9b_2 b_4 b_6 = b_2^2 b_8 + b_6^2 + b_2 b_4 b_6.$$

Deux cas peuvent alors se présenter.

- Cas 1 : $j(\mathscr{C}) \neq 0$

Vu ce qui précède, il vient $a_1 \neq 0$. Dès lors, nous pouvons considérer le changement de variables

$$\mathbb{A}^2(K) \to \mathbb{A}^2(K) : (x, y) \mapsto \left(a_1^2 x + \frac{a_3}{a_1}, a_1^3 y + \frac{a_1^2 a_4 + a_3^2}{a_1^3}\right).$$

Après calculs [9], on tombe alors sur une équation de Weierstrass de la forme annoncée (après avoir rebaptisé les coefficients), *i.e.* une équation avec les paramètres $a_1 = 1$, $a_3 = a_4 = 0$ et $a_2, a_6 \in K$. On tire alors de [13] que

$$\begin{aligned}
b_2 &= a_1^2 + 4a_2 = a_1^2 = 1,\\
b_4 &= 2a_4 + a_1 a_3 = a_1 a_3 = 1 \cdot 0 = 0,\\
b_6 &= a_3^2 + 4a_6 = a_3^2 = 0,\\
b_8 &= a_1^2 a_6 + 4a_2 a_6 - a_1 a_3 a_4 + a_2 a_3^2 - a_4^2 = 1 \cdot a_6 + 0 - 1 \cdot 0 \cdot 0 + a_2 \cdot 0 - 0 = a_6,
\end{aligned}$$

ce qui donne

$$\Delta(\mathscr{C}) = b_2^2 b_8 + b_6^2 + b_2 b_4 b_6 = 1 \cdot a_6 + 0 + 1 \cdot 0 \cdot 0 = -a_6 = a_6.$$

Il vient alors directement

$$j(\mathscr{C}) = \frac{a_1^{12}}{\Delta} = \frac{1}{a_6}.$$

- Cas 2 : $j(\mathscr{C}) = 0$

Dans ce cas, $a_1 = 0$ et l'on considèrera cette fois l'application

$$\mathbb{A}^2(K) \to \mathbb{A}^2(K) : (x, y) \mapsto (x + a_2, y).$$

L'équation de Weierstrass devient alors

$$\begin{aligned}
& y^2 + a_3 y = (x + a_2)^3 + a_2(x + a_2)^2 + a_4(x + a_2) + a_6\\
\Leftrightarrow\ & y^2 + a_3 y = x^3 + 3a_2 x^2 + 3a_2^2 x + a_2^3 + a_2 x^2 + 2a_2^2 x + a_2^3 + a_4 x + a_2 a_4 + a_6\\
\Leftrightarrow\ & y^2 + a_3 y = x^3 + 4a_2 x^2 + (5a_2^2 + a_4)x + 2a_2^3 + a_2 a_4 + a_6\\
\Leftrightarrow\ & y^2 + a_3 y = x^3 + (a_2^2 + a_4)x + a_2 a_4 + a_6,
\end{aligned}$$

qui est bien la forme attendue, en renommant ici également les coefficients.

9. Ceux-ci étant assez lourds (mais pas compliqués), ils ne seront pas détaillés ici afin d'alléger la preuve. Il s'agit simplement de remplacer les variables x et y par leurs nouvelles formes, de développer et puis de regrouper les termes adéquatement. Pour les détails, voir notamment [13].

En considérant donc $a_1 = a_2 = 0$, il vient

$$\begin{aligned} b_2 &= a_1^2 = 0, \\ b_4 &= a_1 a_3 = 0 \cdot a_3 = 0, \\ b_6 &= a_3^2, \end{aligned}$$

d'où $\Delta(\mathscr{C}) = b_2^2 b_8 + b_6^2 + b_2 b_4 b_6 = b_6^2 = a_3^4$.

3. Partant de la Proposition 1.3.5, que nous pouvons appliquer ici étant donné que le champ K est de caractéristique $3 \neq 2$, notons que l'équation définissant la courbe $\mathscr{C}$ peut déjà se mettre sous la forme

$$y^2 = x^3 + a_2 x^2 + a_4 x + a_6,$$

où les coefficients ont été rebaptisés et en tenant compte du fait que $4x^3 = x^3$ dans notre cas.

On a alors

$$c_4 = b_2^2 - 24 b_4 = b_2^2 \quad \text{et} \quad b_2 = a_1^2 + 4a_2 = a_2.$$

Ainsi,

$$j(\mathscr{C}) = \frac{c_4^3}{\Delta} = \frac{\left(b_2^2\right)^3}{\Delta} = \frac{a_2^6}{\Delta}.$$

De plus,

$$\Delta(\mathscr{C}) = -b_2^2 b_8 - 8b_4^3 - 27 b_6^2 + 9 b_2 b_4 b_6 = 2 b_2^2 b_8 + b_4^3.$$

A nouveau, considérons les deux cas suivants.

- <u>Cas 1 :</u> $j(\mathscr{C}) \neq 0$

Puisque $a_2 \neq 0$ ici, considérons le changement de variables

$$\mathbb{A}^2(K) \to \mathbb{A}^2(K) : (x, y) \mapsto \left(x + \frac{a_4}{a_2}, y\right).$$

L'équation ci-dessus prend alors la forme

$$\begin{aligned} & y^2 = \left(x + \frac{a_4}{a_2}\right)^3 + a_2 \left(x + \frac{a_4}{a_2}\right)^2 + a_4 \left(x + \frac{a_4}{a_2}\right) + a_6 \\ \Leftrightarrow\ & y^2 = x^3 + 3\frac{a_4}{a_2} x^2 + 3\frac{a_4^2}{a_2^2} x + \frac{a_4^3}{a_2^3} + a_2 x^2 + 2a_4 x + \frac{a_4^2}{a_2} + a_4 x + \frac{a_4^2}{a_2} + a_6 \\ \Leftrightarrow\ & y^2 = x^3 + a_2 x^2 + 3a_4 x + \frac{2a_4^2}{a_2} + \frac{a_4^3}{a_2^3} + a_6 \\ \Leftrightarrow\ & y^2 = x^3 + a_2 x^2 + \frac{2a_4^2 a_2^2 + a_4^3 + a_6 a_2^3}{a_2^3}, \end{aligned}$$

qui, à nouveau en renommant les coefficient, est bien la forme annoncée.

En considérant alors les paramètres $a_1 = a_3 = a_4 = 0$, on trouve alors

$$\begin{aligned} b_2 &= a_1^2 + 4a_2 = a_2, \\ b_4 &= 2a_4 + a_1 a_3 = 0, \\ b_8 &= a_1^2 a_6 + 4a_2 a_6 - a_1 a_3 a_4 + a_2 a_3^2 - a_4^2 = a_2 a_6, \end{aligned}$$

et donc

$$\Delta(\mathscr{C}) = 2 b_2^2 b_8 + b_4^3 = 2 a_2^3 a_6 = -a_2^3 a_6 \quad \text{et} \quad j(\mathscr{C}) = \frac{a_2^6}{\Delta} = \frac{a_2^6}{-a_2^3 a_6} = -\frac{a_2^3}{a_6}.$$

- Cas 2 : $j(\mathscr{C}) = 0$

Dans ce cas, $a_2 = 0$ et l'équation prend directement la forme annoncée. De plus, comme $a_1 = a_2 = a_3 = 0$, il vient

$$b_2 = a_1^2 + 4a_2 = 0,$$
$$b_4 = 2a_4 + a_1 a_3 = 2a_4,$$

ce qui implique

$$\Delta(\mathscr{C}) = 2b_2^2 b_8 + b_4^3 = 0 + (2a_4)^3 = 8a_4^3 = -a_4^3.$$

□

Remarque 1.3.11. En particulier, le théorème précédent nous apprend que l'équation d'une courbe elliptique réelle peut toujours se mettre sous la forme

$$y^2 = x^3 + px + q,$$

avec $p, q \in \mathbb{R}$. Pour rappel, des exemples de courbes elliptiques réelles ont été donnés dans l'Exemple 1.3.3. Celles-ci avaient pour équations

$$E_1 \equiv y^2 = x^3 - x \quad \text{et} \quad E_2 \equiv y^2 = x^3 - x + 2,$$

et sont redessinées ci-dessous :

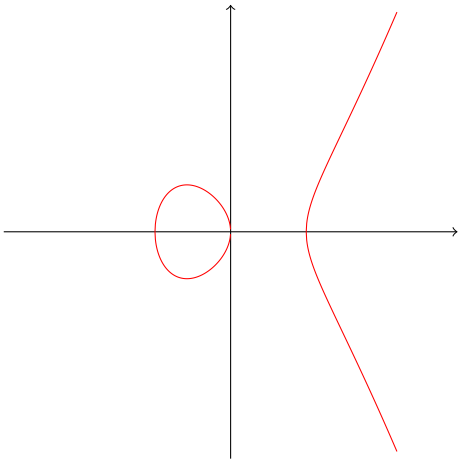

FIGURE 1.9 – Courbe E_1

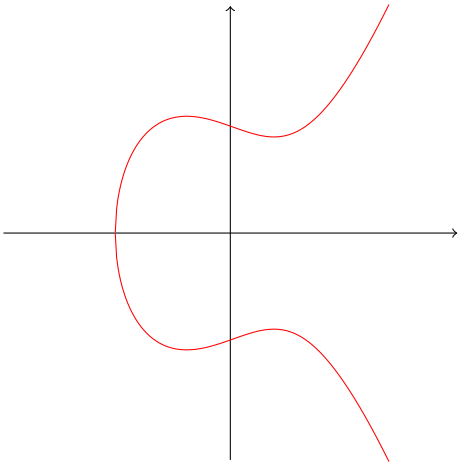

FIGURE 1.10 – Courbe E_2

La remarque précédente nous montre en particulier l'importance des paramètres p et q. En effet, les courbes E_1 et E_2 ne diffèrent que par leur terme indépendant et pourtant, là où E_1 intersecte l'axe des abscisses en 3 points distincts et se présente comme l'union de deux composantes, la courbe E_2 n'admet qu'un seul point d'ordonnée nulle et n'est constituée que d'une seule composante. Il paraît donc opportun d'analyser dans quelle(s) condition(s) se présente l'une ou l'autre situation.

Considérons donc une courbe $\mathscr{C}$, définie sur un champ de caractéristique différente de 2 et 3, et dont l'équation de Weierstrass est donnée par

$$y^2 = x^3 + px + q.$$

Vu le Théorème 1.3.10, son discriminant est donné par la formule

$$\Delta(\mathscr{C}) = -16\left(4p^3 + 27q^2\right).$$

Comme l'auteur de [13], d'aucuns verront ici un lien avec la célèbre formule de Cardan. En effet, en 1545, Gerolamo CARDANO (1501 – 1576), dit Jérôme Cardan en français, expose dans son ouvrage *Ars Magna* une méthode permettant de résoudre une équation du troisième degré $y^3 + ay^2 + by + c = 0$, avec $a, b, c \in \mathbb{R}$. Plus exactement, il s'intéresse aux équations de la forme [10] $x^3 + px + q = 0$, avec $p, q \in \mathbb{R}$.

Sans entrer dans les détails (le lecteur intéressé peut se référer à [17] et [40] pour plus d'informations), Cardan se ramène à une équation du second degré, dont le discriminant est donné par [11]

$$\Delta' = \frac{4p^3 + 27q^2}{27} = \frac{4p^3}{27} + q^2,$$

expression qui n'est pas sans rappeler le discriminant de la courbe $\mathscr{C}$ considérée précédemment. En effet, on peut remarquer que

$$\Delta(\mathscr{C}) = -16\left(4p^3 + 27q^2\right) = -432\left(\frac{4p^3}{27} + q^2\right) = -432\Delta'.$$

Autrement dit, $\Delta(\mathscr{C})$ et Δ' sont soit simultanément nuls, soit de signes contraires. En particulier, nous pouvons reprendre ici les résultats obtenus par Cardan (ainsi que par les autres mathématiciens ayant travaillé sur le sujet), en veillant bien à les inverser par rapport au signe du discriminant de la courbe $\mathscr{C}$. Ainsi,

- si $\Delta(\mathscr{C}) > 0$, alors $\Delta' < 0$. Dans ce cas, l'équation $x^3 + px + q = 0$ admet exactement trois solutions réelles distinctes. On se retrouve alors dans une situation similaire à la courbe E_1 de l'Exemple 1.3.3 (pour laquelle $\Delta(E_1) = 64$), qui rencontre l'axe des abscisses en trois points distincts et est donc constituée de deux composantes ;
- si $\Delta(\mathscr{C}) < 0$, alors $\Delta' > 0$. Ici, l'équation $x^3 + px + q = 0$ admet une solution réelle et deux solutions complexes conjuguées. On est alors dans la même situation que la courbe E_2 de l'Exemple 1.3.3 (pour laquelle $\Delta(E_2) = -1664$), qui est formée d'une seule composante et n'intersecte l'axe des abscisses qu'en un seul point ;
- si $\Delta(\mathscr{C}) = 0$, alors $\Delta' = 0$ et l'équation $x^3 + px + q = 0$ admet des solutions réelles, dont une solution double x_0. Dans ce cas, la courbe $\mathscr{C}$ n'est pas une courbe elliptique, étant donné qu'elle est singulière (en effet, le point de coordonnées $(x_0, 0)$ annulera les dérivées partielles du polynôme $f(x, y) = y^2 - x^3 - px - q$).

Nous allons à présent fournir une caractérisation des courbes non-singulières et donc, par extension, des courbes elliptiques. En effet, en pratique, il est souvent fastidieux de vérifier qu'une courbe projective plane est lisse, étant donné qu'il faut s'assurer que chacun des points de la courbe est non-singulier, ce qui rend la Définition 1.3.1 peu praticable. Ce résultat nous permettra alors, par la suite, de vérifier directement si une courbe définie par une équation de Weierstrass est une courbe elliptique ou non.

Avant toute chose, introduisons quelques résultats préliminaires afin d'alléger la preuve du théorème qui suivra.

---

10. Rappelons que toute équation de la forme $y^3 + ay^2 + by + c = 0$ peut être mise sous la forme $x^3 + px + q = 0$ en posant $x = y + \frac{a}{3}$. Notons que ce changement de variable est défini uniquement en caractéristique différente de 3.
Pour la parenthèse historique, notons également qu'à cette époque, Cardan n'a pas directement analysé l'équation $x^3 + px + q = 0$, mais les équations analogues $x^3 = px + q$, $x^3 + px = q$ et $x^3 + q = px$, étant donné qu'on n'égalait entre elles que des quantités strictement positives. De plus, on ne considérait également que des coefficients $p, q > 0$.

11. Nous insistons ici sur le fait qu'il s'agit bien du discriminant le l'équation du second degré à laquelle Cardan arrive, et non le discriminant de l'équation de départ, même si ceux-ci sont évidemment liés.

Lemme 1.3.12. *Soit E une courbe définie sur un champ K par une équation de Weierstrass générale de la forme*

$$y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6.$$

Le discriminant $\Delta(E)$ est invariant sous le changement de variables

$$\psi_{r,s} : \mathbb{A}^2(K) \to \mathbb{A}^2(K) : (x, y) \mapsto (x + r, y + s),$$

pour tout $r, s \in K$.

Démonstration. En effectuant le changement de variables dans l'équation de Weierstrass, on obtient

$$\begin{aligned}
&(y+s)^2 + a_1(x+r)(y+s) + a_3(y+s) = (x+r)^3 + a_2(x+r)^2 + a_4(x+r) + a_6 \\
\Leftrightarrow\ & y^2 + 2sy + s^2 + a_1xy + a_1sx + a_1ry + a_1rs + a_3y + a_3s \\
&\qquad = x^3 + 3rx^2 + 3r^2x + r^3 + a_2x^2 + 2a_2rx + a_2r^2 + a_4x + a_4r + a_6 \\
\Leftrightarrow\ & y^2 + a_1xy + (a_3 + a_1r + 2s)y = x^3 + (a_2 + 3r)x^2 \\
&\qquad + (a_4 + 2a_2r - a_1s + 3r^2)x + (a_6 + a_4r + a_2r^2 + r^3 - a_1rs - a_3s - s^2).
\end{aligned}$$

Notons E' la courbe définie par cette équation. En posant

$$\begin{aligned}
a'_1 &= a_1, \\
a'_2 &= a_2 + 3r, \\
a'_3 &= a_3 + a_1r + 2s, \\
a'_4 &= a_4 + 2a_2r - a_1s + 3r^2, \\
a'_6 &= a_6 + a_4r + a_2r^2 + r^3 - a_1rs - a_3s - s^2,
\end{aligned}$$

on trouve alors [12]

$$\begin{aligned}
b'_2 &= a'^2_1 + 4a'_2 = b_2 + 12r, \\
b'_4 &= 2a'_4 + a'_1a'_3 = b_4 + rb_2 + 6r^2, \\
b'_6 &= a'^2_3 + 4a'_6 = b_6 + 2rb_4 + r^2b_2 + 4r^3, \\
b'_8 &= a'^2_1a'_6 + 4a'_2a'_6 - a'_1a'_3a'_4 + a'_2a'^2_3 - a'^2_4 = b_8 + 3rb_6 + 3r^2b_4 + r^3b_2 + 3r^4.
\end{aligned}$$

De même, il vient

$$\begin{aligned}
c'_4 &= b'^2_2 - 24b'_4 = c_4, \\
c'_6 &= -b'^3_2 + 36b'_2b'_4 - 216b'_6 = c_6,
\end{aligned}$$

ce qui implique finalement [13] $\Delta(E') = \Delta(E)$, ce qu'on voulait. □

Lemme 1.3.13. *Soit K un champ de caractéristique $p > 0$. L'application*

$$f : K \to K : x \mapsto x^p$$

est un homomorphisme de champs, appelé **homomorphisme de Frobenius**. *Si, de plus, K est fini, alors c'est un automorphisme.*

12. A nouveau, les calculs étant longs et relativement lourds, ils ne seront pas détaillés ici. Il s'agit néanmoins de "simples" substitutions, suivies de l'application des définitions des différents coefficients déjà connus.

13. La conclusion découle ici de l'expression du discriminant dans la Remarque 1.3.8. Pour les champs de caractéristique 2 ou 3, il convient de considérer la formule de la Définition 1.3.7.

Démonstration. Soient $x, y \in K$. Tout d'abord, il est clair que $f(xy) = f(x)f(y)$. De plus, en utilisant le binôme de Newton, il vient

$$(x+y)^p = \sum_{i=0}^{p} C_p^i x^{p-i} y^i = x^p + \sum_{i=1}^{p-1} C_p^i x^{p-i} y^i + y^p.$$

Or, pour tout $1 \leq i \leq p-1$, le coefficient binomial C_p^i est un multiple de p, et donc $C_p^i x^{p-i} y^i = 0$ puisque K est de caractéristique p. Ainsi,

$$f(x+y) = (x+y)^p = x^p + y^p = f(x) + f(y).$$

Comme $f(0) = 0$ et $f(1) = 1$, il s'ensuit que f est bien un homomorphisme.

Pour montrer que f est bijectif (et donc un automorphisme) dans le cas où K est fini, notons qu'un homomorphisme de champs est toujours injectif. En effet, cela découle de trois résultats :

1. le noyaux d'un homomorphisme est un idéal de l'ensemble de départ ;
2. un anneau est un champ si et seulement s'il possède exactement deux idéaux, à savoir K et $\{0\}$;
3. un homomorphisme est injectif si et seulement si son noyau se réduit au singleton $\{0\}$ (où 0 est le neutre pour l'addition).

En combinant les deux premiers résultats, on trouve que $\ker f = \{0\}$ ou $\ker f = K$. Or, $f(1) = 1 \neq 0$, d'où $\ker f \neq K$. Le dernier résultat permet alors de conclure pour l'injectivité de f. Enfin, comme on a une injection d'un champ fini dans lui-même, on en tire directement qu'il s'agit d'une bijection, ce qu'on voulait montrer. □

Corollaire 1.3.14. *Dans un champ fini de caractéristique* 2*, tout élément est un carré (*i.e. *le produit d'un élément du champ avec lui-même). De même, dans un champ fini de caractéristique* 3*, tout élément est un cube.*

Démonstration. Cela découle directement du Lemme 1.3.13. En effet, puisque l'homomorphisme de Frobenius est un automorphisme de K, tout élément du champ est image d'un élément par cette application. □

Remarque 1.3.15. Notons que, lorsque nous définirons une courbe elliptique sur un champ de caractéristique finie, il s'agira toujours d'un champ $\mathbb{F}_q$ (où, pour rappel, q est une puissance d'un nombre premier p, ce dernier correspondant à la caractéristique), et donc d'un champ fini. Ainsi, les résultats précédents pourront être appliqués.

Nous pouvons à présent passer à un des théorèmes majeurs de cette section.

Théorème 1.3.16. *Soit K un champ de caractéristique p. Une courbe E définie sur K par une équation de Weierstrass est non-singulière (ou lisse), si et seulement si $\Delta(E) \neq 0$.*

Démonstration. Soit E une courbe plane définie par l'équation de Weierstrass générale

$$f(x,y) := y^2 + a_1xy + a_3y - x^3 - a_2x^2 - a_4x - a_6 = 0.$$

Montrons tout d'abord que le point à l'infini $\mathcal{O}$ n'est jamais singulier. Pour ce faire, replongeons-nous dans l'espace projectif $\mathbb{P}^2(K)$ et considérons l'équation de Weierstrass homogénéisée

$$F(x,y,z) := y^2z + a_1xyz + a_3yz^2 - x^3 - a_2x^2z - a_4xz^2 - a_6z^3 = 0.$$

On a alors

$$\frac{df}{dz}(x,y,z) = y^2 + a_1xy + 2a_3yz - a_2x^2 - 2a_4xz - 3a_6z^2.$$

Évalué en $\mathscr{O}$, on obtient

$$\frac{df}{dz}(0,1,0) = 1 \neq 0,$$

ce qui permet de conclure.

Pour démontrer le théorème, nous allons montrer que la courbe E est singulière, si et seulement si $\Delta(E) = 0$. La conclusion sera alors immédiate. Commençons par montrer que la condition est nécessaire, et supposons que la courbe E est singulière en un point P de coordonnées (x_0, y_0). Vu le Lemme 1.3.12, le changement de variables

$$(x, y) \mapsto (x + x_0, y + y_0)$$

laisse $\Delta(E)$ invariant. On peut donc supposer, sans perte de généralité, que E est singulière au point de coordonnées $(0, 0)$.
Dès lors, on a

$$a_6 = -f(0,0) = 0, \quad a_4 = -\frac{df}{dx}(0,0) = 0 \quad \text{et} \quad a_3 = \frac{df}{dy}(0,0) = 0.$$

Ainsi, $b_4 = b_6 = b_8 = 0$ et donc, par définition, $\Delta(E) = 0$, ce qu'on voulait.

Pour montrer que la condition est également suffisante, nous allons considérer les différents cas déjà envisagés dans le Théorème 1.3.10.

- Cas 1 : $p \notin \{2, 3\}$

Dans ce cas, l'équation de Weierstrass définissant E peut se mettre sous la forme

$$f(x, y) = y^2 - x^3 - a_4 x - a_6 = 0,$$

et $\Delta(\mathscr{C}) = -16(4a_4^3 + 27a_6^2)$. Vu les conditions imposées à la caractéristique du champ K, nous pouvons utiliser la méthode de Cardan, et en particulier calculer le discriminant Δ' du polynôme $g(x) = x^3 + a_4 x + a_6$.
Par définition, $P(x_0, y_0)$ est un point singulier de E si

$$\begin{cases} f(x_0, y_0) = 0 \\ \dfrac{df}{dx}(x_0, y_0) = 0 \\ \dfrac{df}{dy}(x_0, y_0) = 0 \end{cases} \Leftrightarrow \begin{cases} x_0^3 + a_4 x_0 + a_6 = 0 \\ 3x_0^2 + a_4 = 0 \\ y_0 = 0 \end{cases}.$$

Autrement dit, les points singuliers de E sont les points de coordonnées $(x_0, 0)$, avec x_0 une racine double du polynôme g (puisque x_0 doit annuler g et sa dérivée). Il reste donc à montrer qu'un tel point existe si $\Delta(E) = 0$.
Vu le raisonnement qui suit la Remarque 1.3.11, on sait que $\Delta(E) = -432\Delta'$. Vu notre hypothèse, on a donc $\Delta' = 0$. Or, d'après la méthode de Cardan, cela signifie que le polynôme g admet deux racines, dont une racine double x_0. Le point de coordonnées $(x_0, 0)$ est donc un point singulier de E, ce qui permet de conclure.

- Cas 2 : $p = 2, j(E) \neq 0$

Dans ce cas, l'équation de Weierstrass définissant E peut se mettre sous la forme

$$f(x, y) = y^2 + xy - x^3 - a_2 x^2 - a_6 = 0,$$

avec $\Delta(E) = a_6$. Ainsi, $\Delta(E) = 0 \Leftrightarrow a_6 = 0$. L'équation peut donc en réalité se ré-écrire

$$f(x, y) = y^2 + xy - x^3 - a_2 x^2 = 0.$$

On remarque alors assez facilement que $O(0,0)$ est un point singulier de la courbe, étant donné que

$$f(0,0) = \frac{df}{dx}(0,0) = \frac{df}{dy}(0,0) = 0.$$

- Cas 3 : $p = 2, j(E) = 0$

Dans ce cas, l'équation de Weierstrass définissant E peut se mettre sous la forme

$$f(x,y) = y^2 + a_3 y - x^3 - a_4 x - a_6 = 0,$$

et $\Delta(E) = a_4^3$. Donc, $\Delta(E) = 0 \Leftrightarrow a_4 = 0$, et l'équation devient

$$f(x,y) = y^2 + a_3 y - x^3 - a_6 = 0.$$

Le point $P(x_0, y_0)$ est singulier si

$$\begin{cases} f(x_0,y_0) = 0 \\ \dfrac{df}{dx}(x_0,y_0) = 0 \\ \dfrac{df}{dy}(x_0,y_0) = 0 \end{cases} \Leftrightarrow \begin{cases} y_0^2 + a_3 y_0 - x_0^3 - a_6 = 0 \\ 3x_0^2 = 0 \\ 2y_0 + a_3 = 0 \end{cases} \Leftrightarrow \begin{cases} y_0^2 = a_6 \\ x_0 = 0 \\ a_3 = 0 \end{cases}.$$

Les points singuliers de E sont donc, s'ils existent, les points de coordonnées $(0, y_0)$, où $y_0^2 = a_6$. Autrement dit, la courbe E admet un point singulier si et seulement si a_6 est un carré, ce qui est le cas vu le Corollaire 1.3.14.

- Cas 4 : $p = 3, j(E) \neq 0$

Dans ce cas, l'équation de Weierstrass définissant E peut se mettre sous la forme

$$f(x,y) = y^2 - x^3 - a_2 x^2 - a_6 = 0,$$

avec $\Delta(E) = -a_2^3 a_6$. Comme $\Delta(E) = 0$ et que K est un champ, et donc intègre, cela implique $a_2 = 0$ ou $a_6 = 0$. Or, si $a_2 = 0$, alors $j(E) = -\dfrac{a_2^3}{a_6} = 0$, ce qui contredit les conditions dans lesquelles on se place. On en tire donc que $a_6 = 0$, et l'équation de Weierstrass devient alors

$$f(x,y) = y^2 - x^3 - a_2 x^2 = 0.$$

On remarque aisément que le point $O(0,0)$ est singulier, ce qui permet de conclure.

- Cas 5 : $p = 3, j(E) = 0$

Dans ce cas, l'équation de Weierstrass définissant E peut se mettre sous la forme

$$f(x,y) = y^2 - x^3 - a_4 x - a_6 = 0,$$

et $\Delta(E) = -a_4^3$. Comme $\Delta(E) = 0$, on a $a_4 = 0$. On peut donc ré-écrire l'équation sous la forme

$$f(x,y) = y^2 - x^3 - a_6 = 0.$$

A nouveau, cherchons la forme des points singuliers éventuels de E. Soit $P(x_0, y_0)$ un point singulier. On doit avoir

$$\begin{cases} f(x_0,y_0) = 0 \\ \dfrac{df}{dx}(x_0,y_0) = 0 \\ \dfrac{df}{dy}(x_0,y_0) = 0 \end{cases} \Leftrightarrow \begin{cases} y_0^2 - x_0^3 - a_6 = 0 \\ -3x_0^3 = 0 \\ 2y_0 = 0 \end{cases} \Leftrightarrow \begin{cases} x_0^3 = -a_6 \\ 0 = 0 \\ y_0 = 0 \end{cases}.$$

Les éventuels points singuliers de E sont donc les points de coordonnées $(x_0, 0)$, où $x_0^3 = -a_6$. En d'autres termes, la courbe est singulière si $-a_6$ est un cube, ce qui est assuré par le Corollaire 1.3.14. □

Corollaire 1.3.17. *Une courbe E définie sur un champ K par une équation de Weierstrass est une courbe elliptique si et seulement si elle est irréductible et de discriminant non nul.*

Démonstration. Cela découle directement de la Définition 1.3.1 et du Théorème 1.3.16. □

Le théorème précédent nous fournit donc un critère particulièrement utile pour déterminer si une courbe affine plane est ou non une courbe elliptique, et cela directement à partir de son équation de Weierstrass.

Exemples 1.3.18. Si on considère le champ $\mathbb{R}$ ainsi que les deux courbes

$$E_1 \equiv y^2 = x^3 - 3x + 2 \quad \text{et} \quad E_2 \equiv y^2 = x^3 + 5x + 1.$$

Leurs discriminants valent respectivement $\Delta(E_1) = 0$ et $\Delta(E_2) = -8432$. Ainsi, E_2 est une courbe elliptique, mais pas E_1.
Pour prendre l'exemple d'un champ fini, considérons $\mathbb{F}_9$, qui est donc de caractéristique 3. Soient alors les courbes

$$E_3 \equiv y^2 = x^3 + 8x + 4 \quad \text{et} \quad E_4 \equiv y^2 = x^3 + 7.$$

Après calculs, on trouve que $\Delta(E_3) = 1$ et $\Delta(E_4) = 0$, ce qui permet d'affirmer que E_3 est une courbe elliptique, mais pas E_4.

Remarque 1.3.19. Nous ne nous attarderons pas plus que cela sur le cas $\Delta(\mathscr{C}) = 0$, étant donné qu'il ne correspond pas à des courbes elliptiques. Néanmoins, il est intéressant d'observer le type de courbes que nous obtenons dans ces cas-là.
Plaçons-nous dans le champ $\mathbb{R}$ et considérons les courbes

$$\mathscr{C}_1 \equiv y^2 = x^3 \quad \text{et} \quad \mathscr{C}_2 \equiv y^2 = x^3 - 3x + 2.$$

On vérifie directement que leurs discriminants respectifs sont nuls et donc, a fortiori, que ce ne sont pas des courbes elliptiques réelles. Néanmoins, nous pouvons les représenter dans le plan affine :

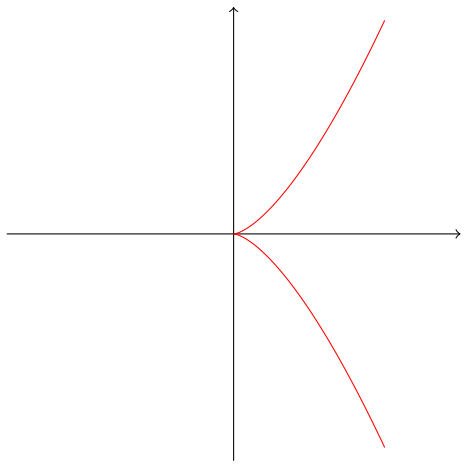

FIGURE 1.11 – Courbe $\mathscr{C}_1$

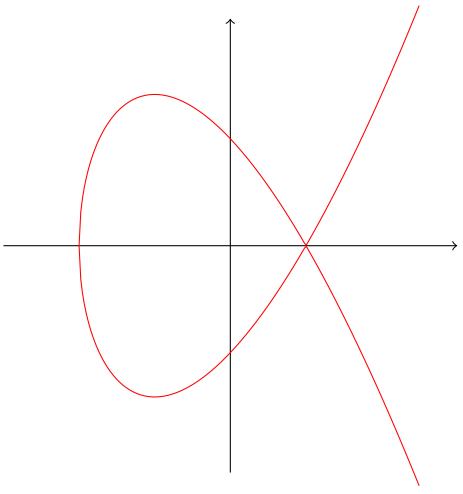

FIGURE 1.12 – Courbe $\mathscr{C}_2$

Ces deux courbes illustrent les deux types de courbes "non-elliptiques" que l'on peut rencontrer. Dans le cas de la courbe $\mathscr{C}_1$, on observe ce qu'on appelle un ***point de rebroussement*** au point de coordonnées $(0,0)$. Cette singularité découle du fait que 0 est une racine triple du polynôme $f(x) = x^3$. Quant à la courbe $\mathscr{C}_2$, elle présente un ***nœud*** au point de coordonnées $(1,0)$. Ici, cela vient du fait que le polynôme $g(x) = x^3 - 3x + 2$ admet une racine double (à savoir 1), ce qui peut se voir en le factorisant comme $g(x) = (x-1)^2(x+2)$.

1.4 Loi de groupe

Avant de nous lancer concrètement dans l'univers de la cryptographie, il nous reste à explorer une facette des courbes elliptiques, et pas des moindres : la structure de groupe formée par les points d'une courbe elliptique. Comme nous le verrons dans le chapitre suivant, c'est d'ailleurs cette propriété des courbes elliptiques qui a rendu leur utilisation en cryptographie possible ! Il semble donc judicieux de s'y arrêter un instant.

Dans cette section, nous nous replongerons en temps voulu dans l'espace projectif $\mathbb{P}^2(K)$, et considèrerons donc des courbes définies par une équation de Weierstrass de la forme (1.2), que nous rappelons ici :

$$F(x,y,z) = y^2z + a_1xyz + a_3yz^2 - x^3 - a_2x^2z - a_4xz^2 - a_6z^3 = 0.$$

La section sera découpée en plusieurs parties. Dans un premier temps, nous nous intéresserons aux champs de caractéristique nulle en prenant l'exemple des courbes elliptiques réelles. Nous nous placerons ensuite dans un cadre plus général, avant de nous arrêter sur le cas des champs finis.

1.4.1 Les courbes elliptiques réelles

Nous nous intéressons ici au champ des réels $\mathbb{R}$, et non aux champs de caractéristique nulle en général, simplement car il est beaucoup plus facile de représenter des courbes elliptiques réelles. Néanmoins, des constructions similaires peuvent être réalisées si l'on souhaite obtenir une loi de composition pour des courbes elliptiques définies sur $\mathbb{Q}$ ou encore sur $\mathbb{C}$.

Rappelons tout d'abord que, vu le Théorème 1.3.10, l'équation d'une courbe elliptique réelle dans l'espace affine $\mathbb{A}^2(\mathbb{R})$ peut être réduite à la forme

$$y^2 = x^3 + px + q, \tag{1.13}$$

avec la condition $4p^3 + 27q^2 \neq 0$.

Avant de définir la loi de composition sur les points d'une courbe elliptique, quelques résultats et définitions préliminaires sont nécessaires.

Lemme 1.4.1. *Soient $P[x_P, y_P, z_P]$ et $Q[x_Q, y_Q, z_Q]$ deux points distincts du plan projectif $\mathbb{P}^2(K)$, où K est un champ. Il existe une unique droite projective $\mathscr{D}$ de $\mathbb{P}^2(K)$ (cf. Définition 1.2.13) passant par les points P et Q.*

Démonstration. Considérons l'application linéaire

$$T : K^3 \to K^2 : \begin{pmatrix} x \\ y \\ z \end{pmatrix} \mapsto \begin{pmatrix} x_P & y_P & z_P \\ x_Q & y_Q & z_Q \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix},$$

où K^3 et K^2 sont vus comme des K-vectoriels. Pour alléger les notations, notons également A la matrice des coordonnées des points P et Q donnée ci-dessus.

Comme P et Q sont distincts, leurs coordonnées homogènes n'appartiennent pas à la même classe d'équivalence au regard de la relation projective définie dans la Définition 1.2.1. Dès lors, les lignes de la matrice A sont linéairement indépendantes, ce qui implique $\mathrm{rg}A = 2$ par définition du rang d'une matrice. Dès lors, par le théorème du rang (ou de la dimension) [14], il vient

$$\dim K^3 = \dim \ker T + \mathrm{rg}T \Leftrightarrow \dim \ker T = \dim K^3 - \mathrm{rg}T = 3 - 2 = 1.$$

Autrement dit, le noyau de T est de dimension 1, *i.e.* il existe $(a, b, c) \in K^3$ tel que

$$\ker T = \left\langle \begin{pmatrix} a \\ b \\ c \end{pmatrix} \right\rangle = \{\lambda(a, b, c) \mid \lambda \in K\}.$$

En particulier, tous les points de $\ker T$ correspondent au même point projectif $[a, b, c] \in \mathbb{P}^2(K)$. Dès lors, nous venons donc de prouver qu'il existe un unique triplet $[a, b, c] \in \mathbb{P}^2(K)$ tel que

$$T(a, b, c) = 0 \Leftrightarrow \begin{pmatrix} ax_P + by_P + cz_P \\ ax_Q + by_Q + cz_Q \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix},$$

i.e. tel que la droite $\mathscr{D}$ d'équation $ax + by + cz = 0$ passe par les points P et Q. □

Lemme 1.4.2. *Soit E une courbe elliptique réelle définie par une équation de la forme (1.13). Si $(x_0, y_0) \in \mathbb{R}^2$ est un point de E, alors $(x_0, -y_0)$ aussi.*

Démonstration. C'est immédiat. Il suffit de remarquer que

$$(-y_0)^2 = y_0^2 = x_0^3 + px_0 + q.$$

□

En particulier, le lemme précédent nous permet de donner un sens à la définition suivante.

Définition 1.4.3. Soit E une courbe elliptique réelle et soit P un point de E de coordonnées (x_P, y_P). On définit l'***opposé*** du point P, noté $-P$, comme étant

1. le point à l'infini $\mathscr{O}$ si $P = \mathscr{O}$,
2. le point de coordonnées $(x_P, -y_P)$ sinon.

Enfin, nous admettrons un résultat dû au mathématicien français Etienne Bézout, qui va nous permettre de donner du sens à la loi de composition définie sur l'ensemble des points d'une courbe elliptique.

Théorème 1.4.4 (Théorème de Bézout, version forte)**.** *Soit K un champ algébriquement clos. Soient $\mathscr{C}_1$ et $\mathscr{C}_2$ deux courbes projectives planes (*i.e. *deux courbes de $\mathbb{P}^2(K)$) dont les équations sont définies respectivement par deux polynômes homogènes f et g. Notons m et n les degrés respectifs de ces polynômes. Si les deux courbes n'ont pas de composante irréductible [15] commune, alors elles ont exactement mn points d'intersection comptés avec leur multiplicité.*

Le lecteur intéressé par ce théorème et sa démonstration peut consulter [5], [13] ou encore [14] pour plus d'informations.

14. Soient E et F deux espaces vectoriels, avec E de dimension finie. Si $T : E \to F$ est une application linéaire, alors

$$\dim E = \dim \ker T + \mathrm{rg}T,$$

où $\mathrm{rg}T$ est la dimension de l'image de T, qui correspond aussi au rang de la matrice représentant T dans une base donnée.

15. Par composante irréductible d'une courbe projective plane définie par un polynôme homogène f, on entend une courbe projective plane associée à un facteur irréductible de f.

Remarque 1.4.5. Comme le nom du théorème le laisse sous-entendre, il existe une version faible du résultat précédent. Celle-ci se passe de l'hypothèse d'un champ algébriquement clos, et se contente de courbes algébriques affines et non projectives. La conséquence est alors qu'on peut simplement affirmer que les deux droites ont au plus mn points d'intersection.

Comme corollaire immédiat, nous pouvons énoncer le résultat suivant.

Corollaire 1.4.6. *Une courbe elliptique E définie sur un champ K et une droite projective $\mathscr{D}$ de $\mathbb{P}^2\left(\overline{K}\right)$ ont exactement 3 points d'intersection, comptés avec leur multiplicité.*

Démonstration. Cela découle directement du Théorème de Bézout et du fait qu'une courbe elliptique est une cubique. □

Enfin, nous avons besoin d'une dernière notion avant de définir proprement la loi de composition.

Définition 1.4.7. Soit E une courbe elliptique réelle vue comme une courbe projective plane de $\mathbb{P}^2(\mathbb{R})$, et soit P un point de E de coordonnées homogènes $[x_0, y_0, z_0]$. Notons F le polynôme homogène définissant l'équation de E, *i.e.* $E \equiv F(x, y, z) = 0$. La ***tangente à la courbe E passant par le point P*** est la droite $\mathscr{D}$ d'équation

$$\frac{dF}{dx}(x_0, y_0, z_0)\, x + \frac{dF}{dy}(x_0, y_0, z_0)\, y + \frac{dF}{dz}(x_0, y_0, z_0)\, z = 0.$$

Remarque 1.4.8. Notons que la définition précédente a bien un sens. En effet, étant donné qu'une courbe elliptique est lisse (par définition), il n'existe aucun point $P[x_0, y_0, z_0]$ tel que

$$\frac{dF}{dx}(x_0, y_0, z_0) = \frac{dF}{dy}(x_0, y_0, z_0) = \frac{dF}{dz}(x_0, y_0, z_0) = 0.$$

L'équation de la tangente en un point d'une courbe elliptique est donc toujours bien définie.

Nous avons à présent toutes les cartes en main pour définir la loi de composition sur l'ensemble des points d'une courbe elliptique réelle.

Définition 1.4.9. Soit E une courbe elliptique réelle et soient P, Q deux points de E. On définit la ***loi de composition***

$$+ : E \times E \to E : (P, Q) \mapsto P + Q$$

sur l'ensemble E selon les règles suivantes :

1. Si $Q = \mathscr{O}$, alors $P + Q = P$, *i.e.* le point à l'infini $\mathscr{O}$ fait office d'élément neutre pour l'opération $+$ définie sur E.
2. Si P et Q n'ont pas la même abscisse, alors la droite $\mathscr{D}$ passant par P et Q intersecte la courbe E en un troisième point R (vu le Corollaire 1.4.6). On pose alors $P+Q = -R$.
3. Si $Q = -P$, alors nécessairement $P + Q = P + (-P) = \mathscr{O}$ (vu le premier point). Par la suite, on notera directement $P - P = \mathscr{O}$.
4. Si $P = Q$, alors la droite $\mathscr{D}$ tangente à E en P intersecte la courbe en un troisième point R (vu le Corollaire 1.4.6, en notant le fait que P est alors un point d'intersection double). On pose alors $P + P = 2P = -R$.

Remarque 1.4.10. Les rôles des points 1 et 3 de la définition précédente auraient pu être inversés, et le point 1 aurait alors été vu comme une conséquence du point 3.

Pour y voir peut-être un peu plus clair, illustrons cette définition sur un exemple. Nous considérons ici la courbe elliptique d'équation $y^2 = x^3 - 2x + 2$.

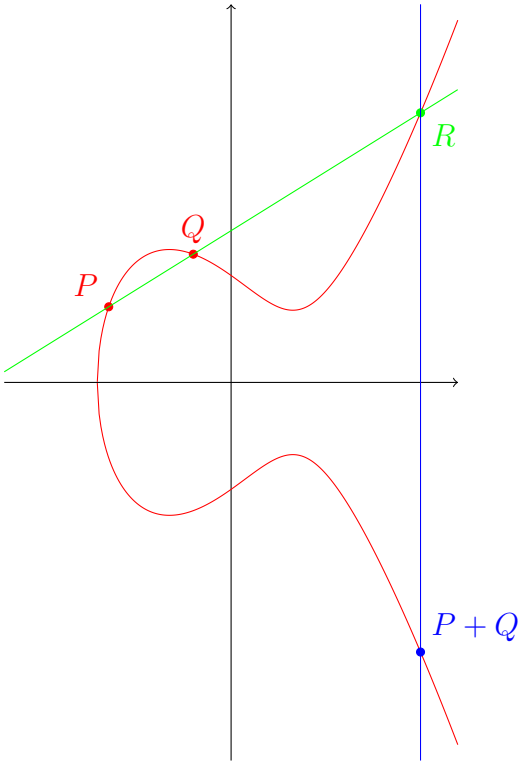

FIGURE 1.13 – Calcul de $P + Q$

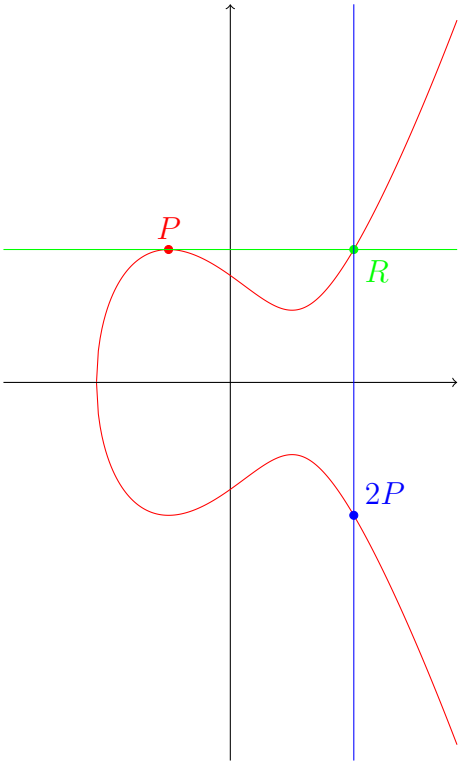

FIGURE 1.14 – Calcul de $2P$

Par la suite, nous utiliserons également la notation suivante : pour tout $m \in \mathbb{N}_0$, nous noterons assez logiquement

$$mP = \underbrace{P + P + \cdots + P}_{m \text{ fois}}.$$

Le théorème suivant, qui constitue en quelque sorte la finalité de cette section, découle assez naturellement de la définition précédente.

Théorème 1.4.11. *Soit E une courbe elliptique réelle. Le triplet $(E, +, \mathscr{O})$, où $+$ est la loi de composition de la Définition 1.4.9, forme un groupe commutatif.*

Démonstration. Tout a été mis en place dans la Définition 1.4.9 de sorte que les propriétés d'un groupe commutatif soient satisfaites. En effet, l'existence d'un neutre ainsi que de celle d'un opposé pour chaque élément sont assurées respectivement par les points 1 et 3 de cette définition. Quant à la commutativité, elle est évidente (la droite passant par P et Q est évidemment la même que celle passant par Q et P).

La preuve est donc essentiellement constituée de la démonstration de l'associativité. Celle-ci ne sera pas faite en détail ici, mais l'idée de la preuve est donnée. La démonstration se base sur un résultat dû au mathématicien français MICHEL CHASLES, mais qui est parfois connu sous le nom de "Théorème de Cayley-Bacharach" [16]. Celui-ci affirme que, étant données deux cubiques s'intersectant en exactement neuf point de $\mathbb{P}^2\left(\overline{K}\right)$ pour un champ K donné, si une troisième cubique passe par huit de ces points, alors elle passe nécessairement aussi par le neuvième.

16. Pour l'histoire, c'est d'abord Chasles qui a énoncé et démontré le théorème. Cayley et Bacharach ont quant à eux généralisé le résultat aux courbes de degré supérieur ou égal à 3. Le lecteur intéressé peut consulter [11] pour de plus amples informations.

Pour utiliser ce résultat, étant donnés trois points P, Q et R de la courbe elliptique, l'idée est donc de construire deux cubiques $\mathscr{C}_1$ et $\mathscr{C}_2$ passant toutes les deux par les huit même points de E, et telles que $\mathscr{C}_1$ passe par l'opposé de $P+(Q+R)$ (noté S) et $\mathscr{C}_2$ par l'opposé de $(P+Q)+R$ (noté T). Ainsi, les courbes E et $\mathscr{C}_1$ s'intersectent en neuf points. Or, par construction, la courbe $\mathscr{C}_2$ passe par huit de ces points. Le théorème de Chasles implique alors qu'elle passe par le 9$^{\text{e}}$, à savoir S. Or, $\mathscr{C}_2$ passe aussi par le point $T \in E$, et comme $\mathscr{C}_2$ et E s'intersectent en exactement neuf points (vu le Théorème 1.4.4, Bézout), alors nécessairement $S = T$. □

1.4.2 De façon générale

Nous allons le voir, cette structure de groupe formée par les points d'une courbe elliptique est particulièrement utile dans le domaine qui nous intéresse dans le cadre de ce mémoire, la cryptographie. Néanmoins, l'interprétation géométrique que nous en avons donné dans la section précédente n'est pas adaptée à tous les cas de figure (notamment lorsque nous manipulons des courbes dans un champ fini). Dès lors, il est intéressant de regarder ce qu'il se passe algébriquement, autrement dit, de décrire la loi de composition définie précédemment en termes de coordonnées de points. Nous allons donc repartir de la Définition 1.4.9 et voir ce qu'elle implique algébriquement pour, par la suite, utiliser ces résultats comme définition de la loi de composition sur les points d'une courbe elliptique définie sur un champ quelconque. Notons tout de même que nous continuerons, dans cette section, à parler de "droites" et de "courbes", même dans le cas d'un champ fini, bien que leurs représentations ne ressemblent pas à celles que nous avons l'habitude de voir.

Comme nous ne sommes plus dans le cas réel, il nous faut redéfinir l'opposé d'un point de la courbe de façon plus générale.

Définition 1.4.12. Soit E une courbe elliptique définie sur un champ K et P un point de E. On définit l'***opposé*** du point P, noté $-P$, comme étant

1. le point à l'infini $\mathscr{O}$ si $P = \mathscr{O}$,
2. le troisième point d'intersection entre E et la droite $\mathscr{D}$ passant par P et $\mathscr{O}$ sinon.

Notons que le sens de la définition précédente est assurée par le Lemme 1.4.1 et le Corollaire 1.4.6. En effet, le premier assure l'unicité de la droite $\mathscr{D}$ passant par les points P et $\mathscr{O}$, et le second l'existence et l'unicité du point $-P$.

Remarque 1.4.13. Si nous nous plaçons dans le cas réel, alors la Définition 1.4.12 est équivalente à la Définition 1.4.3 donnée dans la section précédente.

En effet, d'une part, le cas $P = \mathscr{O}$ est identique dans les deux définitions. D'autre part, si l'on considère une courbe elliptique réelle E, alors son équation dans l'espace affine $\mathbb{A}^2(\mathbb{R})$ est de la forme

$$y^2 = x^3 + px + q,$$

avec $p, q \in \mathbb{R}$ tels que $4p^3 + 27q^2 \neq 0$. Soit alors $P(x_P, y_P)$ un point de E. Pour voir E comme une courbe projective plane de $\mathbb{P}^2(\mathbb{R})$, il suffit d'homogénéiser l'équation ci-dessus, ce qui donne

$$y^2 z = x^3 + pxz^2 + qz^3,$$

et d'assimiler le point P au point projectif P' de coordonnées homogènes $[x_P, y_P, 1]$ (cf. la bijection Φ de la Proposition 1.2.23). Cherchons donc l'équation de la droite $\mathscr{D}$ passant par $\mathscr{O}$ (de coordonnées homogènes $[0, 1, 0]$) et P'.

La preuve du Lemme 1.4.1 nous fournit une méthode pour la déterminer. En effet, il suffit de trouver le noyau de l'application

$$T : \mathbb{R}^3 \to \mathbb{R}^2 : \begin{pmatrix} x \\ y \\ z \end{pmatrix} \mapsto \begin{pmatrix} x_P & y_P & 1 \\ 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix}.$$

Soit donc $(a, b, c) \neq (0, 0, 0)$ un élément de $\ker T$. On a

$$\begin{aligned} T(a, b, c) = 0 &\Leftrightarrow \begin{pmatrix} ax_P + by_P + c \\ b \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \\ &\Leftrightarrow \begin{cases} c = -ax_P \\ b = 0 \end{cases}. \end{aligned}$$

On en tire que l'équation de la droite $\mathscr{D}$ cherchée est donc

$$ax - ax_p z = 0 \Leftrightarrow x = x_P z,$$

en s'assurant [17] que $a \neq 0$. Il reste donc à vérifier que le troisième point d'intersection entre $\mathscr{D}$ et E est le point projectif $P'[x_P, -y_P, 1]$, à savoir le point projectif assimilé à $-P$. Cela montrera ainsi la concordance avec la Définition 1.4.3. Il faut donc chercher les solutions du système

$$\begin{cases} y^2 z = x^3 + pxz^2 + qz^3 \\ x = x_P z \end{cases}.$$

En injectant la seconde équation dans la première, il vient

$$y^2 z = x_P^3 z^3 + p x_P z^3 + q z^3.$$

Si $z = 0$, la solution correspondante est le point à l'infini $\mathscr{O}$. Supposons donc $z \neq 0$ et simplifions par z dans l'équation ci-dessus. On obtient

$$\begin{aligned} y^2 = x_P^3 z^2 + p x_P z^2 + q z^2 &\Leftrightarrow y^2 = \left(x_P^3 + p x_P + q\right) z^2 \\ &\Leftrightarrow y^2 = y_P^2 z^2 = (y_P z)^2 \\ &\Leftrightarrow y = \pm y_P z, \end{aligned}$$

où la seconde équivalence vient du fait que $P(x_P, y_P)$ est un point de E. In fine, on se retrouve donc avec les conditions

$$\begin{cases} y = \pm y_P z \\ x = x_P z \end{cases}.$$

Les triplets de réels satisfaisant ce système sont donc de la forme $(x_P z, y_P z, z)$ et $(x_P z, -y_P z, z)$, avec $z \in \mathbb{R}$. On en tire que la droite $\mathscr{D}$ intersecte E en deux autres points projectifs, de coordonnées homogènes respectives $[x_P, y_P, 1]$ et $[x_P, -y_P, 1]$, à savoir P' et $-P'$, ce qu'on voulait.

17. Si $a = 0$, alors on a également $b = c = 0$, qui est un cas à rejeter.

Cela étant, nous pouvons énoncer le résultat suivant, que nous utiliserons par la suite pour définir l'opération + dans un cadre plus général.

Théorème 1.4.14. *Soit E une courbe elliptique définie sur un champ K par une équation de Weierstrass de la forme*

$$y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6,$$

avec $a_1, a_2, a_3, a_4, a_6 \in K$. Soient également $P(x_P, y_P)$ et $Q(x_Q, y_Q)$ deux points de E.

1. *Si $Q = \mathscr{O}$, alors $P + Q = P$.*
2. *Les coordonnées du point $-P$,* i.e. *de l'opposé de P, sont*

$$x_{-P} = x_P \quad et \quad y_{-P} = -y_P - a_1x_P - a_3.$$

3. *Si $Q = -P$, alors $P + Q = \mathscr{O}$.*
4. *Si $Q \neq -P$, alors les coordonnées du point $P + Q$ sont données par*

$$\begin{cases} x_{P+Q} = \lambda^2 + a_1\lambda - a_2 - x_P - x_Q \\ y_{P+Q} = -\left(\lambda + a_1\right) x_{P+Q} - \nu - a_3 \end{cases},$$

avec

$$\lambda = \begin{cases} \dfrac{y_Q - y_P}{x_Q - x_P} & si\ P \neq Q \\ \dfrac{3x_P^2 + 2a_2x_P + a_4 - a_1y_P}{2y_P + a_1x_P + a_3} & si\ P = Q \end{cases} \quad et \quad \nu = y_P - \lambda x_P.$$

Démonstration.

1. C'est évident, étant donné que $\mathscr{O}$ est défini comme le neutre de l'opération $+$.
2. En passant à l'espace projectif $\mathbb{P}^2\left(\overline{K}\right)$ et en reprenant un développement analogue à celui de la Remarque 1.4.13, on sait que la droite $\mathscr{D}$ passant par P, $-P$ et $\mathscr{O}$ est définie par l'équation

$$x = x_Pz.$$

Comme ce n'est pas le point à l'infini qui nous intéresse ici, nous pouvons repasser aux équations affines par la bijection Φ de la Proposition 1.2.23 afin de simplifier les calculs. On est donc amené à résoudre le système

$$\begin{cases} y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6 \\ x = x_P \end{cases}.$$

La seconde équation nous indique déjà que $x_{-P} = x_P$. En l'injectant dans la première, on a également

$$\begin{aligned} & y^2 + a_1x_Py + a_3y = x_P^3 + a_2x_P^2 + a_4x_P + a_6 \\ \Leftrightarrow\ & y^2 + a_1x_Py + a_3y - x_P^3 - a_2x_P^2 - a_4x_P - a_6 = 0 \\ \Leftrightarrow\ & y^2 + \left(a_1x_P + a_3\right) y - \left(x_P^3 + a_2x_P^2 + a_4x_P + a_6\right) = 0, \end{aligned}$$

qui est donc une équation du second degré en la variable y.

Or, nous savons que y_P et y_{-P} sont solutions de cette équation, étant donné que les coordonnées de P et $-P$ satisfont le système de départ. En particulier, en utilisant la propriété de la somme et du produit des solutions d'une équation du second degré [18], il vient

$$y_P + y_{-P} = -\left(a_1 x_P + a_3\right) \Leftrightarrow y_{-P} = -y_P - a_1 x_P - a_3.$$

3. Cela découle à nouveau du fait que $\mathscr{O}$ est défini comme le neutre de l'opération $+$.
4. Comme P et Q ne sont pas opposés, l'équation de la droite affine $\mathscr{D}$ passant par ces deux points est de la forme $y = \lambda x + \nu$. Pour trouver sa troisième intersection avec E, notons

$$f(x, y) = y^2 + a_1 xy + a_3 y - x^3 - a_2 x^2 - a_4 x - a_6,$$

et substituons la variable y par son expression dans la droite $\mathscr{D}$. Cela donne

$$\begin{aligned} f(x, \lambda x + \nu) &= (\lambda x + \nu)^2 + a_1 x\left(\lambda x + \nu\right) + a_3\left(\lambda x + \nu\right) - x^3 - a_2 x^2 - a_4 x - a_6 \\ &= \lambda^2 x^2 + 2\lambda\nu x + \nu^2 + a_1\lambda x^2 + a_1\nu x + a_3\lambda x + a_3\nu - x^3 - a_2 x^2 - a_4 x - a_6 \\ &= -x^3 + \left(\lambda^2 + a_1\lambda - a_2\right) x^2 + \left(2\lambda\nu + a_1\nu + a_3\lambda - a_4\right) x + \left(\nu^2 + a_3\nu - a_6\right). \end{aligned}$$

Or, par définition, on sait que la droite $\mathscr{D}$ intersecte la courbe E en trois points : P, Q et un point R défini comme l'opposé du point $P + Q$ cherché. Dès lors, le polynôme du troisième degré en la variable x ci-dessus peut se factoriser sous la forme

$$\begin{aligned} f(x, \lambda x + \nu) &= c\left(x - x_P\right)\left(x - x_Q\right)\left(x - x_R\right) \\ &= cx^3 - c\left(x_P + x_Q + x_R\right) x^2 + c\left(x_P x_Q + x_P x_R + x_Q x_R\right) x - c x_P x_Q x_R. \end{aligned}$$

pour un certain $c \in K$. En identifiant les coefficients, on trouve alors

$$c = -1 \quad \text{et} \quad x_P + x_Q + x_R = \lambda^2 + a_1\lambda - a_2.$$

En particulier, $x_R = \lambda^2 + a_1\lambda - a_2 - x_P - x_Q$, et donc $y_R = \lambda x_R + \nu$.
En utilisant le point 2, on obtient finalement

$$\begin{aligned} x_{P+Q} &= x_R = \lambda^2 + a_1\lambda - a_2 - x_P - x_Q \\ y_{P+Q} &= -y_R - a_1 x_R - a_3 = -\lambda x_R - \nu - a_1 x_{P+Q} - a_3 \\ &= -\left(\lambda + a_1\right) x_{P+Q} - \nu - a_3, \end{aligned}$$

ce qui était annoncé.
Pour ce qui est de l'expression des paramètres λ et ν, nous allons considérer les deux cas suivants. Notons au préalable que l'expression de ν ne dépend pas du cas envisagé : comme le point P appartient à la droite, ses coordonnées satisfont son équation. Il vient donc

$$y_P = \lambda x_P + \nu \Leftrightarrow \nu = y_P - \lambda x_P.$$

Intéressons-nous à présent à λ.

- <u>Cas 1 :</u> $P \neq Q$

Dans ce cas, les points P et Q n'ont pas la même abscisse, et cela découle donc directement de l'expression de la pente d'une droite. Autrement dit,

$$\lambda = \frac{y_Q - y_P}{x_Q - x_P}.$$

18. Si on considère une équation du second degré de la forme $ax^2 + bx + c = 0$, avec $a \neq 0$, et que l'on note x_1 et x_2 ses solutions, alors

$$x_1 + x_2 = -\frac{b}{a} \quad \text{et} \quad x_1 \cdot x_2 = \frac{c}{a}.$$

- Cas 2 : $P = Q$

Ici, la droite à considérer est la tangente à E passant par P, dont l'équation est donnée par la Définition 1.4.7. Si on homogénéise l'équation de la courbe elliptique E, et que l'on considère alors le polynôme

$$F(x, y, z) = y^2z + a_1xyz + a_3yz^2 - x^3 - a_2x^2z - a_4xz^2 - a_6z^3,$$

ainsi que les coordonnées homogènes $[x_P, y_P, 1]$ du point P, il vient

$$\begin{aligned}
\frac{dF}{dx}(x_P, y_P, 1) &= a_1y_P - 3x_P^2 - 2a_2x_P - a_4, \\
\frac{dF}{dy}(x_P, y_P, 1) &= 2y_P + a_1x_P + a_3, \\
\frac{dF}{dz}(x_P, y_P, 1) &= y_P^2 + a_1x_Py_P + 2a_3y_P - a_2x_P^2 - 2a_4x_P - 3a_6.
\end{aligned}$$

L'équation de la tangente à la courbe E passant par le point P est donc

$$\begin{aligned}
\left(a_1y_P - 3x_P^2 - 2a_2x_P - a_4\right)x + (2y_P + a_1x_P + a_3)\,y \\
+ \left(y_P^2 + a_1x_Py_P + 2a_3y_P - a_2x_P^2 - 2a_4x_P - 3a_6\right)z = 0.
\end{aligned}$$

En considérant la partie affine de cette droite (à nouveau par l'intermédiaire de la bijection Φ de la Proposition 1.2.23), et en isolant la variable y, on obtient finalement [19]

$$y = -\frac{a_1y_P - 3x_P^2 - 2a_2x_P - a_4}{2y_P + a_1x_P + a_3}x - \frac{y_P^2 + a_1x_Py_P + 2a_3y_P - a_2x_P^2 - 2a_4x_P - 3a_6}{2y_P + a_1x_P + a_3}.$$

On peut alors identifier

$$\lambda = \frac{3x_P^2 + 2a_2x_P + a_4 - a_1y_P}{2y_P + a_1x_P + a_3} \quad \text{et} \quad \nu = \frac{a_2x_P^2 + 2a_4x_P + 3a_6 - y_P^2 - a_1x_Py_P - 2a_3y_P}{2y_P + a_1x_P + a_3},$$

et on vérifie facilement que l'expression de ν correspond également à celle mentionnée précédemment.

□

Exemple 1.4.15. Pour illustrer le théorème précédent, considérons la courbe elliptique E définie sur $\mathbb{F}_7$ par l'équation

$$y^2 = x^3 + 5x + 2.$$

On vérifie facilement que le point $P(4, 4)$ appartient à E. Calculons donc les deux premiers multiples de ce point, en commençant par $2P$. Vu le résultat précédent, on a

$$\lambda_{2P} = \frac{3x_P^2 + a_4}{2y_P} = \frac{3 \cdot 4^2 + 5}{2 \cdot 4} \equiv \frac{4}{1} \equiv 4 \bmod 7,$$

d'où

$$\begin{aligned}
x_{2P} &= \lambda_{2P}^2 - 2x_P = 4^2 - 2 \cdot 4 = 8 \equiv 1 \bmod 7, \\
y_{2P} &= -\lambda_{2P}x_{2P} - (y_P - \lambda x_P) = -4 \cdot 1 - (4 - 4 \cdot 4) = 8 \equiv 1 \bmod 7.
\end{aligned}$$

19. Notons que, comme $Q \neq -P$, le point 2 du théorème nous assure que $2y_P + a_1x_P + a_3 \neq 0$.

Ainsi, les coordonnées du point $2P$ sont $(1,1)$. Pour trouver les coordonnées de $3P$, remarquons que $3P = P + 2P$. Dès lors,

$$\lambda_{3P} = \frac{y_P - y_{2P}}{x_P - x_{2P}} = \frac{4-1}{4-1} \equiv 1 \bmod 7,$$

ce qui implique

$$\begin{aligned} x_{3P} &= \lambda_{3P}^2 - x_P - x_{2P} = 1^2 - 4 - 1 = -4 \equiv 3 \bmod 7, \\ y_{3P} &= -\lambda_{3P} x_{3P} - (y_P - \lambda x_P) = -1 \cdot 3 - (4 - 1 \cdot 4) = -3 \equiv 4 \bmod 7. \end{aligned}$$

Les coordonnées du point $3P$ sont donc $(3,4)$.

On peut dès lors étendre le Théorème 1.4.11 au cas d'un champ K quelconque. La preuve du résultat qui suit peut se baser sur les formules données dans le Théorème 1.4.14, mais ne sera pas réalisée ici. La partie la plus compliquée étant l'associativité, un logiciel de calcul peut par exemple être utilisé afin de la démontrer.

Théorème 1.4.16. *Soit E une courbe elliptique définie sur un champ K quelconque. Le triplet $(E, +, \mathscr{O})$, où $+$ est la loi de composition du Théorème 1.4.14, forme un groupe commutatif.*

Remarque 1.4.17. Étant donnée une courbe elliptique E définie sur un champ K, nous désignerons par $E(K)$ le groupe formé par l'ensemble de ses points si nous souhaitons insister sur la structure de groupe, et non uniquement sur l'ensemble des points de la courbe.

1.4.3 Parenthèse sur les champs finis

Nous terminons ce chapitre par une courte section concernant les courbes elliptiques définies sur des champs finis, car ce sont celles-ci qui nous intéresseront particulièrement par la suite.

Le résultat suivant découle directement du dernier théorème de la section précédente, mais vaut tout de même la peine d'être énoncé.

Corollaire 1.4.18. *Soit E une courbe elliptique définie sur un champ $\mathbb{F}_q$. Le triplet $(E, +, \mathscr{O})$, où $+$ est la loi de composition du Théorème 1.4.14, forme un groupe commutatif fini.*

Démonstration. Cela découle directement du Théorème 1.4.16 et du fait qu'une courbe elliptique définie sur $\mathbb{F}_q$ est constituée d'au plus q^2+1 points (à savoir tous les couples $(x,y) \in \mathbb{F}_q \times \mathbb{F}_q$ possibles et le point à l'infini). □

En réalité, il y a moyen d'être beaucoup plus précis concernant l'ordre du groupe formé par les points d'une courbe elliptique définie sur $\mathbb{F}_q$. En effet, la preuve du théorème précédent nous apprend simplement que son ordre est borné par $q^2 + 1$, mais cette borne n'est pas très intéressante...

Le théorème suivant, dû au mathématicien allemand HELMUT HASSE $(1898 - 1979)$, nous fournit un meilleur intervalle pour l'ordre du groupe.

Théorème 1.4.19 (Hasse, 1936)**.** *Si E est une courbe elliptique définie sur $\mathbb{F}_q$, alors*

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

où $|E(\mathbb{F}_q)|$ désigne le cardinal de l'ensemble des points de E (et donc l'ordre du groupe $E(\mathbb{F}_q)$).

En particulier, l'inégalité précédente peut se mettre sous la forme

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

ce qui permet de mettre en évidence le fait qu'asymptotiquement, les ordres des groupes $E(\mathbb{F}_q)$ et $\mathbb{F}_q^*$ (cf. Définition 1.1.10 et Proposition 1.1.11) sont les mêmes. Dans la pratique, c'est d'ailleurs cette double inégalité qui sera utilisée. De plus, le résultat nous permet parfois de déterminer exactement l'ordre du groupe, comme dans les exemples suivants.

Exemples 1.4.20. Reprenons l'Exemple 1.4.15 et considérons la courbe E_1 d'équation

$$y^2 = x^3 + 5x + 2$$

définie sur $\mathbb{F}_7$. On a $2\sqrt{7} \simeq 5,3$ et, comme l'ordre de $E_1(\mathbb{F}_7)$ est un nombre naturel, on peut considérer $\lfloor 2\sqrt{7} \rfloor = 5$. Ainsi, par le théorème de Hasse, il vient

$$7 + 1 - 5 \leq |E_1(\mathbb{F}_7)| \leq 7 + 1 + 5 \Leftrightarrow 3 \leq |E_1(\mathbb{F}_7)| \leq 13.$$

On peut alors montrer que l'ordre du point $P_1(4, 4)$ vaut 9. Vu la Proposition 1.1.8, $|E_1(\mathbb{F}_7)|$ doit être un multiple de 9, ce qui implique $|E_1(\mathbb{F}_7)| = 9$ vu les bornes ci-dessus.

Considérons à présent un champ plus conséquent, à savoir $\mathbb{F}_{101}$, et la courbe elliptique E_2 définie par l'équation

$$y^2 = x^3 + 7x + 1.$$

Par le théorème de Hasse, comme $\lfloor 2\sqrt{101} \rfloor = 20$, on a

$$101 + 1 - 20 \leq |E_2(\mathbb{F}_{101})| \leq 101 + 1 + 20 \Leftrightarrow 82 \leq |E_2(\mathbb{F}_{101})| \leq 122.$$

Or, on vérifie aisément que le point $P_2(0, 1)$ est un point de la courbe et on peut montrer que son ordre est 116 (par exemple via un logiciel de calcul). Par la même justification que l'exemple précédent, il vient alors $|E_2(\mathbb{F}_{101})| = 116$.

Enfin, nous terminons cette section par un dernier résultat, qui donne une idée de la forme du groupe associé à une courbe elliptique. A nouveau, nous ne le démontrerons pas, car la preuve implique l'introduction de notions et de résultats intermédiaires qui sortent de l'objectif fixé pour ce travail. Le lecteur intéressé peut, par exemple, consulter la référence [26] pour plus d'informations.

Théorème 1.4.21. *Soit E une courbe elliptique définie sur le champ $\mathbb{F}_q$. Le groupe $E(\mathbb{F}_q)$ des points de la courbe est soit cyclique, soit isomorphe au produit de deux groupes cycliques. Dans le premier cas,*

$$E(\mathbb{F}_q) \cong \mathbb{Z}_d, \quad \text{où } d = |E(\mathbb{F}_q)|.$$

Dans le second, il existe un unique couple de naturels (n, m) tel que

$$E(\mathbb{F}_q) \cong \mathbb{Z}_n \times \mathbb{Z}_m,$$

n divise m et n divise $q - 1$.

Exemples 1.4.22. A nouveau, reprenons la courbe elliptique E_1 des Exemples 1.4.15 et 1.4.20, définie sur $\mathbb{F}_7$ par l'équation

$$y^2 = x^3 + 5x + 2.$$

Dans l'exemple précédent, nous avions déterminé que $|E_1(\mathbb{F}_7)| = 9$, et que $P_1(4, 4)$ est un point d'ordre 9 de la courbe. Ceci nous permet alors d'affirmer que $E_1(\mathbb{F}_7)$ est un groupe cyclique d'ordre 9, *i.e.* $E_1(\mathbb{F}_7) \cong \mathbb{Z}_9$.

A présent, considérons la courbe E_2, toujours définie sur le champ $\mathbb{F}_7$, mais cette fois-ci caractérisée par l'équation

$$y^2 = x^3 + 3x.$$

Une rapide vérification permet de montrer que

$$E_2(\mathbb{F}_7) = \{\mathscr{O}, (0,0), (1,2), (1,5), (2,0), (3,1), (3,6), (5,0)\},$$

et donc $|E_2(\mathbb{F}_7)| = 8$. Grâce au théorème précédent, nous pouvons éliminer une bonne partie des groupes d'ordre 8 possibles, et se focaliser sur les deux possibilités restantes :

$$E_2(\mathbb{F}_7) \cong \mathbb{Z}_8 \text{ ou } \mathbb{Z}_4 \times \mathbb{Z}_2\,.$$

On peut alors vérifier qu'aucun point du groupe n'est d'ordre 8, ce qui implique finalement $E_2(\mathbb{F}_7) \cong \mathbb{Z}_4 \times \mathbb{Z}_2$.

Remarque 1.4.23. Par rapport à ce résultat, Koblitz précise dans son article (que l'on peut retrouver en [23]) que, si E_K n'est pas cyclique, alors il est isomorphe à un produit de deux groupes cycliques (vu le théorème précédent) dans lequel un des facteurs cycliques est beaucoup plus petit que l'autre (*i.e.* $n \ll m$ par exemple [20], pour reprendre les notations du théorème), ce qui lui permet d'affirmer que dans ce cas, E_K est tout de même un groupe *"presque"* cyclique.

20. Il est difficile de quantifier cette différence d'ordre de grandeur, étant donné qu'elle dépend fortement de l'ordre du champ considéré.

Chapitre 2

La cryptographie basée sur les courbes elliptiques

Nous entrons bientôt dans le vif du sujet ! Ce court chapitre a pour but de faire le lien entre le précédent et celui qui va suivre. En particulier, après quelques rappels historiques et théoriques, nous nous intéresserons à l'utilité des courbes elliptiques dans le domaine de la cryptographie. En outre, nous nous arrêterons sur un exemple concret de leur utilisation dans le cadre d'un cryptosystème connu.

2.1 La cryptographie, un domaine de recherche qui date !

Avant de nous lancer à proprement parler dans le monde de la cryptographie, il semble intéressant de donner ici quelques éléments historiques et biographiques clés de ce domaine mathématique qui, contrairement à ce que l'on pourrait penser, ne date pas d'hier.

2.1.1 Quelques grandes dates

La cryptographie [1] est un domaine d'étude qui fait en réalité partie d'un plus grand domaine : la ***cryptologie***. La cryptologie regroupe à la fois la ***cryptographie***, qui concerne les techniques de chiffrement d'un message, et la ***cryptanalyse***, qui analyse et étudie le déchiffrement du message chiffré.

D'aucuns pourraient croire que la cryptologie est relativement récente, et qu'elle est affaire d'espions ou d'agences secrètes en tout genre. Mais en réalité, c'est un domaine d'études qui existe depuis de nombreux siècles, et qui est loin de se limiter à cette utilisation ! En fait, à l'heure actuelle, la cryptographie est même omniprésente autour de nous : de nos transactions bancaires à nos messages privés sur Facebook, tout est crypté d'une manière ou d'une autre pour sécuriser nos informations (du moins, du plus grand nombre). Pour reprendre la métaphore utilisée dans [46], il faut imaginer que, lorsque nous effectuons ce genre d'opération, nous sommes comme des émetteurs radio qui envoient des signaux que n'importe qui peut capter mais qui, sans le bon récepteur, ne peuvent pas être déchiffrés. Nous reprendrons d'ailleurs également un passage tiré de cet ouvrage, qui résume bien l'importance d'un tel domaine de recherche :

"Nous avons tendance à penser que la cryptographie est affaire d'espions et de communications militaires, et bien sûr cela en fait partie, mais son utilisation est devenue aussi omniprésente que le réseau WiFi. Elle affecte quotidiennement nos vies et continuera à être un élément essentiel du monde moderne, donc cela semble être un sujet qui vaut la peine d'être compris."

1. Les informations présentées dans cette section sont tirées de [4], [9] et [52].

Le plus vieux document chiffré connu date de l'Antiquité, et plus précisément du 16e siècle avant Jésus-Christ. En fait, on peut retrouver des traces de cryptographie très rapidement après la genèse de l'écriture dans les différentes civilisations. Certains disent d'ailleurs que l'écriture elle-même était une façon de chiffrer un message, étant donné qu'à ces époques, une très faible partie de la population était alphabétisée. Il faut cependant attendre quelques siècles de plus pour trouver les premiers "vrais" cryptosystèmes, tels que les chiffrements par substitution (comme le *Code de César*, qui consistait à remplacer chaque lettre du message par une autre lettre de l'alphabet ; on parle alors de ***substitution monoalphabétique***).

Comme beaucoup d'autres domaines, c'est à la Renaissance que la cryptographie a commencé à être davantage étudiée et apprise. Les motivations pour chiffrer un message étaient nombreuses, allant des enjeux politiques et diplomatiques (notamment les guerres) à la protection de lettres d'amour. Plusieurs techniques de chiffrement sont donc développées, dont certaines seront utilisées encore longtemps après leur création. Citons ici l'exemple du ***nomenclateur***, créé en 1379 par Gabriele de LAVINDE, secrétaire du Pape de l'époque et cryptologue italien, qui est un recueil de codes et de clés.

La cryptographie a connu un second changement majeur au 19e siècle, avec l'apparition de deux nouvelles technologies, à savoir le télégraphe, puis la radio. En effet, les communications par ces deux outils n'étant a priori pas sécurisées, il a fallu développer de nouvelles techniques de chiffrement de messages confidentiels, ceci étant d'autant plus urgent que l'idée qu'une guerre éclate en Europe est de plus en plus probable. Les deux guerres mondiales du 20e siècle vont d'ailleurs donner un grand coup d'accélérateur aux techniques de cryptographie, mais également et surtout à celles utilisées en cryptanalyse. Citons ici l'exemple très connu de la machine *Enigma* utilisée par les Allemands pour communiquer durant la Seconde Guerre Mondiale, mais dont l'invention remonte à une vingtaine d'années avant le début du conflit. La méthode de chiffrement utilisée par *Enigma* était à la fois simple et astucieuse, puisqu'elle consistait à substituer une lettre par une autre, la subsitution changeant d'une lettre à l'autre. Le déchiffrement des messages chiffrés par la machine, auquel a participé notamment le célèbre Alan TURING (1912 – 1954), constituera une aide précieuse pour les offensives des forces anglaises.

Figure 2.1 – La machine *Enigma*[2]

Fin du 20e siècle, l'utilisation et le développement des ordinateurs ont permis l'invention de techniques de chiffrement et de déchiffrement de plus en plus poussées. L'objectif visé étant de fournir des cryptosystèmes davantage sécurisés mais, dans le même temps, nécessitant le moins de stockage, d'énergie et de temps possible, il y a fort à parier que la cryptologie est et restera un domaine d'étude inépuisable en termes de nouvelles inventions, et dont les découvertes continueront de nous surprendre d'année en année.

2. Photo issue de <https://fr.wikipedia.org/wiki/Enigma_(machine)>.

2.1.2 Victor S. Miller et Neal Koblitz, pionniers de la cryptographie des courbes elliptiques

Bien que les courbes elliptiques soient un sujet plus ancien des mathématiques, leur utilisation dans le cadre de la cryptographie est beaucoup plus récente. En effet, c'est respectivement en 1986 et en 1987 que Victor S. MILLER (1947 – ...) et Neal KOBLITZ (1948 – ...) ont mis en évidence les bénéfices de l'utilisation de telles courbes dans le cadre de cryptosystèmes existants, et ce indépendamment l'un de l'autre. Étant donné leur importance dans le cadre de ce travail, il semble relativement opportun d'explorer leurs biographies respectives.

Victor S. Miller (1947 – ...)

Victor Saul MILLER est un mathématicien américain né en 1947 à Brooklyn, New-York. Même si nous disposons d'assez peu d'informations le concernant [3], Miller a passé la plus grande partie de sa vie à étudier les mathématiques, et ce dans plusieurs institutions mondialement reconnues. Il obtient notamment une licence en mathématiques à l'Université de Columbia en 1968 ainsi, en 1975, qu'un doctorat en mathématiques à l'Université de Harvard. Il passera également par le département de mathématiques de l'Université du Massachusetts, à Boston, ainsi que par celui du Centre de recherche Thomas J. Watson à New York. Parmi les sujets qu'il étudie, on trouve notamment la théorie algorithmique des nombres, la combinatoire, la compression de données et évidemment la cryptographie.

Miller ne se contente néanmoins pas de mathématiques pures. En 1978, il rejoint le projet IBM 801, qui est un ordinateur expérimenta, à nouveau au Centre de recherche Thomas J. Watson, mais dans le département d'informatique cette fois. Depuis 1993, Victor S. Miller travaille au *Centre de recherche sur les communications (CCR)* de l'Institut des analyses de défense à Princeton, New Jersey. A l'heure actuelle, il est néanmoins très probable qu'il soit en fin de carrière, si pas retraité. Enfin, notons que Miller a contribué à la création de plusieurs algorithmes en cryptographie et en compression de données.

C'est en 1986 que l'article de Miller détaillant ses découvertes concernant l'utilisation des courbes elliptiques en cryptographie, logiquement intitulé *"Use of Elliptic Curves in Cryptography"* (voir [31]), est publié. Il a reçu plusieurs prix pour ses travaux (dont la découverte de cryptosystèmes basés sur les courbes elliptiques, mais pas uniquement), tels que le *RSA Conference Award for Excellence in Mathematics* ou encore le *Prix Levchin*, parmi d'autres récompenses liées au domaine de la cryptographie. En particulier, en 2013, Miller est reconnu par l'*International Association for Cryptologic Research (IACR)* pour ses multiples travaux en cryptographie.

Figure 2.2 – Victor Saul MILLER

3. Les informations présentes dans cette section sont tirées de [54]. Photo issue de <https://www.iacr.org/fellows/2013/miller.html>.

Neal Koblitz (1948 – ...)

Neal KOBLITZ[4] est également un mathématicien américain, né en 1948. Jeune, il s'est rapidement trouvé une passion pour les mathématiques, et ce dès l'école primaire. Cet engouement le mène tout droit à l'Université d'Harvard, dont il sort diplômé en 1969. En 1974, il obtient son doctorat à l'Université de Princeton. Il retourne à l'Université d'Harvard de 1975 à 1979, mais cette fois-ci en tant qu'instructeur, et c'est finalement en 1979 qu'il commence à travailler à l'Université de Washington comme professeur. C'est dans les années 80 que Koblitz commence à s'intéresser plus profondément à la cryptographie et, en 1987, son article intitulé *"Elliptic Curves Cryptosystems"* (voir [23]), est finalement publié. Bien qu'il arrive à des résultats similaires à ceux obtenus par Miller, ses découvertes sont indépendantes de ce dernier. Tout comme son pair, Koblitz sera récompensé du *Prix Levchin* en 2021 pour ses travaux.

Neal Koblitz a beaucoup voyagé tout au long de sa vie. Il a notamment séjourné en Inde dans sa petite enfance. De plus, comme bon nombre d'autres chercheurs, ses recherches l'ont poussé à voyager aux quatre coins du globe, notamment avec sa femme, Ann Hibner KOBLITZ. Il se rend notamment en Union Soviétique (Moscou), au Nicaragua, à Cuba ou encore au Pérou. Mais c'est particulièrement le Vietnam qui jouera un rôle important dans la vie de Koblitz, puisque ce dernier y séjournera pendant plusieurs années avec son épouse. Il aura notamment à cœur d'améliorer les conditions d'éducation et de recherche scientifique dans le pays.

Koblitz se veut également engagé. Durant ses études, il s'intéresse et s'investit dans des causes concernant des sujets tels que la guerre du Vietnam (dans la foulée des protestations des étudiants de l'Université de Columbia au printemps 1968), ou encore les tensions raciales existant entre les communautés blanche et noire aux États-Unis. Il devient d'ailleurs, à la fin des années 60, un membre actif du mouvement étudiant *Students for a Democratic Society* (SDS), ce qui le poussera à s'engager dans l'armée, avant d'être renvoyé à la vie civile en décembre 1970 et de poursuivre son parcours académique.

En 1985, il crée avec sa femme le *Prix Kovalevskaia*, qui récompense les femmes scientifiques dans les pays en voie de développement. Ce dernier est financé grâce aux revenus des droits d'auteur que perçoit son épouse suite à la publication de la biographie, en 1983, de la mathématicienne russe Sofia Kovalevskaia (1850 – 1891).

Figure 2.3 – Neal KOBLITZ

4. Les informations de cette section sont issues de [53] ainsi que de l'autobiographie de Koblitz [25]. Photo issue de <https://math.washington.edu/people/neal-i-koblitz>.

2.2 Quelques notions de cryptographie

Avant d'aller plus loin, il semble intéressant de s'arrêter un instant sur quelques notions liées à la cryptographie. Le but ici est de donner une idée globale de ce qu'est un cryptosystème et des différentes techniques utilisées dans ce domaine.

2.2.1 Généralités

La ***cryptographie*** peut être définie comme la science des communications secrètes. En effet, le but de ce domaine mathématique est d'établir des méthodes permettant l'échange sécurisé d'informations entre deux parties (deux personnes, un serveur d'un site de vente en ligne et un client, etc.). Celles-ci sont généralement dénommées *Alice* et *Bob*[5]. Cette communication se fait via un canal et doit être telle qu'une personne tierce (typiquement appelée *Oscar* ou *Eve* en fonction de son action[6]) ne puisse pas comprendre le message transmis.

Supposons donc qu'Alice veuille transmettre un message m à Bob. Ce message, qui peut a priori être ce que l'on veut (un texte usuel écrit dans une certaine langue, une série de chiffres, etc.), est appelé ***texte*** ou ***message clair***. Pour que n'importe qui ne puisse pas en connaître la teneur, elle transforme ce texte en un message m', appelé ***texte*** ou ***message secret*** ou ***chiffré***, par des procédés de ***chiffrement*** qui lui sont propres que l'on appelle ***clés***. Elle envoie alors ce résultat à Bob qui, lui aussi, est en possession d'une clé, mais de ***déchiffrement*** cette fois, et qui lui permettra, à lui et à lui seul, de déchiffrer le message m' pour retrouver le texte m de départ. A l'inverse, toute personne qui arriverait à intercepter le message chiffré m' ne saurait qu'en faire, étant donné qu'il ne disposerait pas de la clé permettant de le déchiffrer.

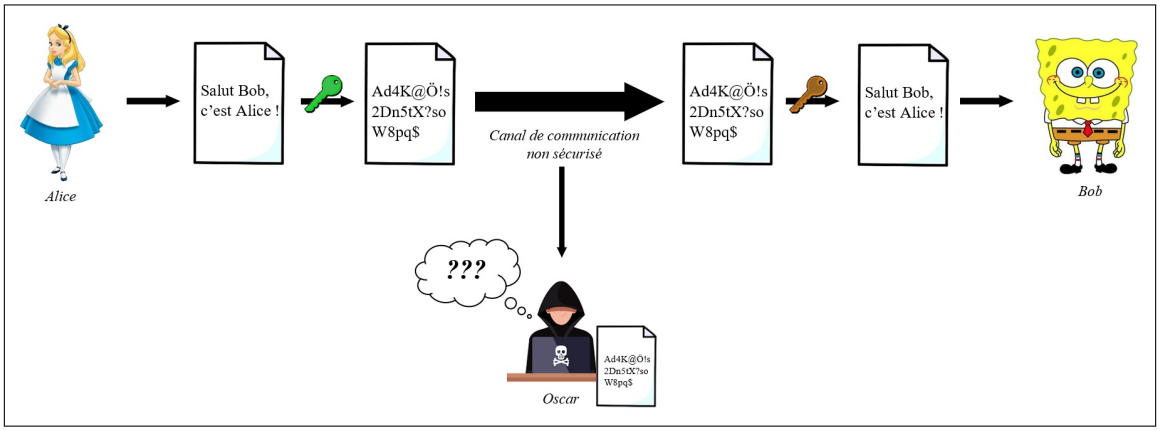

FIGURE 2.4 – Illustration d'une communication entre Alice et Bob[7]

Bien sûr, une telle procédure doit satisfaire plusieurs conditions qui semblent, somme toutes, assez logiques. Si on note m un message clair à transmettre, e la procédure de chiffrement et d celle de déchiffrement, alors

- pour tout message clair m, on devrait avoir $d(e(m)) = m$, *i.e.* chiffrer et déchiffrer le message m successivement permet de retrouver le message clair initial,
- les messages $m' = e(m)$ et $m = d(m')$ devraient être relativement rapides à calculer, sans pour autant que m puisse être facilement trouvé à partir de m'.

5. Pour l'anecdote, ces deux prénoms ont été introduits dans l'article [44] présentant le RSA (cryptosystème dont nous parlerons un peu plus loin) pour éviter l'emploi des termes *"personne A"* et *"personne B"*.

6. On utilise le nom d'*Oscar* pour désigner un opposant actif, qui peut intervenir dans la communication (*opponent* en anglais) et *Eve* pour un opposant passif, qui ne fait qu'écouter (*eavesdropper* en anglais).

7. Création personnelle, inspirée de l'illustration issue de `<http://studybyyourself.com/seminar/cryptography/course/chapter-3-asymmetric-cryptography/?lang=fr>`.

On peut alors donner la définition suivante.

Définition 2.2.1. Un ***système cryptographique*** ou ***cryptosystème*** est un triplet $(\mathscr{P}, \mathscr{C}, \mathscr{K})$ où

- $\mathscr{P}$ est l'ensemble fini des textes clairs possibles (*plaintexts* en anglais),
- $\mathscr{C}$ est l'ensemble fini des textes chiffrés possibles (*ciphertexts* en anglais),
- $\mathscr{K}$ est l'ensemble fini des clés possibles (*keys* en anglais), parfois appelé ***espaces des clés***,

et tel que, pour tout $k \in \mathscr{K}$, il existe une ***fonction de chiffrement*** (*encryption rule* en anglais)

$$e_k : \mathscr{P} \to \mathscr{C} : m \mapsto e_k(m)$$

et une ***fonction de déchiffrement*** (*decryption rule* en anglais)

$$d_k : \mathscr{C} \to \mathscr{P} : m \mapsto d_k(m)$$

telles que $d_k(e_k(m)) = m$ pour tout $m \in \mathscr{P}$.

Remarque 2.2.2. Notons que la fonction de chiffrement doit nécessairement être injective, sinon le déchiffrement du message poserait problème.

Avant de passer au chiffrement d'un message et à son envoi, il se peut également qu'on doive réaliser une phase préalable : le ***codage***.

Définition 2.2.3. Le ***codage*** est une action réalisée afin de transformer un texte clair en un texte équivalent qui soit plus simple à traiter d'un point de vue cryptographique (typiquement, pour avoir des fonctions de chiffrement plus simples à manipuler).

Exemple 2.2.4. Si on veut transmettre un texte écrit en français mais que l'on souhaite plutôt travailler avec des éléments d'un groupe ou d'un anneau (des nombres par exemple), et non pas avec des lettres, on peut par exemple définir une bijection qui, à chaque lettre de l'alphabet, associe sa position dans celui-ci et qui envoie le symbole "espace" sur un nombre également. Concrètement, il s'agirait d'une application

$$\alpha : \{\mathrm{a}, \mathrm{b}, \ldots, \mathrm{z}, \text{espace}\} \to \mathbb{Z}_{27} : \begin{cases} \mathrm{a} & \mapsto & 0, \\ \mathrm{b} & \mapsto & 1, \\ & \vdots & \\ \mathrm{z} & \mapsto & 25, \\ \text{espace} & \mapsto & 26. \end{cases}$$

Remarque 2.2.5. On peut rapprocher la notion de codage définie ci-dessus à celle utilisée communément en combinatoire des mots. En effet, pour rappel, un ***codage*** en combinatoire des mots est un morphisme 1-uniforme, *i.e.* une application $\sigma : A^* \to B^*$ (où A et B sont des alphabets) telle que [8]

$$\sigma(uv) = \sigma(u)\sigma(v),\ \forall u, v \in A^* \quad \text{et} \quad |\sigma(a)| = 1,\ \forall a \in A.$$

Autrement dit, chaque lettre de l'alphabet de départ est remplacée par une unique lettre de l'alphabet d'arrivée. On peut donc voir la correspondance entre la Définition 2.2.3 et celle donnée ci-dessus : il s'agit d'associer à chaque symbole de l'alphabet initial un symbole plus facilement manipulable par la suite (cf. Exemple 2.2.4).

8. Les notations utilisées ici sont les notations habituelles employées en combinatoire des mots.

Pour clôturer cette section, nous insisterons encore sur les deux grands "types" de cryptographie que l'on peut rencontrer : la ***cryptographie symétrique***, aussi appelée ***cryptographie à clé secrète*** ou ***privée***, et la ***cryptographie asymétrique***, quant à elle davantage appelée ***cryptographie à clé publique*** (et abrégé PKCS pour *public-key cryptosystem*).

La cryptographie symétrique est la plus ancienne des deux [9]. Le principe est simple : le chiffrement et le déchiffrement du message clair à transmettre se fait au moyen de la même clé, d'où l'adjectif utilisé pour nommer cette méthode. Dans ce cas, la clé doit donc impérativement rester secrète, sans quoi n'importe quel individu l'ayant en sa possession sera en mesure de déchiffrer les messages chiffrés s'il les intercepte. Un exemple de cryptosystème à clé secrète est le Code de César introduit dans la Section 2.1.1. Malheureusement, ce type de cryptographie présente deux inconvénients majeurs :

- puisque la clé ne doit être connue que par les deux parties souhaitant communiquer, il est nécessaire de la transmettre via un canal sûr et sécurisé, ce qui n'est pas une mince affaire. En effet, il faudrait déjà disposer d'un moyen de communiquer secrètement (pour échanger les clés), alors que c'est exactement ce que l'on souhaiterait faire ! C'est donc en quelque sorte le serpent qui se mord la queue, ce qui complique l'initiation du processus de communication.
- une clé doit être générée pour chaque couple de parties souhaitant communiquer. Imaginons que, pour une quelconque raison, l'Université de Liège souhaite mettre en place un système de communication secrète entre chacun de ses 27 000 étudiants. Si elle veut, pour ce faire, utiliser un cryptosystème à clé secrète, elle devra donc générer pas moins de $C^2_{27000} = \frac{27000 \cdot 26999}{2} = 364\ 486\ 500$ clés différentes !

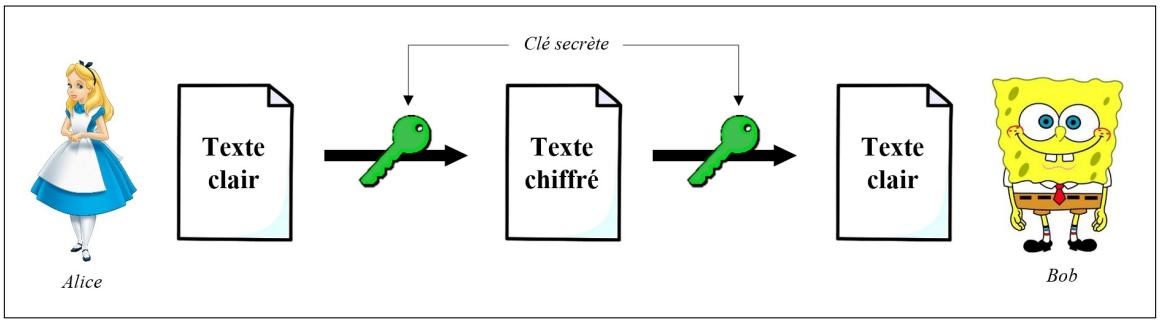

FIGURE 2.5 – Cryptosystème à clé secrète ou symétrique [10]

Bien qu'ils paraissent peu avantageux et peu sécurisés, plusieurs cryptosystèmes à clé secrète sont considérés comme sûrs [11] et sont toujours utilisés à l'heure actuelle, notamment car ils sont plus rapides que certains cryptosystèmes à clé publique. Parmi eux, citons notamment l'*Advanced Encryption Standard (AES)*, mis au point par deux cryptologues belges, JOAN DAEMEN (1965 – ...) et VINCENT RIJMEN (1970 – ...), dans le but remplacer le *Data Encryption Standard (DES)* qui, lui, présentait de plus en plus de failles de sécurité à cause de la puissance de calcul grandissante des machines à l'aube du 21^{e} siècle. Le lecteur intéressé peut notamment consulter [7] pour en apprendre plus sur le fonctionnement de l'AES.

9. On retrouve des traces de ce type de chiffrement datant de 2 000 av. J.-C.
10. Création personnelle.
11. Un des principaux rôles des cryptologues est de continuellement tester les cryptosystèmes utilisés afin de vérifier leur vulnérabilité face à différentes attaques. Bien sûr, un système cryptographique qui était sûr hier peut tout à fait ne plus l'être aujourd'hui, en raison des progrès continus réalisés en termes de puissance de calcul des ordinateurs.

La cryptographie asymétrique est, quant à elle, beaucoup plus récente. Contrairement à la cryptographie symétrique, elle ne repose pas sur une unique clé tenue secrète par les deux parties communicantes. Ici, on distingue deux clés différentes : la ***clé publique*** qui permet de chiffrer le message clair et qui, comme son nom l'indique, peut être divulguée sans problème, et la ***clé secrète*** qui sert à déchiffrer le message chiffré reçu et qui, elle, n'est connue que du récepteur du message (Bob dans la Figure 2.6). Le gros avantage de ce type de cryptosystème est qu'il ne nécessite aucun échange de clé audacieux. Il est néanmoins nécessaire de s'assurer que, dans ce cas, révéler la clé de chiffrement ne permet pas de déterminer facilement la fonction de déchiffrement correspondante, auquel cas un tel cryptosystème perdrait tout son intérêt.

Dans la pratique, si Alice désire transmettre un message à Bob, ce dernier produit un couple de fonctions de chiffrement/déchiffrement (e_k, d_k). Il transmet alors la fonction e_k à Alice (comme elle peut être rendue publique, il n'y a pas de risque lors de la transmission), tandis qu'il conserve précieusement et secrètement sa fonction de déchiffrement d_k. Alice peut alors transmettre son message m à Bob en lui envoyant le texte chiffré $e_k(m)$, que Bob déchiffrera aisément grâce à d_k. Quant à Oscar ou Eve, si même ils arrivent à intercepter le message chiffré, ils ne pourront rien en faire étant donné que la clé de déchiffrement a été gardée secrète et que, par hypothèse, la fonction e_k (connue de tous) ne permet pas de déterminer (du moins, pas facilement) la fonction de déchiffrement correspondante.

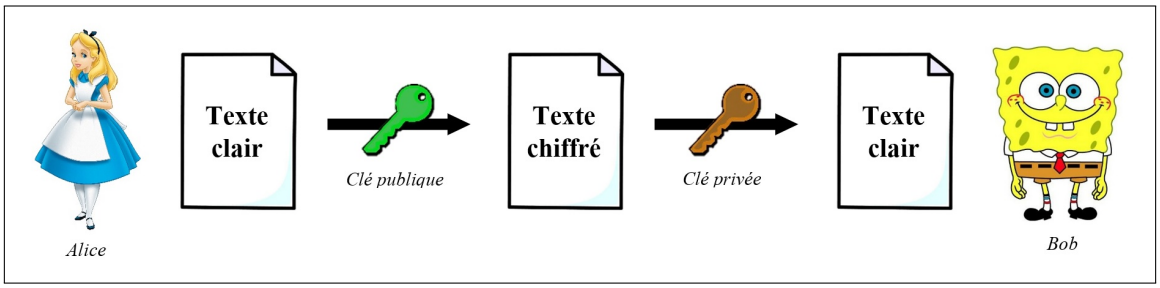

FIGURE 2.6 – Cryptosystème à clé publique ou asymétrique [12]

Pour qu'un tel système soit sécurisé et difficile à "pirater" pour un acteur extérieur à la communication, il faut donc trouver des clés impliquant une tâche qui, pour peu que l'on dispose de la bonne information, est facilement réalisable, et très compliquée à exécuter sinon. Cette information supplémentaire est appelée *trap door* en anglais.

2.2.2 Un exemple de cryptosystème à clé publique : le RSA

Le système cryptographique que nous présentons ci-dessous, fort connu en cryptographie, illustre assez bien les caractéristiques désirées d'un cryptosystème à clé publique énoncées dans la section précédente (c'est d'ailleurs pour cette raison que nous détaillons son fonctionnement ici).

Avant toute chose, introduisons quelques résultats et notions qui nous seront utiles pour le présenter.

Lemme 2.2.6. *L'équation $ax \equiv 1 \mod n$ (avec $a \in \mathbb{Z}_0$ et $n \in \mathbb{N}_0$) admet une solution si et seulement si $pgcd\,(a, n) = 1$. De plus, si une solution existe, alors elle est unique.*

Démonstration. La condition est nécessaire. Si x est solution de $ax \equiv 1 \mod n$, alors il existe $y \in \mathbb{Z}$ tel que $ax + ny = 1$. Posons $d = \text{pgcd}\,(a, n)$. Par définition, $d \,|\, a$ et $d \,|\, n$. On en tire que $d \,|\, ax + ny = 1$, d'où $d = 1$.

12. Création personnelle.

La condition est suffisante. Si $\text{pgcd}\,(a,n) = 1$, alors le théorème de Bezout implique qu'il existe $u, v \in \mathbb{Z}$ tels que $au + nv = 1$. En particulier, $au \equiv 1 \bmod n$, d'où $x \equiv u \bmod n$ est une solution de $ax \equiv 1 \bmod n$.

Pour l'unicité, supposons que l'équation admet deux solutions x et y. Alors, par hypothèse, $ax \equiv ay \equiv 1 \bmod n$. Notons u, v les entiers tels que $au + nv = 1$. Vu ce qui précède, il vient $uax \equiv uay \bmod n$. Or, $au \equiv 1 \bmod n$, ce qui implique $x \equiv y \bmod n$. □

Lemme 2.2.7. *Soient p et q deux nombres premiers distincts, et posons $n = pq$. Si $x, y \in \mathbb{Z}$ sont tels que*

$$y \equiv x \ mod\ p \quad et \quad y \equiv x \ mod\ q,$$

alors

$$y \equiv x \ mod\ n.$$

Démonstration. Par hypothèse, il existe $k_1, k_2 \in \mathbb{Z}$ tels que

$$y = x + k_1 p \quad \text{et} \quad y = x + k_2 q.$$

Ainsi, $k_1 p = k_2 q$. Comme p et q sont premiers et distincts, cette égalité implique d'une part $p \,|\, k_2$ et d'autre part $q \,|\, k_1$. Dès lors, il existe $l_1, l_2 \in \mathbb{Z}$ tels que

$$k_2 = l_2 p \quad \text{et} \quad k_1 = l_1 q.$$

Ainsi, l'égalité ci-dessus devient $l_1 pq = l_2 pq$, *i.e.* $l_1 = l_2$. On en tire alors que

$$y = x + l_1 pq = x + l_1 n, \ \textit{i.e.}\ y \equiv x \bmod n.$$

□

Définition 2.2.8. L'***indicatrice d'Euler*** est l'application

$$\phi : \mathbb{N}_0 \to \mathbb{N}_0 : n \mapsto \#\{m \in \mathbb{N}_0 \mid m \leq n \text{ et } \text{pgcd}\,(m,n) = 1\}.$$

Remarque 2.2.9. Dans l'anneau $\mathbb{Z}_n$, comme les éléments inversibles sont exactement les entiers premiers avec n, on tire de la définition précédente que le groupe multiplicatif $\mathbb{Z}_n^*$ est d'ordre $\phi(n)$.

Théorème 2.2.10 (Petit Théorème de Fermat)**.** *Si $a, n \in \mathbb{N}_0$ sont premiers entre eux, alors*

$$a^{\phi(n)} \equiv 1 \ mod\ n.$$

En particulier, si p est un nombre premier et si a est un entier non divisible par p, alors

$$a^{p-1} \equiv 1 \ mod\ p.$$

Démonstration. Puisque $\text{pgcd}\,(a,n) = 1$ par hypothèse, a est un élément du groupe $\mathbb{Z}_n^*$. Dès lors, l'ordre de a divise l'ordre de $\mathbb{Z}_n^*$, *i.e.* $o(a) \,|\, \phi(n)$. Ainsi, il existe $k \in \mathbb{Z}$ tel que $\phi(n) = k \cdot o(a)$, d'où

$$a^{\phi(n)} \equiv a^{k \cdot o(a)} \equiv \left(a^{o(a)}\right)^k \equiv 1 \bmod n,$$

puisque $a^{o(a)} \equiv 1 \bmod n$.

Quant au cas particulier, il découle directement du fait que $\phi(p) = p - 1$ pour tout nombre premier p. □

Description du RSA

Inventé en 1977, le ***RSA***, dont le nom provient de ses trois créateurs Ronald RIVEST, Adi SHAMIR et Leonard ADLEMAN (voir [44] pour l'article de référence), est un exemple typique de cryptosystème à clé publique qui est encore largement utilisé de nos jours (par exemple pour la sécurisation des transactions liées à la vente en ligne). Le choix des données est relativement simple :

1. choisir deux nombres premiers distincts p et q suffisamment grands, et calculer $n = p \cdot q$ (appelé ***module de chiffrement***),
2. calculer la valeur de l'indicatrice d'Euler en n : $\phi(n) = (p-1) \cdot (q-1)$,
3. choisir un entier positif $e < \phi(n)$ tel que $\operatorname{pgcd}(e, \phi(n)) = 1$, appelé ***exposant de chiffrement***,
4. déterminer l'entier d tel que $e \cdot d \equiv 1 \bmod \phi(n)$, appelé ***exposant de déchiffrement***, son existence et son unicité étant assurées par le Lemme 2.2.6.

Le couple (e, n) constitue la clé publique, et peut donc être divulgué, tandis que l'entier d correspond à la clé privée. Si on reprend les personnages d'Alice et de Bob, l'idée est alors la suivante :

1. phase de codage : Alice transforme le message clair m à transmettre en un entier [13] x tel que $0 \leq x < n$, *i.e.* on considère l'ensemble des messages clairs $\mathscr{P} = \{0, 1, \ldots, n-1\}$,
2. chiffrement : $e_k : x \mapsto x^e \bmod n$
   $\Rightarrow$ Alice calcule $y \equiv x^e \bmod n$, et envoie le résultat à Bob,
3. déchiffrement : $d_k : y \mapsto y^d \bmod n$
   $\Rightarrow$ Bob calcule $y^d \equiv x^{ed} \bmod n$, et comme $x \equiv x^{ed} \bmod n$, il peut retrouver le message de départ (qui est ici un entier) et donc, à l'aide du codage, le message clair m qu'Alice voulait lui transmettre,

où l'équivalence $x \equiv x^{ed} \bmod n$ est justifiée par la proposition suivante.

Proposition 2.2.11. *Avec les notations utilisées ci-dessus, on a*

$$x \equiv x^{ed} \bmod n.$$

Autrement dit, on a bien $x = d_k(e_k(x))$.

Démonstration. Comme $e \cdot d \equiv 1 \bmod \phi(n)$, il existe $k \in \mathbb{Z}$ tel que $ed = k\phi(n) + 1$. Dès lors,

$$x^{ed} \equiv x^{k\phi(n)+1} \equiv x^{k\phi(n)} \cdot x \bmod n.$$

Si x est premier avec n, alors le Petit Théorème de Fermat permet de conclure. Si x n'est pas premier avec n, alors il l'est nécessairement avec p ou q (sinon, x est multiple de $pq = n$, ce qui implique $x \geq n$ et contredit la condition sur x). Supposons sans perte de généralité qu'il est premier avec p. Alors, le Petit Théorème de Fermat implique

$$x^{k\phi(n)} \cdot x \equiv x^{k(p-1)(q-1)} \cdot x \equiv \left(x^{p-1}\right)^{k(q-1)} \cdot x \equiv 1 \cdot x \equiv x \bmod p.$$

Si x est également premier avec q, alors un raisonnement analogue montre que

$$x^{k\phi(n)} \cdot x \equiv x \bmod q.$$

Sinon, l'équivalence ci-dessus est trivialement satisfaite étant donné que les deux membres sont nuls.

13. Si le message est long, plusieurs entiers peuvent s'avérer nécessaires. Il suffit alors d'appliquer la suite de la procédure à chacun d'eux (nous y reviendrons).

Comme p et q sont deux nombres premiers distincts, le Lemme 2.2.7 permet finalement de conclure que

$$x^{k\phi(n)} \cdot x \equiv x \bmod n,$$

ce qui fournit bien le résultat attendu. □

Cet exemple illustre assez bien ce qui a été dit précédemment sur les cryptosystèmes à clé publique. En effet, d'une part, la connaissance des entiers p et q permet facilement de déterminer n, et donc les clés e et d (il s'agit de la fameuse *trap door* introduite plus tôt) : il s'agit simplement d'effectuer des produits et des exponentiations, ce qui est relativement simple. Néanmoins, d'autre part, la seule connaissance du couple (e, n) ne permet pas à un individu quelconque (qui n'a donc pas connaissance des nombres p et q) de déterminer facilement d. En effet, il lui faudrait d'abord factoriser n en un produit de nombres premiers ce qui, sans information suffisante et pour autant que n soit suffisamment grand, est supposé difficile. Là réside donc tout le côté à la fois pratique et sécurisé de ce genre de cryptosystème.

Petite parenthèse sur le codage

Comme mentionné dans la description du RSA, la première étape du protocole consiste à coder le message. Imaginons donc que le texte de départ d'Alice est un texte rédigé en français. A priori, la taille de l'alphabet A de départ n'excédera pas les 100 symboles (même si on compte les majuscules, minuscules, lettres avec accents et autres signes de ponctuation). Pour généraliser, notons N la taille de l'alphabet A de départ, avec [14] $N \ll n$. Considérons à présent un codage qui, à chaque symbole de l'alphabet de départ, associe un entier de $\mathbb{Z}_N$, comme cela a été fait dans l'Exemple 2.2.4. L'ensemble des textes clairs se limiterait donc à l'ensemble $\mathbb{Z}_N = \{0, \dots, N-1\}$. Pour un texte de longueur L, il suffirait alors à Alice de coder son message en utilisant L entiers $x_1, \dots, x_L$, et d'appliquer à chacun la procédure présentée.

Cependant, une telle façon de faire détruirait complètement toute la sécurité du RSA, et exposerait le message d'Alice à un très grand risque de divulgation. En effet, plaçons-nous un instant dans la peau d'Oscar, qui arrive à intercepter les messages chiffrés par Alice, à savoir les entiers $x_1^e, \dots, x_L^e$ modulo n. Comme la taille N de l'alphabet est limitée (nous parlions précédemment d'une centaine de symboles ce qui, d'un point de vue informatique, est parfaitement traitable rapidement), il lui suffit de coder chacun des symboles de l'alphabet de départ en l'entier $x \in \mathbb{Z}_N$ correspondant, puis de calculer $x^e \bmod n$, pour savoir quel entier chiffré correspond à quelle lettre de l'alphabet de départ, et ainsi déchiffrer le message d'Alice.

Pour éviter cela, il convient de profiter de l'envergure de l'intervalle d'entiers $[0, n-1]$ qui nous est offert pour coder le message, et donc trouver une autre méthode de codage, plus sécurisée. Une solution qui est proposée est alors de coder le message de base par blocs de taille constante k, où

$$k := \lfloor \log_N n \rfloor,$$

ce qui implique $N^k \leq n < N^{k+1}$. De la sorte, en commençant par associer chaque symbole à un entier de $\mathbb{Z}_N$ (comme proposé initialement), chaque bloc de k caractères consécutifs pourra être codé à l'aide d'un entier de $x \in \mathbb{Z}_{N^k}$ représenté en base N. En d'autres termes, étant donnés $x_0, \dots, x_{k-1} \in \mathbb{Z}_N$ des entiers codant les symboles $a_0, \dots, a_{k-1} \in A$, le bloc $a_0 \cdots a_{k-1}$ sera finalement codé par l'entier

$$x = x_{k-1} \cdot N^{k-1} + \cdots + x_1 \cdot N^1 + x_0 \cdot N^0 \in \mathbb{Z}_{N^k}\,.$$

14. Dans la pratique, le RSA utilise des nombres premiers p et q de l'ordre de 2^{512}, de sorte d'assurer la sécurité de la communication. Ainsi, même si l'on considère un alphabet de départ relativement grand, sa taille restera extrêmement petite par rapport à l'entier n considéré, qui est donc de l'ordre de 2^{1024}.

En particulier, $x < N^k \leq n$ (vu la valeur de k), et l'entier x est donc bien dans l'intervalle désiré [15]. Une fois le codage effectué, Alice doit ensuite chiffrer le nombre obtenu en calculant $y \equiv x^e \bmod n$. En particulier, $y < n < N^{k+1}$, et la représentation de y en base N peut donc être de longueur $k+1$. On en tire que chaque bloc de longueur k du message clair de départ sera envoyé sur un bloc de longueur $k+1$.

Pour y voir plus clair, considérons l'exemple suivant (tiré de [43]) qui, même s'il est peu réaliste (dans le sens où les valeurs considérées sont beaucoup trop petites), permet d'illustrer le codage présenté ci-dessus et le fonctionnement du RSA.

Exemple 2.2.12. Soient les entiers

$$p = 11, \quad q = 23 \quad \text{et} \quad n = p \cdot q = 253.$$

On a $\phi(n) = 10 \cdot 22 = 220$, on peut donc prendre $e = 3$. Pour déterminer la clé privée d, remarquons que $220 = 3 \cdot 73 + 1$. Dès lors,

$$3^{-1} \equiv -73 \bmod 220 \equiv 147 \bmod 220,$$

et on trouve donc $d = 147$. Considérons également l'alphabet $A = \{a, b, c, d\}$, et le codage $\varphi : A^* \to \{0, 1, 2, 3\}^*$ tel que

$$\varphi(a) = 0, \quad \varphi(b) = 1, \quad \varphi(c) = 2 \quad \text{et} \quad \varphi(d) = 3.$$

Dans cet exemple, $k = \lfloor \log_4 253 \rfloor = 3$.

Supposons qu'Alice désire envoyer le message

$$bccadb$$

à Bob. Pour ce faire, elle va commencer par découper ce dernier en blocs de 3 lettres, puis coder et chiffrer chacun de ces blocs. Pour le bloc *bcc*, elle obtient le nombre

$$\varphi(b) \cdot 4^2 + \varphi(c) \cdot 4^1 + \varphi(c) \cdot 4^0 = 1 \cdot 4^2 + 2 \cdot 4^1 + 2 \cdot 4^0 = 26 < 4^3 = 64.$$

Pour le chiffrement, elle calcule alors

$$26^3 \bmod 253 \equiv 119 \bmod 253,$$

et comme

$$119 = 1 \cdot 4^3 + 3 \cdot 4^2 + 1 \cdot 4^1 + 3 \cdot 4^0 = \varphi(b) \cdot 4^3 + \varphi(d) \cdot 4^2 + \varphi(b) \cdot 4^1 + \varphi(d) \cdot 4^0,$$

cet entier correspond au mot *bdbd*. En procédant de même avec le bloc *adb*, on trouve qu'il correspond à l'entier

$$0 \cdot 4^2 + 3 \cdot 4^1 + 1 \cdot 4^0 = 13,$$

et son chiffrement

$$13^3 \bmod 253 \equiv 173 \bmod 253, \quad \text{avec } 173 = 2 \cdot 4^3 + 2 \cdot 4^2 + 3 \cdot 4^1 + 1 \cdot 4^0,$$

nous donne donc le mot *ccdb*. Alice transmet donc le message

$$bdbdccdb$$

à Bob. Comme il dispose de la clé publique (notamment n), Bob sait que $k = 3$ et qu'il doit donc découper le mot reçu en blocs de longueur $3 + 1 = 4$. Il commence par le mot *bdbd*, qui correspond à l'entier 173. Il calcule donc

$$173^{147} \bmod 253 \equiv 26 \bmod 253,$$

et retrouve bien le bloc de départ *bcc* en utilisant le codage. En faisant de même avec *ccdb*, il retrouve le mot *adb*, et donc le message clair de départ envoyé par Alice.

15. Cela permet aussi de voir que la valeur de k ainsi définie est la plus grande que l'on puisse envisager pour un découpage en blocs.

2.3 Pourquoi les courbes elliptiques ?

En cryptographie, la sécurité des cryptosystèmes repose principalement sur le caractère supposé insoluble de deux problèmes mathématiques : le ***problème de la décomposition en facteurs premiers*** et le ***problème du logarithme discret***, dont les énoncés sont les suivants.

Problème 2.3.1 (Problème de la décomposition en facteurs premiers)**.** Étant donné un entier n produit de deux nombres premiers p et q, trouver un des deux facteurs du produit.

Exemple 2.3.2. Le RSA, présenté dans la section précédente, repose sur la complexité de la résolution de ce problème : étant donnée la clé publique n, il est pratiquement impossible de trouver les entiers p et q choisis initialement (pour autant que n soit suffisamment grand).

Définition 2.3.3. Soit G un groupe cyclique d'ordre n, et soit y un générateur de G, *i.e.* $G = \langle y \rangle$. Le ***logarithme discret*** de $x \in G$, noté $\operatorname{dlog} x$, est le plus petit entier $k \in \{0, \ldots, n-1\}$ tel que [16] $x = y^k$.

Problème 2.3.4 (Problème du logarithme discret)**.** Étant donnés un groupe cyclique G d'ordre n et un de ses générateur y, ainsi qu'un élément $x \in G$, déterminer $\operatorname{dlog} x$, *i.e.* trouver $k \in \{0, \ldots, n-1\}$ tel que $x = y^k$.

Enfin, il existe en réalité un dernier problème qui fait référence dans le domaine de la cryptographie : le ***problème du logarithme discret pour les courbes elliptiques***. Comme son nom le laisse sous-entendre, il s'agit de l'analogue du problème du logarithme discret dans le cadre d'un groupe formé par les points d'une courbe elliptique. Plus précisément, son énoncé est le suivant.

Problème 2.3.5 (Problème du logarithme discret pour les courbes elliptiques)**.** Étant donnés une courbe elliptique E sur le champ $\mathbb{F}_q$ et deux point $P, Q \in E$, trouver un entier x tel que $Q = xP$ si un tel x existe.

Bien que ces problèmes semblent simples en apparence, ils s'avèrent en réalité souvent difficiles à résoudre, notamment en termes de temps et de puissance de calcul à fournir. En particulier, ce sont les deux problèmes du logarithme discret (classique et dans le cas des courbes elliptiques) qui vont particulièrement nous intéresser ici.

L'avantage du problème dans le cas des courbes elliptiques est qu'il est significativement plus difficile à résoudre que son analogue classique. Cette caractéristique est notamment mise en évidence par Koblitz dans [23] et [24]. Ainsi, l'avantage des cryptosystèmes utilisant les courbes elliptiques est que, pour des paramètres ayant des valeurs semblables, ils seront beaucoup plus sécurisés que leurs analogues classique. De façon équivalente, on peut également dire que ces systèmes cryptographiques basés sur les courbes elliptiques permettent d'utiliser des paramètres ayant des valeurs beaucoup plus petites, tout en gardant un niveau de sécurité équivalent à leurs analogues classiques utilisant quant à eux des paramètres aux valeurs plus importantes. En particulier, cette propriété met en évidence tout l'intérêt de ces cryptosystèmes à courbes elliptiques. En effet, l'utilisation de plus petits paramètres permet notamment d'augmenter la vitesse de calcul (et donc une économie de temps) et de diminuer l'espace de stockage nécessaire, le tout ici sans affecter la sécurité du système ! On comprend donc aisément pourquoi les cryptologues se sont de plus en plus intéressés à ces cryptosystèmes.

16. Nous utilisons ici la notation multiplicative.

Bien sûr, difficile à résoudre ne veut pas dire impossible. En effet, plusieurs algorithmes permettant de résoudre le problème du logarithme discret classique ont vu le jour. Citons par exemple [17] la méthode trouvée par POHLIG, HELLMAN et POLLARD, qui permet d'obtenir le logarithme discret d'un élément d'un groupe G d'ordre n en un temps $O(\sqrt{p})$, où p est le plus grand nombre premier divisant n, ou encore l'algorithme d'ADLEMAN, qui permet de le déterminer en un temps $O(\exp(\log n \log\log n))$.

Comme l'ensemble des points d'une courbe elliptique forme un groupe, ces algorithmes peuvent aussi être utilisés pour trouver le logarithme discret d'un point de la courbe. En fait, jusque dans les années 90, ces algorithmes étaient d'ailleurs les seuls que l'on connaissait pour résoudre le problème du logarithme discret pour les courbes elliptiques. Depuis, des méthodes s'attaquant spécialement à ces courbes ont été mises en place. Par exemple [18], MENEZES, OKAMOTO et VANSTONE ont trouvé un processus permettant de plonger le groupe des points d'une courbe elliptique E définie sur un champ $\mathbb{F}_q$ dans le groupe multiplicatif d'une extension du champ $\mathbb{F}_{q^k}$, pour un certain $k \in \mathbb{N}_0$, ce qui permet de ramener le problème du logarithme discret pour une telle courbe au problème classique. Néanmoins, ceci ne fonctionne que si le ***degré d'extension*** k est petit, ce qui est le cas des courbes que l'on appelle ***supersingulières***. Sans entrer dans les détails [19], ce genre de courbes ne constitue heureusement pas la majorité des courbes elliptiques. En particulier, dans son ouvrage [24], Koblitz identifie deux familles de courbes supersingulières. Il s'agit des courbes définies sur un champ $\mathbb{F}_q$ de caractéristique p par une équation de la forme

1. $y^2 = x^3 + ax$, avec $p \equiv 3 \bmod 4$,
2. $y^2 = x^3 + b$, avec $p \equiv 2 \bmod 3$.

Il conviendra dès lors de ne pas sélectionner de telles courbes pour les cryptosystèmes.

En résumé. Les cryptosystèmes utilisant les courbes elliptiques sont particulièrement intéressants de par le fait qu'ils permettent l'utilisation de plus petits paramètres que leurs analogues classiques, tout en gardant un niveau de sécurité équivalent, cela grâce à la difficulté supposée de résolution du problème du logarithme discret. A ce jour, il n'existe d'ailleurs aucun algorithme sous-exponentiel général permettant de résoudre ce dernier dans le cas des courbes elliptiques.

Néanmoins, il convient de faire attention à la courbe choisie. On veillera notamment à ce qu'elle ne soit pas supersingulière, et que son ordre soit divisible par un nombre premier suffisamment grand (de sorte que les algorithmes de résolution du problème du logarithme discret classique soient inenvisageables en termes de temps et de puissance de calcul nécessaires).

Quant au champ sur lequel la courbe est définie, on choisira un champ de la forme $\mathbb{F}_p$, où p est un nombre premier suffisamment grand, ou $\mathbb{F}_{2^m}$, à nouveau avec $m \in \mathbb{N}_0$ relativement important.

2.4 Un exemple concret

Le but de la présente section est de décrire concrètement comment les courbes elliptiques peuvent être utilisées dans le cadre d'un cryptosystème connu. Ici, nous nous intéresserons au système de Massey-Omura, car il nous semble relativement parlant pour mettre en lumière le parallélisme entre l'algorithme originel, et son analogue en termes de courbes elliptiques.

17. Ces algorithmes ne seront pas détaillés ici, ceci sortant du cadre fixé pour ce travail. Le lecteur intéressé peut néanmoins consulter [15] pour les détails, ou encore les références de [24] et [31] pour retrouver les articles liés aux méthodes citées.

18. A nouveau, le lecteur intéressé est renvoyé à l'article de référence [29].

19. Des informations complémentaires sur les courbes supersingulières peuvent notamment être trouvées dans [21], [26] ou [47].

Bien sûr, d'autres cryptosystèmes peuvent être modifiés afin d'obtenir leurs analogues en termes de courbes elliptiques (citons notamment les systèmes de Diffie-Hellman et d'ElGamal, également fort connus), mais ceux-ci ne seront pas envisagés ici afin de ne pas allonger excessivement le contenu de ce chapitre. Néanmoins, le lecteur intéressé peut, entre autres, se référer à [13] ou [24] pour explorer d'autres exemples.

2.4.1 Quelques informations sur le système de Massey-Omura

Le système de Massey-Omura a été introduit en 1982 (voir [28] pour l'article de référence), par James Lee MASSEY (1934 – 2013), cryptologue américain, et Jimmy K. OMURA (1940 – ...), ingénieur électricien et théoricien de l'information, américain également. Le but était d'améliorer le *protocole de Shamir*, un algorithme de cryptographie déjà existant créé par Adi SHAMIR (1952 – ...), mathématicien et cryptologue israélien ayant contribué au RSA.

Le cryptosystème de Massey-Omura est un système à clé publique. Il s'agit également d'un protocole à trois passes, ce qui signifie que l'émetteur et le récepteur du message vont s'échanger ce dernier trois fois (sous différentes formes) avant que le récepteur ne puisse décoder entièrement le message [20].

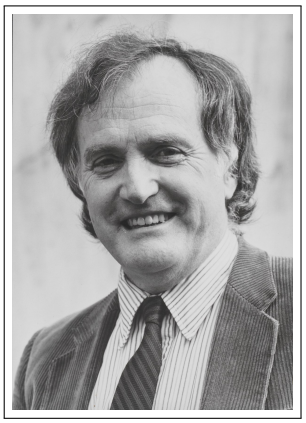

Figure 2.7 – James Lee MASSEY [21]

Figure 2.8 – Jimmy K. OMURA [22]

2.4.2 Algorithme originel

Nous décrivons ici, étape par étape, le fonctionnement du système cryptographique de Massey-Omura. A nouveau, nous considérerons que nos deux intervenants se nomment Alice et Bob, et que c'est Alice qui désire envoyer un message à ce dernier. Notons également que, dans les explications qui suivent, nous avons adopté la notation multiplicative.

1) Choix des données publiques

Les correspondants choisissent un alphabet A, ainsi qu'une taille maximale N pour la longueur d'un message (*i.e.* le nombre maximal de caractères pouvant être envoyés). Ils se mettent également d'accord sur un codage φ permettant de convertir les éléments de A en nombres et inversement. Notons ici que le fait d'imposer une taille maximale pour la longueur des messages n'est pas sans rappeler la méthode de codage décrite pour le RSA.

A partir de ces données, on détermine le plus grand entier L pouvant représenter un message (en fonction donc de A, N et φ). Enfin, on choisit un nombre premier $p > L$.

Les données A, N, φ, L et p sont alors rendues publiques.

20. Contrairement au RSA qui, lui, ne nécessite qu'un seul envoi d'information
22. Photo issue de <https://en.wikipedia.org/wiki/James_Massey>.
22. Photo issue de <https://www.xwhos.com/person/jim_k_omura-whois.html>.

2) Choix des clés de (dé)chiffrement

Chacun des correspondants choisit un entier e_i tel que $\text{pgcd}\,(e_i, p-1) = 1$. Cet entier sera la clé de chiffrement du correspondant i.

On définit ensuite la clé de déchiffrement de ce même correspondant comme étant l'entier d_i tel que $d_i \cdot e_i \equiv 1 \bmod (p-1)$, son existence et son unicité étant assurées par le Lemme 2.2.6 (e_i est premier avec $p-1$ par hypothèse).

Chaque correspondant i dispose ainsi d'une clé de chiffrement e_i et d'une clé de déchiffrement d_i. A l'inverse des données choisies à l'étape 1, qui étaient publiques, ces clés sont privées.

3) Premier chiffrement du message

Revenons à nos deux protagonistes, Alice et Bob. Notons respectivement e_A et d_A les clés de chiffrement et de déchiffrement d'Alice. De même, e_B et d_B sont respectivement les clés de chiffrement et de déchiffrement de Bob.

Alice construit son message m à partir de l'alphabet A, sans dépasser la longueur maximale N imposée précédemment. Elle convertit ensuite son message en son équivalent numérique M en utilisant le codage φ. Par définition, on a $M \leq L$.

Finalement, Alice chiffre une première fois son message M en calculant

$$M^{e_A} \bmod p,$$

et envoie ce résultat à Bob.

4) Deuxième chiffrement du message

Bob réceptionne le résultat et chiffre à nouveau le message en calculant

$$(M^{e_A})^{e_B} \equiv M^{e_A e_B} \bmod p.$$

Il renvoie alors ce résultat à Alice.

5) Premier déchiffrement du message

Alice reçoit donc le résultat, et commence à déchiffrer le message en calculant

$$(M^{e_A e_B})^{d_A} \equiv (M^{e_B})^{e_A d_A} \bmod p.$$

Par définition, on a

$$e_A d_A \equiv 1 \bmod (p-1).$$

Autrement dit, il existe $k \in \mathbb{Z}$ tel que $e_A d_A = 1 + k(p-1)$. Dès lors,

$$(M^{e_B})^{e_A d_A} \equiv (M^{e_B})^{1+k(p-1)} \equiv M^{e_B} \cdot \left(M^{p-1}\right)^{k e_B} \bmod p.$$

Or, $M \leq L < p$ par définition. Ainsi, comme p est premier, on a $\text{pgcd}\,(M, p) = 1$. On en tire alors que $M^{p-1} \equiv 1 \bmod p$ par le Petit Théorème de Fermat, et il vient

$$M^{e_B} \cdot \left(M^{p-1}\right)^{k e_B} \equiv M^{e_B} \bmod p.$$

Alice envoie alors ce résultat à Bob.

6) Déchiffrement complet du message

C'est à présent Bob qui va achever le déchiffrement du message initial envoyé par Alice, et donc découvrir son contenu ! Pour ce faire, Bob calcule

$$(M^{e_B})^{d_B} \equiv M^{e_B d_B} \bmod p.$$

On va alors effectuer un raisonnement analogue à celui de l'étape précédente. En effet, à nouveau par définition,

$$e_B d_B \equiv 1 \bmod (p-1),$$

i.e. il existe $l \in \mathbb{Z}$ tel que $e_B d_B = 1 + l(p-1)$. Ainsi,

$$M^{e_B d_B} \equiv M^{1+l(p-1)} \equiv M \cdot M^{l(p-1)} \equiv M \cdot \left(M^{p-1}\right)^l \bmod p.$$

En appliquant à nouveau le Petit Théorème de Fermat, on obtient que $M^{p-1} \equiv 1 \bmod p$. Dès lors,

$$M \cdot \left(M^{p-1}\right)^l \equiv M \bmod p.$$

Or, par définition, $M \leq L < p$, d'où $M \bmod p = M$.

Il suffit alors à Bob d'utiliser le codage φ pour reconvertir le nombre M et retrouver son analogue alphabétique m.

Remarque 2.4.1. Notons que, par rapport au RSA, le système de Massey-Omura nécessite deux fois plus d'opérations d'exponentiation, ce qui impacte nécessairement son efficacité. Ainsi, à puissance de calcul égale, l'envoi d'un message par l'intermédiaire de ce cryptosystème nécessitera plus de temps que le RSA.

2.4.3 Application au cas des courbes elliptiques

Maintenant que nous savons comment fonctionne le système de Massey-Omura, nous allons voir comment l'adapter au cas des courbes elliptiques.

Pour ce faire, reprenons nos deux correspondants, Alice et Bob, et décrivons étape par étape ce qu'ils doivent faire pour s'échanger un message auquel eux seuls auront accès.

1) Choix des données publiques

Alice et Bob se mettent d'accord sur une puissance suffisamment grande $q = p^n$ d'un nombre premier p, puis sur le choix d'une courbe elliptique E sur le champ $\mathbb{F}_q$ (de sorte que le problème du logarithme discret soit difficile à résoudre pour les données choisies). On pose alors $N = |E(\mathbb{F}_q)|$.

Les correspondants s'accordent également sur un plongement $\varphi : m \mapsto P_m$ qui à un message m associe un point P_m de la courbe E. En fait, on peut comparer le plongement φ au codage utilisé dans l'algorithme originel.

Toutes ces données sont donc supposées connues par nos deux correspondants, et peuvent même être rendues publiques.

2) Premier chiffrement du message

Supposons qu'Alice désire envoyer un message m à Bob. Elle y associe le point $P_m \in E$ par le plongement φ, puis elle choisit un entier x tel que $\operatorname{pgcd}(x, N) = 1$. Elle est la seule à connaître ce nombre. Elle calcule alors le point xP_m et l'envoie à Bob.

3) Deuxième chiffrement du message

Bob reçoit le point xP_m de la part d'Alice et sélectionne à son tour un entier y tel que $\text{pgcd}\,(y, N) = 1$, qu'il garde également secret. Il transmet alors le point $y(xP_m)$ à Alice.

Résultats intermédiaires

Avant de poursuivre la description avec la phase de déchiffrement du message, nous avons besoin de résultats intermédiaires qui nous assureront le sens de nos prochains calculs.

Lemme 2.4.2. *Si E est une courbe elliptique définie sur le champ fini $\mathbb{F}_q$, et si on note $N = |E(\mathbb{F}_q)|$, alors*

$$NP = \mathscr{O}$$

pour tout point P de E.

Démonstration. Soit P un point de E. Vu la Proposition 1.1.8, $o(P) \mid N$, *i.e.* il existe $k \in \mathbb{N}$ tel que $N = k \cdot o(P)$. Ainsi, par définition de $o(P)$,

$$NP = (k \cdot o(P))P = k \cdot (o(P) \cdot P) = k \cdot \mathscr{O} = \mathscr{O}.$$

□

Lemme 2.4.3. *En se plaçant dans les conditions d'une courbe elliptique E définie comme dans l'étape 1, si $n \in \mathbb{Z}$ est tel que $pgcd(n, N) = 1$, et si on note n^{-1} l'inverse de n dans $\mathbb{Z}_N$, alors*

$$n^{-1}nP = nn^{-1}P = P$$

pour tout point P de E.

Démonstration. Notons tout d'abord que l'existence de $n^{-1} \in \mathbb{Z}_N$ est assurée par le fait que n et N sont premiers entre eux (cfr. Lemme 2.2.6).

Par définition, $nn^{-1} \equiv 1 \bmod N$. Autrement dit, il existe $k \in \mathbb{Z}$ tel que $nn^{-1} = kN + 1$. De plus, comme $N = |E(\mathbb{F}_q)|$, on sait par le Lemme 2.4.2 que $NP = \mathscr{O}$ pour tout point $P \in E$. Dès lors, étant donné $P \in E$, on a

$$\begin{aligned} nn^{-1} = kN + 1 &\Rightarrow nn^{-1}P = (kN + 1)P \\ &\Rightarrow nn^{-1}P = kNP + P \\ &\Rightarrow nn^{-1}P = \mathscr{O} + P = P, \end{aligned}$$

puisque $\mathscr{O}$ est neutre pour l'addition définie sur E, ce qui nous fournit le résultat attendu. □

4) Premier déchiffrement du message

A la réception de l'information, Alice calcule l'inverse de x dans $\mathbb{Z}_N$, que nous notons x^{-1}. Remarquons que l'existence de x^{-1} est assurée par le fait que x est premier avec N par définition, et donc par le Lemme 2.2.6. Elle calcule alors le point $x^{-1}(yxP_m) = y(x^{-1}xP_m) = yP_m$, et envoie le résultat à Bob.

5) Déchiffrement complet du message

A son tour, Bob trouve $y^{-1} \in \mathbb{Z}_N$ (à nouveau, les hypothèses assurent son existence) et calcule alors $y^{-1}(yP_m) = P_m$ à partir de l'information reçue. Il peut alors utiliser φ pour décoder le message m qu'Alice voulait lui envoyer !

On voit donc que l'algorithme basé sur les courbes elliptiques est similaire à l'algorithme originel ! In fine, la méthode reste la même, seule la structure du groupe utilisé change.

Parenthèse sur le plongement φ

Nous exposerons ici une méthode permettant de convertir des messages (rédigés en français par exemple) en points de la courbe elliptique considérée. D'autres processus sont notamment présents dans [23].

Comme précédemment, considérons une courbe elliptique E définie sur un champ $\mathbb{F}_q$ de caractéristique $p \notin \{2, 3\}$ par l'équation

$$y^2 = x^3 + ax + b, \quad a, b \in \mathbb{F}_q\,.$$

L'idée est de trouver une méthode systématique qui, à un message m, associerait un point P_m de la courbe. En effet, fonctionner au "cas par cas" et attribuer aléatoirement des points à tous les messages possibles serait relativement contraignant [23]. Néanmoins, il est à noter qu'il n'existe à ce jour à notre connaissance [24] aucun algorithme déterministe permettant de coder un grand nombre de messages par les points d'une courbe elliptique en un temps polynomial (en $\log q$). Dès lors, l'algorithme que nous présenterons ici est probabiliste. Cependant, la probabilité d'échec est tellement faible que, dans la pratique, il peut parfaitement être utilisé.

Étape 1 : codage lettres $\leftrightarrow$ nombres. Cette première étape est en fait la même que pour le RSA (cf. Section 2.2.2). Étant donnés un alphabet A de départ de taille N (qui n'a ici rien à voir avec l'ordre du groupe $E(\mathbb{F}_q)$), une longueur de bloc $l \in \mathbb{N}_0$ ainsi qu'une bijection φ associant chaque lettre de A à un élément de $\mathbb{Z}_N$, chaque mot $w \in A^*$ de longueur l peut être associé à un élément x_w de $\mathbb{Z}_{N^l}$ écrit en base N. Ainsi, si $w = a_0 a_1 \cdots a_{l-1}$, avec $a_i \in A$ pour tout $0 \leq i \leq l-1$, alors

$$x_w = \varphi(a_{l-1}) \cdot N^{l-1} + \cdots + \varphi(a_1) \cdot N + \varphi(a_0), \quad \text{avec } 0 \leq x_w < N^l.$$

Étape 2 : codage nombres $\leftrightarrow$ points de E. L'idée maintenant est de trouver un moyen d'associer cet entier x_w à un point de la courbe E. Malheureusement, rien ne nous dit que x_w correspond à l'abscisse d'un point de la courbe (ce qui nous aurait arrangé, évidemment). Le but va donc être de trouver un entier $x_1 \in \mathbb{F}_q$ qui, d'une part, soit l'abscisse d'un point de la courbe et, d'autre part, soit relativement proche de notre entier x_w de départ. Plus précisément, nous aimerions qu'étant donné un entier positif k, la probabilité de trouver une abscisse x_1 d'un point de la courbe telle que $x_w \leq x_1 < x_w + k$ soit de $1 - \left(\frac{1}{2}\right)^k$.

Considérons donc $k \in \mathbb{N}_0$ tel que la probabilité de succès soit suffisamment grande [25] et $q > kN^l$ (on remarque ici un deuxième intérêt à considérer q suffisamment grand). Vu les conditions imposées, on obtient que $kx_w + j \in \mathbb{F}_q$ pour tout $0 \leq j < k$. L'idée est alors de passer en revue successivement chacun des entiers $kx_w + j$ $(0 \leq j < k)$ pour vérifier s'il correspond (ou non) à l'abscisse d'un point de la courbe. Pour ce faire, on calcule son image par la fonction $f : x \mapsto x^3 + ax + b$, et on vérifie s'il s'agit d'un carré de $\mathbb{F}_q$, auquel cas nous pourrons en extraire la racine carré et donc déterminer l'ordonnée correspondante. Le point $P_w(kx_w + j, f(kx_w + j)) \in E(\mathbb{F}_q)$ ainsi trouvé sera alors le point correspondant à l'entier x_w.

En ce qui concerne l'expression de la probabilité de succès mentionnée précédemment (à savoir $1 - \left(\frac{1}{2}\right)^k$), elle provient en réalité du fait que $f(x)$ est un carré de $\mathbb{F}_q$ pour environ 50% des $x \in \mathbb{F}_q$. En effet, on peut montrer [26] que le champ $\mathbb{F}_q$ contient exactement $\frac{q+1}{2}$ carrés, ce qui correspond à environ la moitié des éléments.

23. Un tel système nécessiterait sûrement de construire une table de conversion messages $\leftrightarrow$ points de la courbe ce qui, outre le coût et le temps considérable que cela engendrerait, serait dépourvu de toute sécurité.

24. Le problème a été exposé par Koblitz dans son ouvrage [24] en 1994. Néanmoins, nous n'avons trouvé aucun article mettant en lumière un algorithme répondant à la question posée.

25. Dans [24], Koblitz estime que $k = 30$ est une bonne valeur en pratique, quitte à aller jusque $k = 50$ dans le pire des cas.

26. Le lecteur intéressé peut notamment retrouver une preuve de ce résultat dans [39], p. 74.

Étape 3 : retrouver un nombre à partir d'un point de la courbe. Finalement, si on dispose d'un point $P_w(x_P, y_P)$ de la courbe, on peut facilement retrouver l'entier x_w (et donc le mot w) correspondant, étant donné que

$$x_w = \left\lfloor \frac{x_P}{k} \right\rfloor .$$

2.4.4 Un exemple simple

Considérons la courbe E définie sur $\mathbb{F}_5$ par l'équation

$$y^2 = x^3 + 2x + 1.$$

Comme pour le RSA, l'exemple que nous prenons ici n'est pas du tout réaliste, étant données les valeurs choisies pour les paramètres. Néanmoins, cet exemple sera suffisant pour illustrer l'algorithme de Massey-Omura.

On peut facilement vérifier que la courbe considérée est constituée de sept points, à savoir

$$E(\mathbb{F}_5) = \{\mathscr{O}, (0,1), (0,4), (1,2), (1,3), (3,2), (3,3)\}.$$

Le peu de points dont nous disposons nous limite assez fort quant à l'ensemble des messages clairs de départ possibles. Nous nous contenterons donc, dans cet exemple, du langage fini

$$\begin{aligned}\mathscr{L} = \{&\text{"Salut !","A bientôt !", "Je t'aime !", "Je possède 1 BTC.",}\\ &\text{"J'ai acheté un NFT !", "J'ai terminé mon mémoire !"}\},\end{aligned}$$

et du plongement $\varphi : \mathscr{L} \to E(\mathbb{F}_5)$ défini par

$$\begin{aligned}\varphi(\text{"Salut !"}) &= P_1(0,1),\\ \varphi(\text{"A bientôt !"}) &= P_2(0,4),\\ \varphi(\text{"Je t'aime !"}) &= P_3(1,2),\\ \varphi(\text{"Je possède 1 BTC."}) &= P_4(1,3),\\ \varphi(\text{"J'ai acheté un NFT !"}) &= P_5(3,2),\\ \varphi(\text{"J'ai terminé mon mémoire !"}) &= P_6(3,3).\end{aligned}$$

Notons que le point à l'infini ne peut pas coder un mot du langage, étant donné que $m\mathscr{O} = \mathscr{O}$ pour tout $m \in \mathbb{N}_0$.

Supposons qu'Alice, éperdument amoureuse de Bob, souhaite lui transmettre le message clair "Je t'aime !". Tout d'abord, elle y associe le point de la courbe correspondant, à savoir P_3. Ensuite, elle choisit se clé privée. Pour rappel, il s'agit d'un entier x tel que $\operatorname{pgcd}(x, N) = 1$, avec $N = |E(\mathbb{F}_5)| = 7$. Ainsi, $x = 2$ convient, par exemple. Elle calcule alors les coordonnées du point $2P_3$. On a

$$\lambda_{2P_3} = \frac{3 \cdot 1^2 + 2}{2 \cdot 2} = \frac{5}{4} \equiv 0 \bmod 5,$$

d'où

$$\begin{aligned}x_{2P_3} &= 0^2 - 1 - 1 = -2 \equiv 3 \bmod 5,\\ y_{2P_3} &= -0 \cdot 5 - 2 + 0 \cdot 1 = -2 \equiv 3 \bmod 5.\end{aligned}$$

Ainsi, $2P_3 = P_6$. Alice envoie donc le message "J'ai terminé mon mémoire !" à Bob [27].

27. Notons qu'ici, vu la taille du langage et de la courbe, il n'y a pas de grande différence entre envoyer directement le point P_6, ou son équivalent dans le langage $\mathscr{L}$. Nous choisissons ici la seconde option.

Bob reçoit le message, et doit lui aussi se créer une clé secrète. Il choisit par exemple $y = 5$. Il doit donc déterminer quel point de la courbe correspond à $5P_6$. Par des calculs analogues à celui effectué par Alice, il trouve successivement

$$2P_6 = P_2, \quad 3P_6 = P_4, \quad 4P_6 = P_3 \quad \text{et finalement} \quad 5P_6 = P_1.$$

Il renvoie donc le point P_1 (ou le message "Salut !") à Alice.

Pour effectuer le déchiffrement, Alice doit déterminer l'inverse de 2 dans $\mathbb{Z}_7$. Comme $2 \cdot 4 = 8 \equiv 1 \mod 7$, on en tire que $2^{-1} \equiv 4 \mod 7$. Alice doit donc calculer $4P_1$. Après calculs, elle tombe sur le point P_5 ("J'ai acheté un NFT !"), qu'elle renvoie à Bob.

Bob détermine lui aussi l'inverse de sa clé secrète : comme $5 \cdot 3 = 15 \equiv 1 \mod 7$, on a $5^{-1} \equiv 3 \mod 7$. Pour déchiffrer complètement le message initialement envoyé par Alice, Bob doit donc calculer les coordonnées du point $3P_5$. Il obtient finalement le point P_3, qui correspond bien au doux message d'amour "Je t'aime !" qu'Alice désirait lui envoyer !

Chapitre 3

La blockchain et son application au Bitcoin

A l'heure actuelle, nous avons tous déjà entendu parler de cryptomonnaies, et leur utilisation est d'ailleurs de plus en plus répandue. Le but du présent chapitre sera donc de décrire précisément le fonctionnement de l'une d'entre elles, le Bitcoin (qui, au passage, fut la toute première cryptomonnaie).

Après une brève introduction historique, nous nous intéresserons à quelques outils cryptographiques qui interviendront par après. Le fonctionnement de la blockchain, ici dans le cas particulier du Bitcoin, sera ensuite exposé progressivement et de façon naturelle. Nous terminerons finalement par une section liant la première partie de ce mémoire, à savoir les courbes elliptiques, à ce chapitre, en décrivant où elles interviennent dans la chaîne du Bitcoin.

3.1 Un peu d'Histoire : d'où vient le Bitcoin ?

De nos jours, le mot "Bitcoin" fait écho chez beaucoup de monde (pour ne pas dire tout le monde). Il ne semble donc pas inutile de s'intéresser à son.sa créateur.trice. Mais malheureusement, personne ne sait réellement qui il ou elle est [1]...

Une des seules informations dont nous disposons est que la personne à l'origine de la célèbre cryptomonnaie est mentionnée sous le pseudonyme de Satoshi NAKAMOTO. Comme les auteurs de [33] le précisent avec justesse, rien n'indique si cette personne est un homme ou une femme. Par facilité, nous emploierons les pronoms masculins dans la suite de ce texte, le pseudonyme "Satoshi" étant un nom masculin.

Bien que l'identité du créateur soit inconnue, nous disposons de plusieurs informations le concernant, celui-ci ayant été relativement actif au début du Bitcoin. D'après lui, il commence à coder la cryptomonnaie en mai 2007. En août 2008, il enregistre le nom de domaine `bitcoin.org` et, en octobre de la même année, il publie un "livre blanc" (*white paper* en anglais), dans lequel il détaille le protocole utilisé (voir [32]). Peu après, il publie le code originel du Bitcoin. Dans la foulée, il est également très actif par mail et sur les forums, autant pour trouver de potentiels futurs intéressés par le concept que pour répondre aux questions.

Des informations laissent également à penser que Satoshi Nakamoto ne serait pas un académique. En effet, la plupart du temps, les chercheurs réfléchissent d'abord à des idées, imaginent les problèmes qu'ils pourraient rencontrer et les mettent sur papier avant de finalement rédiger un compte-rendu de leurs découvertes. Satoshi, lui, a affirmé faire l'inverse : selon ses dires, il a d'abord rédigé tout le code du Bitcoin, avant de se convaincre qu'il pouvait résoudre les problèmes rencontrés et de rédiger son article.

1. Les informations de cette section sont tirées de la préface de [33].

Outre la question de son genre, d'aucuns se sont également demandé s'il s'agissait d'une seule personne ou d'un collectif. A cette question, les auteurs de [33] penchent davantage pour l'hypothèse d'une personne unique, compte tenu de la difficulté pour un groupe de devoir partager un unique et même compte pour dialoguer avec les gens, le tout sans se contredire.

Une autre question, et non des moindres, concerne la raison de cet anonymat : pourquoi donc ne pas révéler son identité ? A cela, plusieurs hypothèses sont également évoquées dans [33]. Cela pourrait simplement être "pour le fun", tout comme certains artistes (tel Banksy). A l'inverse, cette décision pourrait être motivée par une raison beaucoup plus sérieuse, par exemple d'un point de vue légal, ou de la sécurité personnelle du créateur. En effet, ayant amassé beaucoup d'argent suite au succès du Bitcoin, il n'est pas idiot de penser que cela puisse être une motivation pour garder l'anonymat. Cependant, comme le précisent très justement les auteurs, rester anonyme n'est pas une décision que l'on prend une fois pour toutes : l'anonymat est quelque chose que l'on décide de préserver au fur et à mesure du temps. Il y a donc fort à parier que, aux balbutiements du Bitcoin, ce n'est pas sa sécurité personnelle qui a motivé Satoshi Nakamoto à préserver son identité, mais peut-être plutôt la peur que son invention soit un échec...

Il reste donc beaucoup de zones d'ombre autour du créateur du Bitcoin. Cela peut paraître surprenant, surtout lorsque l'on connait l'importance capitale de cette cryptomonnaie dans le monde actuel. Mais qui sait ? Peut-être qu'un jour, nous saurons qui se cache derrière cette invention que d'aucuns qualifieraient de révolutionnaire...

Dans l'introduction de son article, Nakamoto affirme qu'une des idées principales qui a mené à la création du Bitcoin (et, par extension, à la création d'autres cryptomonnaies par d'autres personnes), était d'inventer un système de paiement ***décentralisé***, autrement dit ***de pair à pair***. Cela signifie que les transactions se font directement entre deux individus, sans passer par une quelconque institution principale qui gèrerait et vérifierait leur validité. Somme toute, cela s'oppose au fonctionnement des devises traditionnelles, pour lesquelles les transactions sont vérifiées par des institutions financières, les banques. Là où le système bancaire traditionnel est basé sur la confiance accordée à une institution tierce, les cryptomonnaies s'appuient quant à elles sur des preuves cryptographiques qui permettent d'éviter les fraudes tant que la majorité des intervenants participent honnêtement au réseau (nous y reviendrons dans la Section 3.3).

Figure 3.1 – Vision artistique d'un Bitcoin [2]

2. Photo issue de <https://www.cryptoify.news/2022/02/bitcoin-has-brought-in-more-than-1300.html>.

3.2 Quelques prérequis mathématiques et cryptographiques

3.2.1 Fonctions de hachage

Nous introduisons ici un outil mathématique dont l'utilité sera primordiale dans la blockchain : les ***fonctions de hachage***. Avant de les définir formellement, rappelons ici quelques notions de combinatoire des mots, que nous avons d'ailleurs déjà évoquées dans la Remarque 2.2.5.

Définition 3.2.1. Un ***alphabet*** est un ensemble fini dont les éléments sont appelés ***symboles*** ou ***lettres***.

Dans le cadre de ce mémoire, nous désignerons un alphabet par une lettre majuscule (typiquement, A puis B, etc.). Notons toutefois qu'un alphabet est souvent représenté par une lettre grecque majuscule, telle que Σ, Γ ou encore Λ.

Exemples 3.2.2. Les ensembles

$$A = \{0, 1\}, \quad B = \{a, b, c, d, e\} \quad \text{et} \quad C = \{\blacktriangle, \blacksquare, \spadesuit, \clubsuit\}$$

sont des alphabets.

Définition 3.2.3. Étant donné un alphabet A, un ***mot*** sur A est une suite ordonnée, finie ou infinie, de lettres de A. Un mot fini w de A peut donc s'écrire

$$w = w_1 w_2 \cdots w_n,$$

où $w_1, w_2, \ldots, w_n \in A$ et $n \in \mathbb{N}_0$. On définit alors la ***longueur*** d'un mot fini w, notée $|w|$, comme étant le nombre de lettres qui le composent, *i.e.* $|w| = n$ ici.
L'ensemble des mots finis sur A est noté A^*, et l'ensemble des mots de longueur n ($n \in \mathbb{N}_0$) est désigné par A^n.

Définition 3.2.4. Soit A un alphabet. L'opération de ***concaténation*** $\cdot$ sur A^* est définie de la façon suivante : pour tous mots $u, v \in A^*$, avec $u = u_1 \cdots u_k$ et $v = v_1 \cdots v_l$, la concaténation de u et v, notée $u \cdot v$ ou plus simplement uv, est le mot

$$w = w_1 \cdots w_{k+l}, \text{ où } \begin{cases} w_i = u_i & \text{si } 1 \leq i \leq k \\ w_{k+i} = v_i & \text{si } 1 \leq i \leq l \end{cases}.$$

Exemples 3.2.5. Si l'on considère l'alphabet $A = \{a, b, c\}$, alors $u = abba$ et $v = cbaacab$ sont des mots finis sur A, de longueur 4 et 7 respectivement. De plus, on peut construire le mot

$$u \cdot v = abba \cdot cbaacab = abbacbaacab,$$

qui est tel que $|u \cdot v| = |u| + |v| = 11$.

Nous pouvons également construire des mots infinis, par exemple

$$w = abcabcabcabc \cdots = (abc)^\omega,$$

où la notation $.^\omega$ désigne la concaténation infinie de la suite de symboles concernée. Notons cependant que, dans le cadre de ce mémoire, nous ne manipulerons que des mots de longueur finie.

Nous pouvons à présent définir ce qu'est une fonction de hachage. Dans la suite, et sauf indications contraires, nous supposerons que A désigne un alphabet quelconque.

Définition 3.2.6. Soient A,B deux alphabets et n un naturel non nul. Une ***fonction de hachage*** (*hash function* en anglais) est une fonction

$$H : A^* \to B^n$$

qui à tout mot fini de longueur quelconque associe un mot de longueur n, parfois appelé ***hash***.

Les fonctions de hachage que nous utiliserons par la suite sont en particulier des ***fonctions de hachage cryptographiques***. La différence entre une telle fonction et une fonction de hachage classique est que l'on impose, en plus, qu'elle doive satisfaire diverses propriétés. Ainsi, une fonction de hachage cryptographique [3] doit être

1. facilement calculable ; autrement dit, étant donné un mot fini x, son image $H(x)$ doit pouvoir être calculée en un temps raisonnable (et par "raisonnable", nous entendons ici un temps de l'ordre de $O(n)$ si n est la longueur du mot de sortie [4]),
2. telle qu'une petite modification du mot d'entrée entraîne un changement drastique de la valeur de sortie,
3. ***résistante aux collisions*** (*collision resistant* en anglais),
4. ***dissimulante*** (*hiding* en anglais), ou encore ***à sens unique*** [5],
5. ***puzzle friendly***.

Nous avons introduit ici beaucoup de nouveaux termes, que nous allons maintenant définir et expliciter afin d'y voir plus clair. Néanmoins, avant de poursuivre, il semble important ici de clarifier la terminologie qui sera utilisée dans les pages qui suivent. Par *"il est impossible de trouver"*, nous entendrons *"il est impossible de trouver* ***en pratique***", dans le sens où la mise en pratique de tel ou tel calcul ne pourrait se faire en un temps raisonnable.

Définition 3.2.7. Une fonction de hachage H est dite ***résistante aux collisions*** s'il est impossible de trouver deux mots distincts $x_1, x_2 \in A^*$ tels que $H(x_1) = H(x_2)$.

Remarque 3.2.8. La définition précédente est un bel exemple soulignant l'importance de la distinction terminologique mentionnée plus haut. En effet, si l'on considérait l'impossibilité de trouver deux mots distincts n'ayant pas la même image d'un point de vue purement théorique, la notion de résistance aux collisions correspondrait alors à celle d'injectivité. Or, une fonction de hachage n'est jamais injective, étant donnée que son ensemble de départ est infini et que son ensemble d'arrivée, lui, est fini (il contient $(\#A)^n$ éléments). Il existe donc forcément des mots de A^* qui ont la même image par la fonction H. Néanmoins, le temps moyen qu'il faudrait pour trouver au moins un de ces couples de mots dépasse l'imaginable : si on prend l'exemple d'une fonction de hachage qui donne en sortie des chaînes de 256 bits [6], il faudra en moyenne calculer 2^{128} hash ce qui, à un rythme de 10000 valeurs calculées par seconde, prendrait plus de 10^{27} ans à calculer [7] !

3. Dans la suite de ce travail, l'adjectif "cryptographique" sera sous-entendu lorsque nous parlerons de fonctions de hachage.

4. Pour rappel, étant données deux fonctions f et g, on note $f \in O(g)$ s'il existe $k > 0$ tel que $|f(n)| \leq k|g(n)|$ pour tout n suffisamment grand, *i.e.* lorsque $n \to +\infty$. Ici, le temps de calcul doit donc être inférieur à kn pour un certain $k > 0$. Rappelons aussi que le "temps" dans ce contexte correspond au nombre d'opérations nécessaires pour effectuer un calcul.

5. Cette terminologue semble davantage rencontrée dans la littérature scientifique française.

6. Un ***bit***, contraction de *binary digit* ou chiffre binaire en français, est un élément constitutif du système de numération binaire, qui ne peut donc prendre que deux valeurs, généralement 0 ou 1.

7. A titre de comparaison, l'âge de l'Univers est d'approximativement $13,8$ milliards ($\sim 10^{10}$) d'années.

Ainsi, dans la suite de ce mémoire, lorsque nous emploierons une tournure de la forme *"il est impossible de"*, il sera sous-entendu qu'*"il est extrêmement difficile en pratique de"*, toujours dans le sens où le temps nécessaire pour effectuer la procédure sera beaucoup trop important que pour l'envisager réellement.

Exemple 3.2.9. Comme dit dans la remarque précédente, il est au final toujours possible de trouver deux mots distincts qui auront la même image par la fonction de hachage considérée. Il existe néanmoins des fonctions plus "difficiles", dans le sens où la seule façon connue ou raisonnable de trouver un tel couple est de calculer des valeurs prises au hasard, jusqu'à tomber sur une paire qui convient. Ce sont ces fonctions qui sont qualifiées de résistantes aux collisions. Par exemple, la fonction

$$H : x \mapsto x \bmod 2^{256}$$

est une fonction de hachage (elle accepte tout mot fini [8] pour renvoyer un mot d'une longueur fixe de 256 bits) qui n'est pas résistante aux collisions, étant donné que les mots 1 et $1 + 2^{256}$ ont la même image et qu'il est très facile de les trouver (vu la forme de la fonction).

Avant d'introduire les notions suivantes, arrêtons-nous un instant sur la définition ci-dessous.

Définition 3.2.10. En théorie de l'information, l'***entropie min*** d'une variable aléatoire discrète X est une mesure de la prédictibilité des valeurs que peut prendre X. Plus formellement, si X peut prendre les valeurs $1, \ldots, n$ avec des probabilités respectives $p_1, \ldots, p_n$, alors son entropie min est définie comme étant la valeur

$$H_\infty(X) = \min_{1 \leq i \leq n} (-\log p_i) = -\max_{1 \leq i \leq n} (\log p_i) = -\log \left(\max_{1 \leq i \leq n} p_i \right).$$

Autrement dit, plus l'entropie min est grande, plus les probabilités $p_1, \ldots, p_n$ sont petites, et on peut alors dire que la distribution de la variable aléatoire est relativement étendue. Par exemple, s'il nous est demandé de choisir aléatoirement un mot d'une longueur de 256 bits, la probabilité d'obtenir une valeur particulière est exactement $1/2^{256}$ (étant donné que, pour chaque bit, la probabilité d'avoir 0 ou 1 est de 1/2), et la variable aléatoire X associée à cette expérience aura donc une forte entropie min : si l'on considère le logarithme en base 2, on obtient $H_\infty(X) = 256$.

Cette notion permet alors de définir la propriété suivante que doit satisfaire une fonction de hachage.

Définition 3.2.11. Une fonction de hachage H est dite ***dissimulante*** ou ***à sens unique*** si, étant donnée une valeur de hachage $H(rx)$, où r est une valeur secrète choisie aléatoirement au sein d'une distribution de probabilités ayant une forte entropie min [9], il est impossible de trouver x.

Remarque 3.2.12. D'aucuns pourraient se demander pourquoi ne pas avoir simplement imposé, dans la définition précédente, qu'étant donné $H(x)$, il doit être impossible de trouver x. La raison est simple. Imaginons une expérience qui consiste à jouer à pile ou face (un lancer) avec une pièce de monnaie classique. On dispose également d'une fonction de hachage qui, à chaque mot "pile" ou "face", associe un hash différent. Si on demande à un individu qui n'a pas vu le jeu, mais qui dispose de la valeur de hachage correspondante, d'en déterminer le résultat, il lui suffira de calculer les images par la fonction des deux entrées possibles (à savoir "pile" ou "face") et de voir laquelle correspond aux informations dont il dispose.

8. Rappelons ici qu'un nombre entier peut être vu comme un mot en considérant sa représentation dans une certaine base.

9. Nous n'avons malheureusement pas trouvé de valeur permettant de quantifier avec précision ce qu'est une "forte" entropie min.

Le problème de l'exemple donné ci-dessus est que les valeurs possibles de l'expérience ont toutes les deux une importante probabilité d'apparition (à savoir 1/2). Le fait d'ajouter, devant les mots "pile" ou "face", une suite d'une longueur déterminée mais provenant d'une distribution à forte entropie min permet de contourner ce problème : en faisant cela, un individu ne disposant que de la valeur de hachage ne pourra plus déterminer avec certitude l'entrée de la fonction de hachage, puisque le nombre d'entrées possibles sera beaucoup plus important, et que chacune d'entre elles aura une probabilité d'apparition beaucoup plus faible. Il aura bien sûr toujours 1 chance sur 2 de réussir, mais aussi de se tromper.

Définition 3.2.13. Une fonction de hachage H est dite ***puzzle friendly*** si pour toute valeur de sortie y d'une longueur de n bits, étant donnée une valeur r choisie aléatoirement au sein d'une distribution de probabilités ayant une forte entropie min, il est impossible de trouver x tel que $H(rx) = y$ en un temps significativement plus petit que 2^n.

Remarque 3.2.14. La définition précédente est relativement proche de la Définition 3.2.11, mais elle est néanmoins différente. Pour clarifier un peu la situation, reformulons les deux définitions un peu moins formellement :

- d'un côte, le fait qu'une fonction H soit à sens unique nous dit qu'il est très compliqué de retrouver le mot d'entrée rx si on dispose uniquement de la valeur de sortie y de la fonction (le mot r est donc inconnu),
- de l'autre, si elle est puzzle friendly, cela signifie qu'il est particulièrement difficile de trouver une valeur d'entrée x qui, si elle est concaténée avec un mot r connu, est envoyée précisément sur la valeur de sortie y donnée ou, en d'autres termes, il n'existe pas de meilleure stratégie que de simplement tester aléatoirement différentes valeurs possibles de x jusqu'à ce que l'on tombe sur une qui fonctionne.

Ainsi, là où r est une contrainte imposée dans la propriété de puzzle friendliness, ce n'en est pas une dans celle d'une fonction à sens unique. La distinction entre les deux notions s'éclaircira encore davantage lorsque nous les illustrerons concrètement dans le cadre de la blockchain du Bitcoin.

Un exemple de fonction de hachage : SHA-256

De façon générale, pour construire une fonction de hachage (qui, pour rappel, doit accepter en entrée des mots de longueur quelconque), on peut partir d'une fonction qui n'accepte que des mots d'une longueur fixée. Cette dernière est appelée ***fonction de compression***. Parmi les procédés qui existent, on peut notamment citer la construction de Merkle-Damgård, que nous allons décrire ici. L'avantage de cette méthode est que, si la fonction de compression est résistante aux collisions, alors la fonction de hachage construite à partir de celle-ci le sera également (cela a été démontré indépendamment par Merkle et Damgård en 1989 dans leurs articles, voir [8] et [30] pour les lecteurs intéressés).

Supposons disposer d'une fonction de compression f qui prend en entrée des mots de longueur m et qui renvoie des mots de longueur n, avec $m > n$. Pour construire une fonction de hachage H à partir d'une telle fonction de compression en utilisant la construction de Merkle-Damgård, considérons un mot x de longueur quelconque. Si cette longueur n'est pas un multiple de $m - n$, alors on effectue un ***remplissage*** ou ***bourrage*** (*padding* en anglais), qui consiste à compléter le message x de départ afin qu'il ait la taille désirée pour l'algorithme (en l'occurrence, un multiple de $m - n$). En pratique, une solution simple consisterait à ajouter des 0 à la fin du message. Néanmoins, cela poserait quelques problèmes en termes de sécurité, mais aussi de collisions (par exemple, les mots x et $x0$ seraient remplis de la même manière, et auraient par conséquent le même hash, ce qui est problématique). Pour prévenir cela, on ajoute, en plus des 0, un 1 au début du mot de remplissage, ainsi que la longueur du mot initial x à la fin.

Passée cette étape, on procède alors comme suit :

1. on se donne un ***vecteur d'initialisation*** b_0, qui est simplement un mot de longueur n,
2. le mot x de départ (éventuellement rempli) est divisé en blocs $b_1, \ldots, b_l$ de longueur $m-n$,
3. tour à tour, chaque bloc est concaténé avec le hash du bloc précédent par la fonction f, *i.e.* on calcule[10] $f(b_i f(b_{i-1}))$ pour tout $i \in \{1, \ldots, l\}$ (pour le premier bloc, on utilise donc le vecteur d'initialisation),
4. on pose finalement $H(x) = f(b_l f(b_{l-1}))$, *i.e.* l'image de x correspond à la valeur de sortie du dernier bloc par la fonction f.

La fonction de hachage SHA-256 (où SHA est l'acronyme de *Secure Hash Algorithm*) utilise la construction de Merkle-Damgård. En effet, elle est construite à partir d'une fonction de compression qui prend en entrée des mots d'une longueur de 768 bits et qui fournit des valeurs de sortie d'une longueur de 256 bits. Pour obtenir l'image d'un mot de longueur quelconque, on doit donc le découper en blocs de 512 bits. La figure ci-dessous, tirée de [33], représente le fonctionnement de SHA-256.

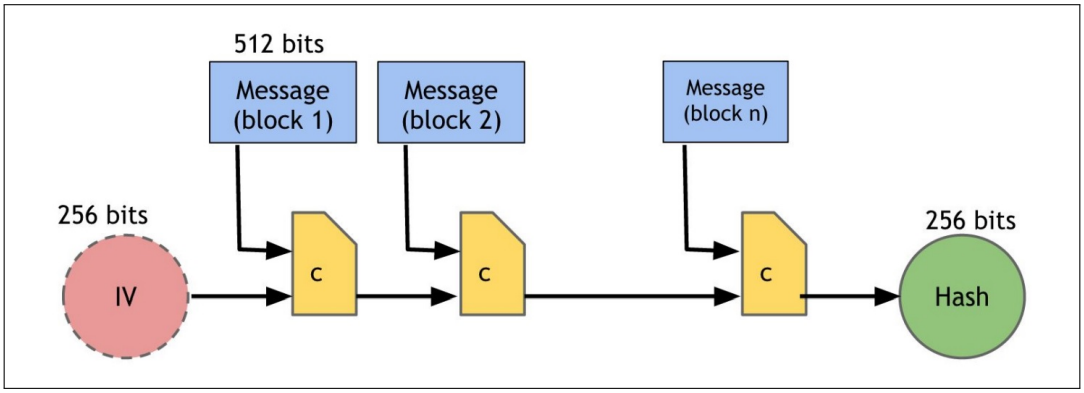

FIGURE 3.2 – Fonctionnement de la fonction de hachage SHA-256.

Nous ne décrirons pas ici explicitement le fonctionnement de la fonction SHA-256, mais le lecteur intéressé peut notamment se référer à [6] pour en apprendre plus. Nous invitons également le lecteur à se rendre sur

`<https://emn178.github.io/online-tools/sha256.html>`

s'il souhaite obtenir le hash d'un message particulier par l'intermédiaire de cette fonction. On remarquera d'ailleurs que les mots de sortie de la fonction sont des suites de 64 caractères, qui représentent en réalité l'entier de 256 bits correspondant dans le système hexadécimal.

3.2.2 La blockchain

Dans cette section, nous introduisons la notion de blockchain, sur laquelle repose le réseau Bitcoin (cf. Section 3.3). Avant de la définir formellement, introduisons un objet étroitement lié aux fonctions de hachage présentée ci-dessus.

Définition 3.2.15. En programmation, un ***pointeur*** est une variable contenant l'adresse mémoire d'une donnée. En d'autres termes, un pointeur contient l'information de l'endroit où est stockée une certaine donnée dans la mémoire.
Un ***pointeur de hachage*** (*hash pointer* en anglais) est un pointeur contenant en plus un hash de l'information pointée.

10. Les mots créés en concaténant sont de longueur $(m-n)+n = m$, on peut donc bien leur appliquer la fonction de compression f.

Remarque 3.2.16. L'avantage d'un pointeur de hachage est que, non seulement il permet d'indiquer l'endroit de stockage d'une information, mais également de vérifier que les données concernées n'ont pas été falsifiées. En effet, la moindre modification, ne serait-ce que d'une partie des données, entraînerait un changement radical du hash obtenu, et serait donc facilement repérable.

Définition 3.2.17. Une ***blockchain*** (ou ***chaîne de blocs*** en français) est une structure de données organisée en blocs qui se suivent, au sein de laquelle chaque bloc est relié au précédent à l'aide d'un pointeur de hachage. Le pointeur visant le dernier bloc de la chaîne est appelé la ***tête***.

Pour y voir peut-être plus clair, voici une illustration théorique (également tirée de [33]) de ce qu'est une blockchain.

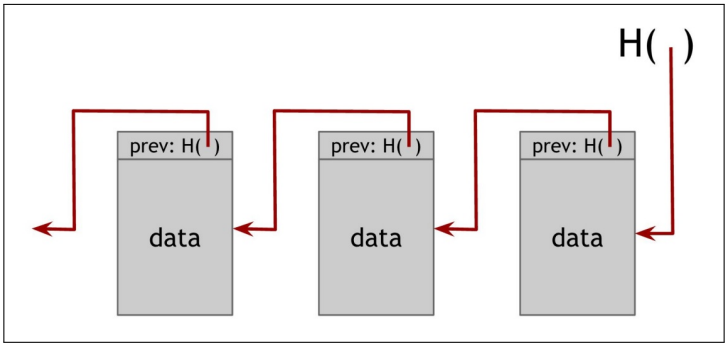

FIGURE 3.3 – Fonctionnement d'une blockchain.

A nouveau, tout l'intérêt de cette structure repose dans l'utilisation des pointeurs de hachage. En effet, outre le fait qu'un pointeur va désigner l'emplacement du bloc précédent, il va aussi nous permettre de nous assurer que les informations d'un bloc n'ont pas été modifiées (cf. Remarque 3.2.16).

Mieux encore, la structure même d'une blockchain est conçue pour repérer rapidement toute falsification potentielle des données. En effet, imaginons qu'Oscar modifie légèrement une information dans un des blocs de la chaîne. Dans ce cas, le pointeur de hachage associé (qui est donc situé dans le bloc suivant) s'en retrouve également modifié. Par conséquent, l'information du bloc suivant est elle aussi changée, ce qui entraîne une modification du pointeur suivant, et donc du bloc suivant, etc. Autrement dit, tous les blocs comptés à partir du bloc corrompu se retrouvent modifiés (de part la transformation des pointeurs de hachage), et donc en particulier la tête de la chaîne. Le schéma suivant illustre la situation décrite.

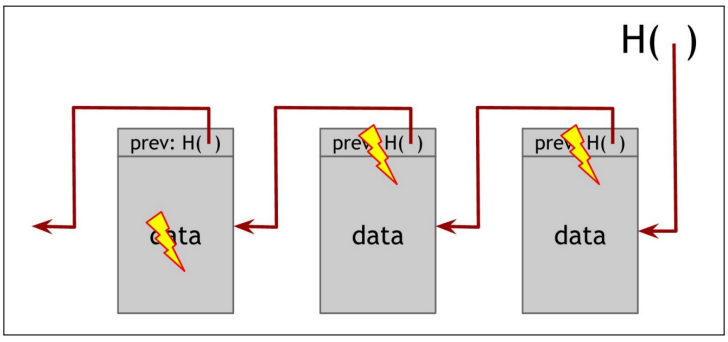

FIGURE 3.4 – Effet d'une modification d'un bloc de la chaîne.

Dès lors, toute modification d'une donnée de la chaîne, peu importe sa position, entraînera un changement radical de la tête et sera donc rapidement détectée ! Il suffira alors de repérer précisément quel bloc pose problème.

3.2.3 Signatures digitales

En cryptographie, le problème de l'identité des individus se pose assez souvent : comment peut-on être sûr que le message reçu par Bob a bien été envoyé par Alice, et non pas par Oscar, qui aurait pu intercepter le message d'origine et en transmettre un autre à Bob pour diffuser de fausses informations ? A l'instar d'une signature manuscrite figurant au bas d'un document officiel et attestant tantôt notre accord, tantôt notre identité, les signatures digitales permettent d'attester qu'une information provient bien de telle ou telle personne.

Idéalement, tout comme une signature manuscrite, un individu particulier doit être le seul à pouvoir produire sa signature digitale, mais n'importe quelle autre personne doit pouvoir attester son authenticité. Nous exigerons également d'une signature digitale qu'elle soit liée à l'information signée, de sorte qu'un individu en possession de la signature d'autrui ne puisse pas signer tous les documents qu'il souhaite.

En pratique, le procédé de création de signatures digitales fait intervenir les trois algorithmes suivants :

1. un algorithme de génération de clés `GenerateKeys`, qui prend en entrée un entier représentant la taille des clés souhaitée et qui renvoie un couple de clés (s_k, p_k), où s_k est une clé secrète connue uniquement de l'individu concerné et qui lui servira à signer ses messages, et p_k est une clé publique qui permettra aux autres de vérifier l'authenticité de la signature s_k,
2. un algorithme de signature `Sign`, qui prend en entrée un message m et une clé secrète s_k, et qui renvoie une signature correspondant à ces deux paramètres,
3. un algorithme de vérification `Verify`, qui prend en entrée un message m, une clé publique p_k ainsi qu'une signature, et qui renvoie la valeur "vrai" si la signature encodée est valable pour les données m et p_k (*i.e.*, si le message a bien été signé avec la clé secrète correspondante et pour le même message), et "faux" sinon.

Notons qu'il est préférable que les deux premiers algorithmes soient ***probabilistes*** ou ***randomisés***, *i.e.* qu'ils utilisent une part de hasard. Ce faisant, cela permettra de générer des clés et des signatures de façon aléatoire, même si l'on entre une même longueur de clé pour différents individus. L'algorithme de vérification, lui, doit être ***déterministe***, *i.e.* il renverra toujours la même sortie pour les mêmes données entrées.

On exigera alors également que, si une signature est valide, alors elle vérifie l'égalité

$$\texttt{Verify}(m, p_k, \texttt{Sign}(m, s_k)) = \text{vrai}.$$

pour tout message m. De plus, on impose que les signatures ne puissent pas être contrefaites : si Oscar a en sa possession la clé publique p_k d'Alice, ainsi que ses signatures pour certains messages $m_1, \ldots, m_n$, il ne doit pas être capable de signer un autre message m pour lequel il n'a pas connaissance de la signature d'Alice. En fait, la seule possibilité pour Oscar de signer le message m à la place d'Alice serait d'essayer toutes les solutions possibles jusqu'à ce qu'il tombe sur la bonne, ce qui est impossible dans la pratique [11]. Cette seconde propriété peut d'ailleurs être testée par l'intermédiaire d'un jeu (en référence à la théorie des jeux) ; le lecteur intéressé peut consulter [33] pour plus d'informations.

Un exemple de protocole de signature digitale sera abordé dans la Section 3.4, qui fera le lien entre les courbes elliptiques et le Bitcoin.

11. Si l'on considère une signature de 256 bits, cela correspond à 2^{256} possibilités différentes ! Penser pouvoir toutes les passer en revue relèverait de la folie. L'auteur de [1] dit d'ailleurs très justement que *"dire d'un tel nombre qu'il est astronomique reviendrait à donner beaucoup trop de crédit à l'astronomie"*.

3.3 Le Bitcoin

Afin d'amener progressivement et naturellement les différentes notions liées au Bitcoin et, plus généralement, à la blockchain, nous allons utiliser le même canevas que celui utilisé dans la vidéo [1], que nous trouvons particulièrement pertinente d'un point de vue pédagogique.

La situation sera donc la suivante. Imaginons quatre individus (Alice, Bob, Carole et David) qui souhaitent mettre en place un moyen efficace de gérer leurs dépenses et remboursements. Nous allons partir du plus simple protocole pouvant être mis en place, et le complexifier progressivement en tenant compte des différents problèmes que nos quatre amis pourraient rencontrer. Ce faisant, nous dresserons en quelque sorte la *"wishlist"* du Bitcoin, reprenant tous les éléments qui doivent intervenir dans la chaîne afin d'obtenir un protocole sûr, efficace et sécurisé. Nous entamerons ensuite une description plus formelle de la cryptomonnaie.

Notons que l'objectif de la présente section n'est pas d'analyser les éléments économiques en lien avec les cryptomonnaies (comme les taux de change ou les méthodes concernant la meilleure façon d'investir dans ce genre de devises), mais bien de comprendre ce qu'est exactement un Bitcoin et comment il fonctionne. En fait, comme le mentionne très justement l'auteur de [1], ces deux aspects d'une cryptomonnaie (à savoir fonctionnement par opposition à utilisation) sont relativement indépendants l'un de l'autre : on peut parfaitement utiliser de la cryptomonnaie sans savoir réellement comment cela fonctionne, tout comme l'on peut payer avec des devises traditionnelles sans connaître toutes les ficelles de notre système bancaire.

3.3.1 Le Bitcoin, qu'est-ce que c'est ?

Situation de départ

Comme annoncé, nous considérons la situation suivante [12]. Quatre amis, que nous appellerons Alice, Bob, Carole et David, souhaitent mettre en place un moyen efficace de gérer leurs dépenses et remboursements. Bien entendu, le réseau Bitcoin est beaucoup plus étendu, puisqu'il touche des millions d'utilisateurs, et ce dans le monde entier. Néanmoins, ce que nous décrirons pour nos quatre individus se transposera aisément à un nombre plus important d'utilisateurs.

Partons donc de la méthode la plus simple à laquelle on puisse penser : un grand livre de compte (*ledger* en anglais). Le principe est, plutôt que d'échanger de l'argent à tout va, de noter toutes les dépenses et remboursements effectués afin d'en garder une trace écrite. Ce livre doit être accessible à tous et modifiable par tous (dans le sens où chacun peut y ajouter une transaction). Au bout d'un certain temps (par exemple, chaque semaine, chaque mois, etc.), on fait les comptes et on voit qui doit payer et qui doit recevoir de l'argent.

<u>Problème n°1 :</u> Tout le monde peut écrire dans le livre.

Comme décrit précédemment, l'intérêt d'un tel livre de compte est que tout le monde puisse le modifier et ajouter des lignes spécifiant une transaction. Dès lors, partant de ce constat, qu'est-ce qui pourrait empêcher Bob d'écrire une ligne spécifiant que quelqu'un lui doit de l'argent alors que ce n'est pas le cas, par exemple "Alice paye 100€ à Bob" ?

Se reposer sur la confiance que l'on accorderait à autrui est malheureusement une idée utopique. Pour éviter ce genre de situation, nous allons donc faire intervenir un premier élément cryptographique vu dans la Section 3.2 : les signatures digitales. Au vu de la définition donnée et des propriétés que doit satisfaire une signature digitale, cela permettrait d'affirmer qu'une ligne de transaction a bien été créée par la personne concernée (dans notre cas, Alice) et, qui plus est, que personne d'autre n'aurait pu contrefaire sa signature pour frauder.

12. Les illustrations de cette section sont des créations personnelles, inspirées des schémas présents dans [1].

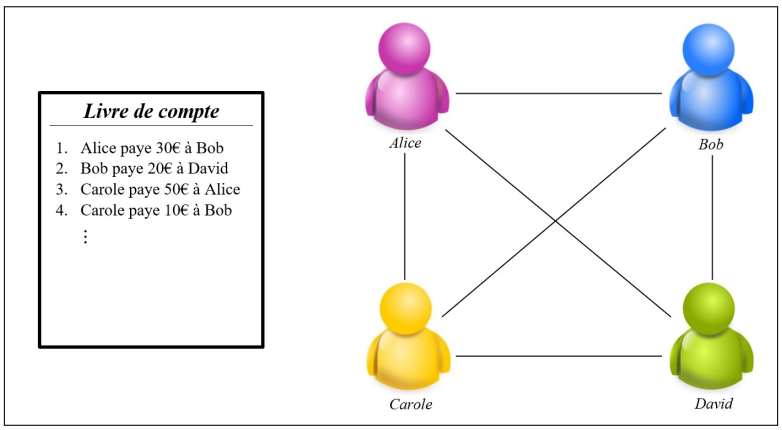

FIGURE 3.5 – Situation de départ.

De plus, l'avantage des signatures digitales est que tout le monde peut vérifier l'authenticité de la transaction. En effet, comme tout le monde est en possession de la clé publique d'Alice, un individu extérieur à la transaction tel que Carole peut aisément vérifier sa validité (dans le sens où elle a bien été écrite par Alice), simplement en exécutant l'algorithme de vérification `Verify` introduit dans la section précédente.

<u>Problème n°2 :</u> Bob peut alors juste recopier la ligne de transaction signée par Alice autant de fois qu'il le souhaite.

En effet, même si Bob ne peut copier la signature d'Alice sur n'importe quelle transaction (puisque celle-ci dépend de la transaction elle-même), il pourrait très bien copier et coller la transaction "Alice paye 100€ à Bob" autant de fois qu'il le souhaite !

Pour contourner cela, on va simplement ajouter à la transaction une certaine valeur faisant office d'"identité". De la sorte, même si une transaction est effectuée plusieurs fois (dans le sens où le même montant serait transféré au sein d'un même couple d'individus), chacune d'entre elles devrait être associée à une signature différente qui permettra de les distinguer. Dès lors, Bob ne pourra pas voler d'argent à Alice simplement en copiant une transaction effectuée une fois auparavant.

<u>Problème n°3 :</u> Que faire si quelqu'un a des dettes qu'il refuse de payer ?

C'est en effet une situation qui pourrait se produire : à la fin du mois, même si toutes les transactions ont été correctement enregistrées, signées et validées par tout le monde, qu'est-ce qui nous garantit qu'un individu, Bob par exemple, devant de l'argent à d'autres, règlera effectivement bien ses dettes ? A nouveau, une partie du système repose ici sur la confiance accordée à autrui...

Comme le mentionne très justement l'auteur de [1], au final, la seule raison qui va pousser nos quatre individus à revenir à de l'argent réel pour régler leurs comptes est le fait qu'un ou plusieurs d'entre eux doive(nt) beaucoup d'argent aux autres. En effet, si le système mis en place prévoyait que Bob ne puisse pas dépenser plus que ce qu'il ne possède effectivement, une telle situation ne pourrait ne produire.

Un système qui pourrait donc être mis en place, en plus du livre de compte, est celui du pot commun. Chaque individu verserait une même somme au début du mois, en ayant conscience qu'il ne pourra pas dépenser plus d'argent que ce qu'il n'injecte dans le pot. Les premières lignes du livre de compte correspondraient alors au montant disponible de chacun, par exemple "Alice obtient 100€", et une transaction serait considérée comme invalide si elle requiert des fonds qu'un individu ne possède pas.

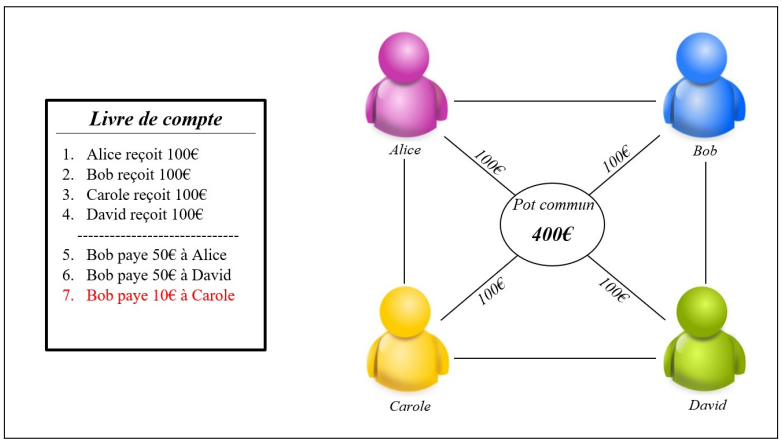

FIGURE 3.6 – Système du pot commun.

Considérons par exemple la situation ci-dessus. Ici, chaque individu a un budget de 100€. Les deux premières transactions de Bob sont valables (pour autant qu'elles soient correctement signées), étant donné qu'il dispose d'assez d'argent. La troisième, par contre, ne sera pas validée, car il n'a plus assez de fonds.

<u>Problème n°4 :</u> Il faut donc conserver un historique de toutes les transactions effectuées...

Pour pouvoir valider ou non une transaction en fonction des fonds dont un individu dispose, il est nécessaire de connaître les gains et dépenses précédents le concernant. Il faut donc conserver une trace de toutes les transactions ayant cours. En particulier, c'est ce problème qui va nous mener à la notion de ***cryptomonnaie***.

En effet, dans l'idée, si tous les individus concernés utilisent le livre de compte pour effectuer leurs paiements, et en tenant compte du fait qu'une personne ne peut dépenser plus qu'elle ne possède, quel serait l'intérêt de revenir aux devises traditionnelles ? C'est alors là qu'intervient la notion de ***cryptomonnaie*** : comme nous en avons besoin, c'est l'historique de toutes les transactions qui va jouer le rôle de devise. Autrement dit, on peut en quelque sorte voir un Bitcoin (tout comme une autre cryptomonnaie) comme une suite de transactions enregistrées et validées.

Dès lors, si nous revenons à nos quatre amis, à la place d'effectuer des transactions en euros dans le livre de compte, nous utiliserons une autre unité à valeur monétaire, que nous pourrions par exemple appeler l'EuroCoin (EC), et qui remplacera la devise traditionnelle utilisée.

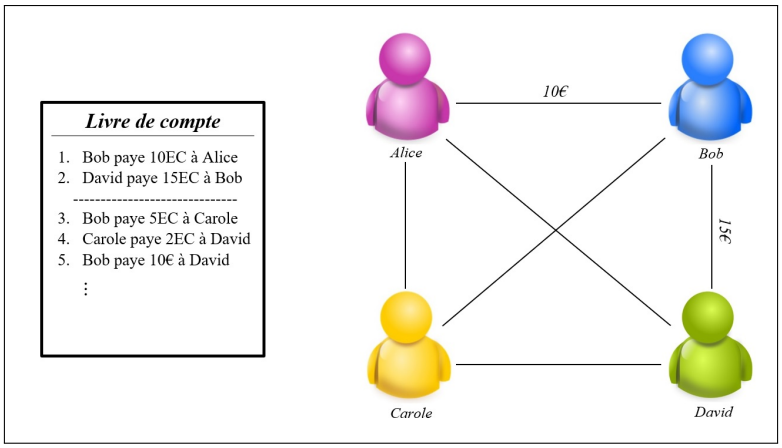

FIGURE 3.7 – Introduction de l'EuroCoin.

Bien sûr, il existe un lien entre les euros et les EuroCoins, tout comme il y en a entre l'euro et le dollar américain. Alice pourrait par exemple décider de donner 10€ à Bob pour que celui-ci inscrive dans le livre de compte "Bob paye 10 EC à Alice" (en supposant un taux de change 1€ = 1 EC) [13]. Néanmoins, le taux de change entre les devises traditionnelles et les cryptomonnaies est relativement indépendant du fonctionnement en lui-même de ces nouvelles espèces. Il dépend davantage des lois du marché et des cours des devises traditionnelles.

Problème n°5 : Ce processus est centralisé

Rappelons ici l'idée principale qui a poussé NAKAMOTO à réfléchir à un nouveau système de paiement : se baser sur un réseau décentralisé, qui ne repose pas sur la confiance accordée à une partie tierce qui gèrerait les transactions. Or, même si nous avons déjà parcouru un bout de chemin depuis notre situation initiale, ce problème subsiste : si nos quatre individus utilisent un livre de compte partagé, qui va gérer celui-ci ? Un site Internet ? Un individu extérieur ? Il nous faut donc trouver une solution.

Celle-ci s'avère toutefois relativement simple a priori : il suffit que chaque personne faisant partie du réseau possède sa propre copie du livre de comptes ! Pour y inscrire une transaction, on exigera alors que l'individu concerné diffuse les informations la concernant afin que tout le monde puisse l'indiquer dans sa copie du livre de compte.

Néanmoins, cette solution amène un problème encore plus grand : comment être sûr que tout le monde reçoive les informations concernant toutes les transactions effectuées ? Comment s'assurer que tout le monde dispose de la même copie du livre de compte, et que cela n'engendre pas des discordes par la suite ? Prenons un simple exemple, si Alice diffuse l'information "Alice paye 10 EC à Bob", mais que Carole ne l'a pas reçue, comment pourrait-elle être sûre que la transaction "Bob paye 10 EC à Carole" est valide ? Sans oublier qu'une telle méthode engendrerait certainement aussi des différences concernant l'ordre des transactions, ce qui amènerait inévitablement encore plus de désordre...

Pour répondre à toutes ces questions et résoudre les problèmes restant, nous allons réellement nous plonger dans la structure et le fonctionnement du réseau Bitcoin.

3.3.2 Fonctionnement général du réseau Bitcoin

Dans la section précédente, nous avons esquissé intuitivement ce qu'était réellement un Bitcoin, puisque nous l'avons "défini" comme une suite de transactions. Néanmoins, nous nous sommes arrêtés sur des problèmes indispensables à résoudre pour obtenir un protocole viable.

Nous avons déjà abordé certaines caractéristiques du Bitcoin, que nous rappelons et résumons ici :

- les transactions doivent être signées pour pouvoir être validées, et n'importe qui peut en vérifier l'authenticité en utilisant la bonne clé publique,
- mis à part le taux de change, le Bitcoin n'a rien à voir avec les devises traditionnelles : c'est une espèce à part entière,
- chaque membre du réseau dispose de sa propre copie du ***ledger*** [14],
- chaque transaction est diffusée à l'ensemble du réseau afin que chacun puisse mettre à jour son ledger, et avoir un historique de toutes les transactions effectuées.

Les illustrations de cette section sont tirées de [33] et [38].

13. Dans le cas du Bitcoin, la cyptomonnaie n'arrive pas dans le livre de compte par une simple conversion cash ↔ BTC. Nous expliquerons plus loin comment les Bitcoins sont progressivement insérés dans le réseau.
14. Pour rappel, *ledger* est l'équivalent anglais de *livre de compte*.

Un point sur l'identité et la vie privée

Au sein du réseau Bitcoin, un individu n'est pas directement lié à son identité dans le monde réel. Les transactions s'effectuent entre des ***adresses*** cryptées, qui découlent directement des clés secrète et publique générées pour chaque individu (qui servent respectivement à signer les transactions et à vérifier l'authenticité de la signature). En effet, dans la pratique, les adresses sont obtenues en calculant le hash de la clé publique par une ou plusieurs fonction(s) de hachage [15]. Dans une transaction, on spécifiera alors l'adresse à laquelle doit être envoyé le montant encodé. Ainsi, une clé publique fait en quelque sorte office d'identité dans le réseau Bitcoin.

L'avantage de ce concept d'identité est que l'anonymat est relativement bien garanti. En effet, d'une part, rien ne relie l'identité réelle d'un individu à sa clé publique, donc encore moins à son adresse. De plus, par propriété des fonctions de hachage, disposer de l'adresse de quelqu'un ne révèle rien sur sa clé publique. D'autre part, cette conception permet à un individu de contrôler plusieurs adresses, et donc plusieurs identités au sein du réseau. En effet, il lui suffit simplement de générer un nouveau couple de clés secrète et publique (et donc une nouvelle adresse). Dans son article, Nakamoto indique d'ailleurs qu'utiliser une nouvelle paire de clés pour chaque nouvelle transaction permet d'éviter d'être identifié par les autres utilisateurs [16].

La blockchain et la notion de consensus partagé

Nous abordons ici l'élément au cœur du fonctionnement du Bitcoin, à savoir la blockchain. Notons tout de même que la description faite ici est assez générale, les détails concernant les blocs et la chaîne en elle-même seront explicités dans la Section 3.3.3.

Avant toute chose, introduisons un mot de vocabulaire qui sera abondamment utilisé dans les pages qui suivent.

Définition 3.3.1. Un ***nœud*** est un élément du réseau Bitcoin, stockant la blockchain et participant à sa construction.
Un nœud est dit ***honnête*** s'il participe positivement au réseau, dans le sens où il ne valide que des transactions valables et n'essaye pas d'attaquer le réseau d'une quelconque façon. A l'inverse, il sera dit ***malveillant*** s'il tente d'intégrer dans la chaîne des transactions frauduleuses ou d'attaquer le réseau.

Remarque 3.3.2. Notons ici que les notions d'adresse et de nœud sont bien distinctes. D'un côté, les adresses servent à transférer des Bitcoins d'une personne à l'autre (ce sont en quelque sorte des tirelires), et de l'autre, les nœuds sont des (groupes d') individus participant à la gestion du réseau (vérification des transactions, construction de la chaîne, etc.). Ainsi, un individu peut posséder une ou plusieurs adresse(s) Bitcoin, sans pour autant constituer un nœud. Par contre, la personne ou le groupe d'individus derrière un nœud gère généralement des adresses Bitcoin également (nous verrons pourquoi par la suite), même si à nouveau, ce n'est pas obligatoire.

Même si nous n'avons pas encore expliqué comment la blockchain du Bitcoin fonctionnait réellement, nous pouvons simplement déjà retenir qu'un nœud est un intervenant actif dans le réseau.

15. Pour le Bitcoin, on retrouve notamment la fonction SHA-256 décrite précédemment. Quelques modifications sont ensuite apportées pour arriver au format classique des adresses Bitcoin.

16. Même si on ne peut directement déterminer l'identité de quelqu'un, les transactions qu'une personne effectue en disent souvent long sur elle. Ainsi, si quelqu'un n'utilise qu'une seule adresse pour toutes ses transactions, les utilisateurs les plus aguerris pourront aisément dresser un profil plutôt précis de cette personne. C'est notamment pour éviter cela qu'il est conseillé d'utiliser plusieurs adresses.

Le rôle que va jouer la blockchain au sein du réseau est en fait celui de ledger partagé. En effet, rappelons que l'idée de base à l'origine du Bitcoin est que tout le monde puisse ajouter et vérifier des transactions, sans passer par une autorité centrale, ce que permet la blockchain. Elle contient l'ensemble de toutes les transactions effectuées, qui sont groupées par blocs. De plus, comme détaillé dans la section dédiée à la blockchain, cette façon de structurer les données permet de détecter facilement si des informations ont été corrompues, ce qui se révèle d'autant plus important lorsque cela relève d'échanges financiers.

Cependant, un des problèmes auquel nous étions confrontés dans la section précédente subsiste : comment s'assurer que tous les utilisateurs disposent de toutes les transactions, et dans le même ordre ? C'est ici que la notion de ***consensus partagé*** fait son apparition. En quelques mots, l'idée est que tous les nœuds se mettent d'accord sur les transactions qui ont été émises et leur ordre.

Évidemment, dans la pratique, ils ne se réunissent pas périodiquement autour d'une table pour en discuter. C'est par l'intermédiaire de la blockchain que ce consensus est atteint : c'est une fois qu'elle est intégrée dans la chaîne de blocs qu'une transaction est considérée comme validée (du moins, à peu de choses près, nous y reviendrons bientôt). La blockchain est donc en quelques sorte le ledger "officiel", l'historique des transactions sur lequel chaque nœud doit se baser pour construire la suite de la chaîne (et donc éviter, par exemple, d'y insérer à nouveau une transaction déjà validée).

Sachant cela, comment fait-on pour arriver à un consensus sur une transaction ou, en d'autres termes, comment ajoute-t-on une transaction à un bloc dans la chaîne ? Comme rappelé précédemment, toutes les transactions sont diffusées au sein du réseau par la personne concernée [17]. Individuellement, chaque nœud prend note des informations qu'il reçoit [18], et construit progressivement une liste de transactions (qui n'ont donc pas encore été validées). Périodiquement (environ toutes les 10 minutes), un nœud propose sa liste comme le prochain bloc de la chaîne (nous verrons bientôt comment ce nœud est choisi parmi l'ensemble du réseau). C'est alors ici qu'intervient la notion de ***consensus implicite*** : l'ensemble des nœuds ne va pas voter ou non pour le bloc, et il n'existe pas d'algorithme permettant d'accepter ou non un bloc. En fait, les nœud montreront qu'ils acceptent le bloc simplement en continuant la construction de la chaîne à la suite du bloc qui vient d'être proposé. A l'inverse, ils le rejetteront en l'ignorant et en repartant d'un autre bloc pour poursuivre la chaîne.

Remarque 3.3.3. Comme les nœuds ont des listes de transactions légèrement différentes, il est possible que l'une d'entre elles ne soit pas reprise dans le bloc créé. Néanmoins, cela ne pose pas de problème : il suffira de l'intégrer dans un des prochains blocs.

Il reste alors une question qu'il est légitime de se poser : que se passe-t-il si deux nœuds A et B proposent (presque) simultanément un bloc ? Dans ce cas, il est fort probable que tous les nœuds du réseau ne reçoivent pas le même bloc : certains vont d'abord voir le bloc A, tandis que d'autres obtiendront le B, ce qui a pour effet de créer deux branches différentes dans la blockchain. On parle alors de ***fourche***. Si cette situation se présente, chaque nœud travaille alors par rapport au bloc reçu en premier, tout en conservant l'autre branche (qu'ils finiront nécessairement par capter). A un moment donné, un nouveau bloc C sera proposé, et celui-ci se greffera nécessairement à une des deux branches A ou B. Le protocole du Bitcoin spécifie alors que c'est toujours la chaîne de blocs la plus longue qui doit être considérée comme "la bonne".

17. A priori, un individu n'est pas obligé de gérer un nœud pour pouvoir effectuer une transaction (cf. Remarque 3.3.2). Néanmoins, si les deux parties concernées par une transaction veulent être sûres que celle-ci a bien été validée, il est préférable qu'elles fassent partie d'un des nœuds du réseau.

18. Le réseau n'étant pas parfait, il est possible que certains nœuds reçoivent des données sur des transactions que d'autres n'ont pas reçues, et inversement. Dès lors, cela peut donner lieu à des listes de transactions personnelles légèrement différentes.

Sachant cela, les nœuds laisseront alors tomber le bloc B et continueront la construction de la blockchain sur la branche A, tel qu'illustré sur la figure ci-dessous. Les blocs délaissés sont qualifiés de ***blocs orphelins***.

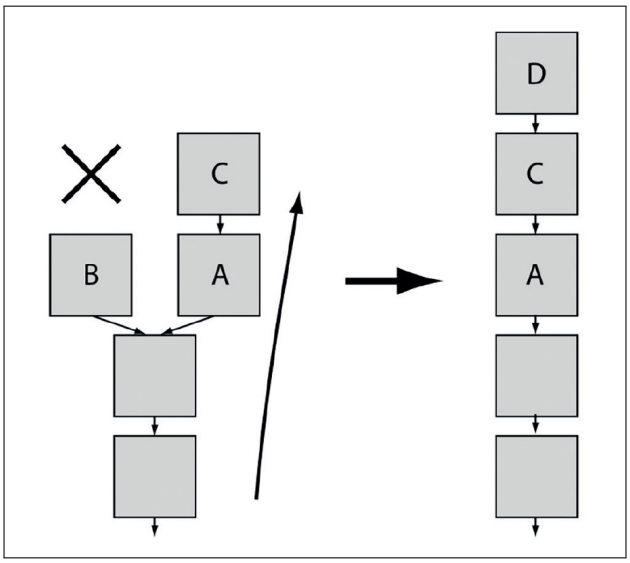

FIGURE 3.8 – Apparition d'une fourche dans la blockchain.

A présent, il semble intéressant de voir si un tel système peut faire face à des attaques ou autres tentatives de fraudes.

Vol de Bitcoin

Ce problème est en fait contré par la cryptographie mise en place derrière le Bitcoin. En effet, pour rappel, chaque transaction doit être signée pour pouvoir être validée. Ainsi, si Alice souhaite dérober des Bitcoins à Bob, elle devra nécessairement créer et diffuser une transaction de la forme "Bob paye n BTC à Alice". Or, seul Bob est capable de signer ce message, et chaque membre du réseau peut vérifier si la signature est authentique ou non grâce à sa clé publique.

Attaque par déni de service

Supposons à présent qu'Alice a une dent contre Bob et qu'incessamment, elle refuse d'inclure les transactions le concernant dans les blocs qu'elle propose. Comme elle n'inclut pas de transaction illégale dans ses blocs, ceux-ci auront de grandes chances d'être acceptés. Néanmoins, Bob ne s'en retrouvera pas lésé pour autant : il lui suffit d'attendre qu'un autre nœud intègre ses transactions dans un futur bloc de la chaîne. Ainsi, a priori, nul ne se retrouvera ignoré du réseau.

Double dépense

C'est un des problèmes majeurs auquel doit faire face un système de paiement décentralisé. Là où le problème du vol est entièrement réglé par la cryptographie cachée derrière le Bitcoin, celui de la double dépense repose complètement sur la notion de consensus évoquée ci-dessus.

Pour illustrer une situation problématique, nous reprendrons l'exemple explicité dans [33], que nous trouvons particulièrement pertinent.

Supposons donc qu'Alice se trouve sur un site Internet géré par Bob, et que ce dernier propose des téléchargements payants, en acceptant le paiement en Bitcoins. Pour rendre la situation réellement concrète, imaginons qu'Alice désire télécharger un logiciel de retouche photos qui coûte 0,01 BTC [19]. Pour ce faire, Alice va créer une transaction partant d'une adresse lui appartenant vers une adresse contrôlée par Bob, et diffuser les informations au sein du réseau, de sorte que les nœuds puissent l'inclure dans un prochain bloc de la chaîne.

C'est maintenant qu'Alice va lancer son attaque. Comme beaucoup d'autres nœuds, Alice va elle aussi construire sa liste de transactions dans le but de proposer le prochain bloc de la chaîne, à la différence près qu'au lieu d'inclure la transaction dans laquelle elle paye Bob, elle va y insérer à la place une opération dans laquelle elle envoie 0,01 BTC à une de ses adresses personnelles, et non plus à Bob. A partir de là, deux cas peuvent se produire :

- soit un bloc contenant la transaction vers Bob a déjà été ajouté à la chaîne, auquel cas Alice reliera le bloc qu'elle vient de créer (et contenant la transaction frauduleuse) à l'avant-dernier bloc, ignorant donc celui qui vient d'être créé,
- soit elle arrive à proposer son bloc simultanément avec un autre nœud.

Dans tous les cas, cela donne lieu à une fourche au sein de la blockchain [20]. Comme explicité précédemment, les nœuds vont alors poursuivre la construction de la chaîne en fonction du premier bloc reçu, tout en gardant l'autre sur le côté.

Dans la figure ci-dessous, le bloc contenant la transaction $C_A \rightarrow B$ est le "bon" bloc, tandis que celui contenant la transaction $C_A \rightarrow A'$ est le bloc frauduleux contenant la tentative de double dépense. Chacune de ces transactions est signée par Alice, et fait référence à une transaction précédente de la chaîne (cf. Section 3.3.4 consacrée aux transactions).

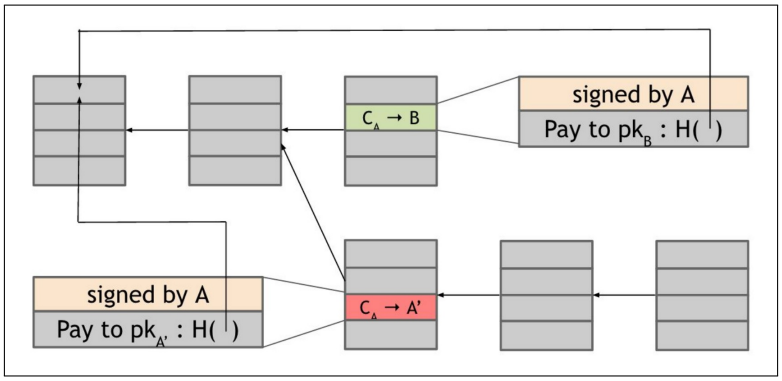

FIGURE 3.9 – Situation de double dépense.

A ce stade, il y a donc autant de chances que le bloc frauduleux se retrouve dans la chaîne que le bloc "correct" (*i.e.* celui contenant la transaction de Alice vers Bob). En effet, d'un point de vue cryptographique, les deux blocs sont valables : nous, ayant connaissance de la situation, pouvons affirmer que la transaction Alice → Alice est frauduleuse, mais ce n'est pas le cas d'un nœud extérieur, pour qui cette transaction est tout autant valable que celle Alice → Bob !

19. Au moment de la rédaction de cet exemple, le cours du Bitcoin est d'environ 1 BTC = 30 000€. Le montant à payer par Alice est donc d'environ 300€.

20. Il faut savoir que les deux blocs (à savoir celui contenant la transaction vers Bob et celui contenant la tentative de double dépense) ne pourront pas se retrouver simultanément dans la chaîne. En effet, lors d'une transaction, on doit également spécifier d'où viennent les fonds que l'on souhaite dépenser (c'est d'ailleurs ce qui est illustré par les grandes flèches brisées sur la Figure 3.9. Nous y reviendrons dans la section dédiée aux transactions. Quoi qu'il en soit, proposer deux transactions utilisant les mêmes fonds sera considéré comme invalide, et les blocs ne peuvent donc pas apparaître tous les deux dans la chaîne.

Un problème de taille se pose alors ici : comment peut-on être sûr que le bloc qui se retrouvera finalement dans la chaîne est bien celui contenant la transaction Alice → Bob, *i.e.* la bonne transaction, et non celui qui contient la double dépense Alice → Alice ? Et bien la réponse est simple : attendre. En effet, comme chaque nœud va travailler sur l'un ou l'autre bloc et que ceux-ci sont tous les deux valables d'un point de vue cryptographique, l'un comme l'autre pourrait se retrouver in fine dans la blockchain. Ainsi, si Bob autorise directement le téléchargement du logiciel, il encourt le risque de ne jamais être payé si c'est le bloc frauduleux qui est finalement accepté.

A la place, Bob peut attendre avant de proposer le téléchargement à Alice : plus d'autres blocs seront construits à la suite de celui contenant la transaction Alice → Bob, plus celle-ci aura des chances de rester dans la chaîne, et donc d'être validée (rappelons la notion de consensus implicite vue précédemment). En effet, on peut montrer que la probabilité que la transaction frauduleuse finisse dans la chaîne décroit exponentiellement en fonction du nombre de blocs construits à la suite du "bon" bloc [21]. En moyenne, on préconise d'attendre 6 confirmations (*i.e.* la création de 6 blocs) pour être sûr qu'une transaction sera effectivement validée. Dans le cas d'Alice et Bob, cela signifie qu'elle devrait attendre environ une heure avant de pouvoir télécharger son logiciel.

Évidemment, tout cela se base sur des probabilités, on n'est donc jamais à 100% sûr qu'une transaction finira bel et bien dans la chaîne. Néanmoins, les auteurs de [33] précisent que le nombre de 6 confirmation constitue un bon équilibre entre le temps à attendre et la la garantie qu'une transaction sera bel et bien validée, *i.e.* finira bien dans un des blocs de la chaîne.

3.3.3 Focus sur les blocs

A présent que nous avons décrit le fonctionnement général du Bitcoin, il semble intéressant de se plonger plus en détails sur la construction de la chaîne : qu'est-ce qui permet de décider quel nœud proposera le prochain bloc ? que contiennent ces blocs ? comment sont-ils construits et reliés aux blocs précédents ? quels sont les outils mathématiques cachés derrière une telle structure ? C'est à ces différentes questions que nous allons répondre dans la présente section.

Gagner de l'espace de stockage

Comme on peut s'y attendre, stocker une telle quantité d'informations (à savoir l'historique de toutes les transactions effectuées) nécessite pas mal d'espace de stockage. D'autant plus que la mémoire nécessaire augmente au fur et à mesure de la construction de la blockchain qui, elle, ne s'arrête pas ! Pour pallier ce problème, le protocole du Bitcoin fait appel à une structure de données particulière : l'***arbre de Merkle***.

Le principe est le suivant. Au sein d'un bloc, on commence tout d'abord par calculer le hash de chaque transaction qui y est répertoriée (rappelons ici qu'un bloc de la chaîne n'est rien d'autre qu'une liste de transactions) par l'intermédiaire d'une certaine fonction de hachage. Ensuite, ces hash sont groupés par paires via l'opération de concaténation [22], et on calcule le hash de chacun de ces couples ainsi formés. On recommence alors le procédé : les hash obtenus sont à nouveau groupés par deux, et les hash des mots ainsi construits sont calculés. On répète le processus jusqu'à l'obtention d'un unique hash, appelé ***racine de Merkle***. Ainsi, cette dernière valeur de hachage contient en quelque sorte les informations de toutes les transactions du bloc. La figure ci-dessous illustre le fonctionnement d'un arbre de Merkle [23].

21. Nakamoto décrit les calculs effectués dans son article [32], nous y reviendrons.

22. Si on est en présence d'un nombre impair de transactions, alors le dernier couple est formé par la concaténation de la dernière transaction avec elle même.

23. Plus précisément, on peut voir sur le schéma que ce sont des pointeurs de hachage qui sont utilisés, comme dans la blockchain.

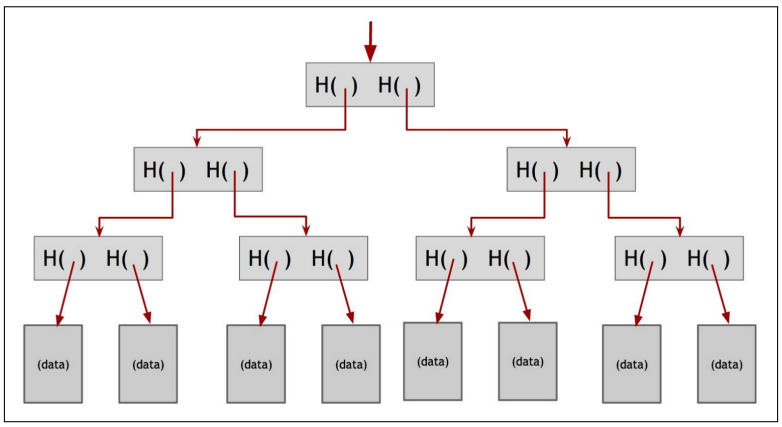

FIGURE 3.10 – Arbre de Merkle.

En particulier, cette façon de structurer les données offre le même atout majeur que la blockchain : la moindre modification des données entraînera des changements radicaux en chaîne de tous les hash suivants, et donc de la racine de Merkle, ce qui permet de détecter toute tentative de falsification.

Cette structuration est donc opérée au sein de chaque bloc de la chaîne. On ne conserve alors que la racine de Merkle du bloc, qui est stockée dans ce qu'on appelle l'***en-tête du bloc*** (en anglais, on parle de *block hash* ou *block header*), qui contient notamment également le hash du bloc précédent. La liste des transactions peut alors quant à elle être compactée pour prendre moins de place lorsque cela s'avère nécessaire. Selon Nakamoto, l'en-tête d'un bloc sans transaction nécessite environ 80 bytes de stockage. En tenant compte de la fréquence de création des blocs (qui, pour rappel, est de 10 minutes), cela correspond à un espace total de 4,2 MB par an, ce qui convient parfaitement à la capacité des mémoires RAM à l'heure actuelle.

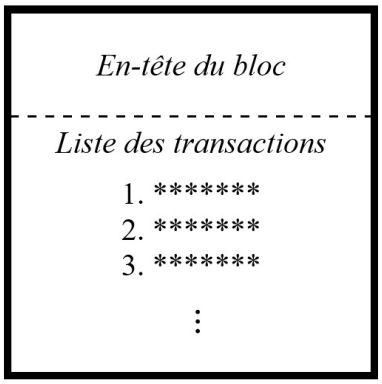

FIGURE 3.11 – Structure d'un bloc.

Preuve de travail

Comme décrit précédemment, nous savons que la blockchain du Bitcoin est construite petit à petit par les nœuds constituant le réseau. Chaque nouveau bloc est proposé par un nœud particulier, à des intervalles réguliers d'environ 10 minutes. Il faut donc trouver un système qui définira quel nœud est éligible pour proposer le prochain bloc.

En choisir un au hasard serait une très mauvaise idée. En effet, même si l'on suppose que le réseau compte davantage de nœuds honnêtes que malveillants, il subsisterait toujours une probabilité qu'un bloc soit créé par un de ces derniers...

L'idée est donc de se fier à une autre variable que l'honnêteté : la puissance de calcul. Autrement dit, plus un nœud dispose d'une grande puissance de calcul, plus la probabilité qu'il propose le prochain bloc sera grande. On peut voir cela comme un genre de compétition entre tous les nœuds du réseau : chacun essaye de déterminer le prochain bloc de la chaîne, en fonction des ressources informatiques dont il dispose. Procéder de cette façon, *i.e.* accorder la confiance aux nœuds disposant de la plus grande puissance de calcul, est appelé une vérification par ***preuve de travail*** (*proof of work* en anglais).

Dans la pratique, cette preuve de travail consiste en la résolution d'un problème impliquant des fonctions de hachage (on parle de *hash puzzles* en anglais). Ceux-ci sont généralement assez simples à première vue (dans le sens où l'énoncé n'est pas compliqué à comprendre), mais extrêmement difficiles à résoudre en pratique si on ne dispose pas d'une grande puissance de calcul.

Dans le cas du Bitcoin, le problème de hachage à résoudre est le suivant. Tout d'abord, rappelons que l'en-tête de chaque bloc doit contenir d'une part le hash du bloc précédent, et d'autre part la racine de Merkle associée à la liste de transactions du bloc à intégrer dans la chaîne. En plus de cela, elle doit également contenir trois autres informations :

- le moment où il a été créé, que l'on appelle ***horodatage*** (autrement dit, un bloc spécifie la date et l'heure à laquelle il est construit),
- un paramètre de difficulté (nous y reviendrons),
- la version de ***Bitcoin Core***[24] utilisée.

Le problème consiste alors à trouver une certaine valeur entière, appelée ***nonce***[25], telle que, si on concatène cette valeur avec tous les éléments cités ci-dessus, et que l'on applique une fonction de hachage au mot ainsi construit, la valeur de sortie obtenue appartient à un certain ensemble-cible de l'espace d'arrivée.

Éclaircissons un peu les choses. Si on considère une fonction de hachage $H : A^* \to B^m$, alors résoudre le problème énoncé ci-dessus revient à déterminer une valeur n (le nonce) telle que

$$H(v \cdot h \cdot R(tx_1, \ldots, tx_k) \cdot T \cdot d \cdot n) \in X \subseteq B^m,$$

avec

- v la version de Bitcoin Core utilisée,
- h le hash du bloc précédent,
- $R(tx_1, \ldots, tx_k)$ la racine de Merkle associée aux transactions $tx_1, \ldots, tx_k$ inscrites dans le bloc,
- T l'horodatage,
- d le paramètre de difficulté,
- n le nonce.

La fonction de hachage utilisée pour le Bitcoin est la fonction SHA-256 présentée dans la Section 3.2.1. Pour être plus précis, elle est utilisée deux fois successivement, *i.e.* on utilise la fonction

$$H = \text{SHA-256} \circ \text{SHA-256}.$$

Autrement dit, la valeur de sortie est un mot de 256 bits, que l'on peut voir comme un entier compris entre 0 et $2^{256} - 1$ représenté en base 2. Sachant cela, ce qui va jouer le rôle d'ensemble-cible pour le Bitcoin est en réalité une valeur maximale qu'il ne faut pas dépasser.

24. Sans entrer dans les détails, ***Bitcoin Core*** est le logiciel libre principal utilisé par les nœuds pour la résolution du problème de hachage décrit ici.

25. Cette valeur est appelée ainsi en raison du fait qu'elle n'est utilisée qu'une seule fois (le mot vient de l'anglais *once*).

Ainsi, si on utilise les mêmes notations que précédemment, l'objectif est de trouver un nonce n tel que

$$H(v \cdot h \cdot R(tx_1, \ldots, tx_k) \cdot T \cdot d \cdot n) < x,$$

où x est un entier compris entre 0 et $2^{256} - 1$. On peut également interpréter cela en termes de mots. En effet, imposer une borne supérieure au hash revient à demander que ce dernier commence par un certain nombre de 0.

Toute la difficulté réside donc dans la résolution de ce problème. Par propriété des fonctions de hachage, il est impossible de déterminer la valeur d'un nonce qui pourrait convenir directement à partir de la valeur de sortie (où ici, l'ensemble auquel elle doit appartenir) ainsi que des contraintes imposées (à savoir les autres informations présentes dans l'en-tête du bloc) : c'est la fameuse propriété de puzzle friendliness. Ainsi, la seule et meilleure stratégie possible pour trouver la solution est d'essayer une par une toutes les possibilités. A l'inverse, il est très facile de vérifier qu'une solution proposée est la bonne, puisqu'il suffit de calculer le hash de l'en-tête du bloc ainsi obtenue. L'ensemble des nœuds peut donc facilement vérifier la validité du bloc proposé.

C'est la résolution de ce problème de hachage que l'on appelle communément ***minage***. Les nœuds participant à l'exécution de cette tâche sont alors qualifiés de ***mineurs***.

On comprend donc aisément pourquoi la puissance de calcul est un élément déterminant dans la création des blocs : plus on en dispose, plus vite on peut passer en revue toutes les possibilités et donc proposer le bloc au réseau. Néanmoins, cela ne signifie pas qu'un nœud disposant de ressources informatiques moindres ne trouvera jamais de bloc. Leur fréquence d'apparition sera simplement plus faible.

Comme déjà mentionné précédemment, un bloc est créé en moyenne toutes les 10 minutes. Pour maintenir ce rythme, la difficulté [26] du problème est régulièrement ajustée (tous les 2 016 blocs pour être précis) : si la création des blocs est trop rapide, alors la difficulté augmente, et inversement. Cela permet entre autre de s'adapter constamment aux progrès technologiques ainsi qu'à l'évolution de la puissance de calcul du réseau.

Petit point écologique

Comme nous venons de le voir, construire un bloc de la chaîne demande beaucoup de ressources informatiques : des ordinateurs puissants, pouvant effectuer et vérifier des calculs très rapidement. Bien sûr, tout cela nécessite de l'énergie (parfois en très grande quantité, en fonction de l'ampleur de l'infrastructure), ce qui n'est pas toujours bien vu à l'heure actuelle !

Tout d'abord, notons que l'empreinte écologique d'un mineur dépend évidemment de son ampleur : un particulier indépendant qui mine du Bitcoin de son côté aura beaucoup moins d'impact sur l'environnement qu'une entreprise qui en fait son business. En ce qui concerne l'énergie utilisée par la gestion du réseau Bitcoin (et en particulier par le minage), on retrouve d'une part l'électricité nécessaire pour alimenter les ordinateurs effectuant les calculs (qui sont parfois regroupés par centaines dans des hangars, que l'on appelle communément ***fermes de minage***), et d'autre part l'énergie nécessaire au refroidissement de ces machines, essentiel à leur bon fonctionnement.

Bien sûr, des solutions sont possibles. Certains mineurs utilisent par exemple la chaleur dégagée par les machines pour chauffer leur maison. D'autres encore, dans certains pays, rachètent le surplus d'énergie produit à un moindre coût pour ne pas la gaspiller.

26. La difficulté de résolution du problème est ici liée à la borne supérieure x choisie : plus celle-ci est grande, plus le problème est facile à résoudre, et vice versa. Cette difficulté est matérialisée par le paramètre d introduit plus haut.

Quoi qu'il en soit, il est difficile de déterminer exactement si le Bitcoin est néfaste ou non d'un point de vue écologique. Comme le reprennent d'ailleurs très justement les auteurs de [48], *"le Bitcoin consommera toujours trop d'énergie pour ceux qui le jugent inutile"*. Néanmoins, est-ce réellement honnête de blâmer la consommation d'énergie du Bitcoin et la pollution qui en découle, alors que paradoxalement, d'autres efforts en termes de diminution des émissions de gaz à effet de serre sont loin d'être atteints ? Nous entrons là dans un débat certes très intéressant et qui vaudrait sans aucun doute la peine d'être abordé, mais qui relève davantage de l'éthique que des mathématiques. Pour le lecteur intéressé, l'article [48] (au titre accrocheur mais aux propos nuancés) regorge de références qui valent la peine d'être parcourues. NARAYANAN ET. AL [33] proposent également une analyse de l'énergie consommée par le minage et ses implications (Section 5.3 de l'ouvrage), mais celle-ci est plus ancienne.

Récompenses de minage

Outre le coût écologique qu'il engendre, le minage demande un certain investissement, et surtout un certain prix ! Afin de récompenser les nœuds participant à la construction de la blockchain, le protocole du Bitcoin permet donc aux mineurs d'intégrer aux blocs qu'ils créent une transaction supplémentaire. Celle-ci autorise le mineur à envoyer un certain nombre de Bitcoins, défini à l'avance, à l'adresse qu'il souhaite (bien souvent, à une adresse qui lui appartient). Cette compensation est appelée ***récompense de minage*** ou ***de bloc*** (*block reward* en anglais).

Le montant en Bitcoins de cette récompense est définie selon le schéma suivant. Au début du Bitcoin, elle était de 50 BTC par bloc créé. Ensuite, la règle veut que ce montant soit divisé par 2 tous les 210 000 blocs. Considérant le fait qu'un nouveau bloc voit le jour environ toutes les 10 minutes, cela correspond à une période de plus ou moins 4 ans. Au moment de la rédaction de ce mémoire, la récompense est de 6,25 BTC [27]. Le montant de départ a donc déjà été divisé par 8 !

Cette récompense de minage est en réalité ce qui permet d'injecter des Bitcoins dans le réseau. En effet, comme mentionné au début de cette section sur la cryptomonnaie, les Bitcoins ne sont pas générés par de l'argent réel : ils ne sont pas issus des conversions devises traditionnelles/cryptomonnaies. Ils sont directement créés à partir de ces récompenses. De plus, le fait que le montant de cette compensation diminue périodiquement implique qu'ultimement, seul un nombre fini de Bitcoins seront ainsi générés. En effet, même si un Bitcoin peut être fractionné (par exemple, on peut parler de 0,005 BTC), il existe une limite à ce découpage : la plus petite valeur possible est de 10^{-8} BTC, et est appelée un ***satoshi***. Dès lors, après 32 divisions par 2 du montant initial de la récompense de minage [28], soit aux alentours de l'an 2140, celle-ci atteindra une valeur inférieure à un satoshi, ce qui implique d'une part que les mineurs ne seront plus rémunérés pour leur travail, et d'autre part que le nombre maximal de Bitcoins présents dans le réseau sera atteint. En particulier, ce nombre peut être calculé :

$$\begin{aligned} 210000 \cdot \sum_{i=0}^{32} \frac{50}{2^i} &= 10,5 \cdot 10^6 \cdot \sum_{i=0}^{32} \left(\frac{1}{2}\right)^i \\ &= 10,5 \cdot 10^6 \cdot \frac{1-\left(\frac{1}{2}\right)^{33}}{1-\frac{1}{2}} = 10,5 \cdot 10^6 \cdot \frac{1-\frac{1}{2^{33}}}{\frac{1}{2}} = 10,5 \cdot 10^6 \cdot \left(2 - \frac{1}{2^{32}}\right) \\ &\simeq 21 \cdot 10^6. \end{aligned}$$

27. Avec le cours actuel (1 BTC $\simeq$ 30 000€), cela correspond à environ 187 500€. A titre de comparaison, un mineur des premières années du Bitcoin qui aurait conservé ses 50 BTC de récompense serait millionnaire, étant donné qu'il disposerait virtuellement de 1,5 millions d'euros !

28. La 33^{e} division donne une récompense d'environ $5,82 \cdot 10^{-9}$ BTC $< 10^{-8}$ BTC.

Ainsi, il n'y aura a priori pas plus de 21 millions de Bitcoins en circulation dans le réseau. Néanmoins, avoir un nombre limité de Bitcoin sur le marché est plutôt une bonne chose : si on pouvait en générer à l'infini, quelle valeur monétaire auraient-ils ?

Un problème se pose tout de même : qu'est-ce qui motivera les mineurs à continuer leur travail lorsque la récompense de minage n'existera plus ? Pour répondre à cette question, un autre type de compensation est mis en place dans le protocole du Bitcoin : les frais de transaction. En effet, et nous y reviendrons dans la section qui leur est consacrée, un individu effectuant une transaction peut décider d'augmenter un peu le montant de celle-ci afin d'y inclure des frais de transaction. Ceux-ci seront alors récupérés par le mineur qui construira le bloc dans lequel sera intégrée la transaction. Ainsi, même si la récompense de minage disparaît, les mineurs continueront d'être rémunérés d'une manière ou d'une autre !

Nœuds malveillants

Dans les paragraphes précédents, nous avons implicitement fait l'hypothèse que la majorité des nœuds du réseau étaient honnêtes. Néanmoins, il serait intéressant de se questionner sur ce qu'il se passerait si une attaque était lancée par un nœud malveillant.

D'une part, il serait impossible pour un quelconque attaquant de subvertir la cryptographie mise en place derrière le Bitcoin impunément. En effet, de par leur structure intrinsèque, la blockchain tout comme l'arbre de Merkle permettent de détecter presque instantanément toute tentative de fraude : la modification ne serait-ce que d'un caractère d'une donnée du bloc entraînera un hash totalement différent de celui obtenu. Quelqu'un qui souhaiterait donc falsifier des transactions se verrait contraint de refaire tout le travail de calcul nécessaire au minage d'un bloc pour ré-obtenir un hash de la bonne forme. Sans oublier qu'il serait également contraint de modifier les hash de tous les blocs suivants. Autant le dire, le système mis en place rend impossible une telle attaque dans la pratique.

D'autre part, il est très peu probable qu'un attaquant arrive à saboter le principe du consensus implicite mis en place par la blockchain. Fournissons quelques explications supplémentaires à ce propos. Supposons qu'un attaquant intègre un bloc frauduleux dans la chaîne, avec pour but de le faire valider (autrement dit, de l'intégrer dans la chaîne à long terme) [29]. Cela mènera donc inévitablement à la création d'une fourche à un moment donné, et le but du nœud malveillant sera alors de faire progresser la branche contenant son bloc plus vite que la branche "honnête". Dans son article [32], Nakamoto calcule la probabilité que l'attaquant réussisse. Il matérialise la course entre les deux branches comme une marche aléatoire binomiale : un succès correspond à étendre la chaîne honnête d'un bloc, et donc creuser l'écart d'une unité, et une défaite à l'extension de la chaîne frauduleuse, réduisant l'écart d'une unité. On peut alors faire un parallèle avec le problème de la ruine [30], et donc utiliser les résultats relatifs à ce genre de situation [31]. Ainsi, si

- p désigne la probabilité qu'un nœud honnête trouve le prochain bloc,
- q désigne la probabilité qu'un nœud malveillant trouve le prochain bloc,

29. Notons qu'une telle attaque ne permettrait pas au nœud malveillant de créer des Bitcoins à sa guise ou d'en voler, puisque les nœuds honnêtes n'accepteront jamais des transactions impliquant des Bitcoins obtenus "illégalement". On supposera donc que l'attaquant souhaite récupérer des Bitcoins dépensés préalablement (cf. attaque de la double dépense, Section 3.3.2).

30. Deux joueurs A et B disposent chacun d'une certaine somme d'argent de départ, disons a euros pour A et b euros pour B. A chaque tour, un joueur prend 1€ à l'autre, A avec une probabilité p et B avec une probabilité $q = 1 - p$. Le jeu se termine alors lorsqu'un des deux joueurs est ruiné.

31. Ces résultats ne seront pas explicités ici, mais le lecteur intéressé peut se référer à [2] et [37] pour plus d'informations.

avec $q = 1 - p$, alors la probabilité que l'attaquant rattrape la chaîne honnête s'il a n blocs de retard est donnée par

$$P(n) = \begin{cases} 1 & \text{si } p \leq q \\ \left(\frac{q}{p}\right)^n & \text{si } p > q \end{cases} .$$

En particulier, en supposant que $p > q$, ce qui est le cas si la majorité des nœuds du réseau sont honnêtes, la probabilité ci-dessus décroît exponentiellement avec le nombre de blocs à rattraper. Ainsi, il y a très peu de chances qu'un attaquant réussisse son coup, celles-ci diminuant d'autant plus qu'il est à la traîne par rapport à la chaîne de blocs correcte.

Une situation que beaucoup craignent alors est celle d'un attaquant qui disposerait de plus de la moitié de la puissance de calcul du réseau, et qui tenterait de l'attaquer. Cette attaque est connue sous le nom d'***attaque des 51 pourcents***. Comme nous l'avons déjà mentionné, s'en prendre à la cryptographie derrière le Bitcoin est quasiment (pour ne pas dire complètement) impossible. A nouveau, c'est la notion de consensus implicite qui serait ici menacée. Cependant, le plus gros risque dans ce cas est que la connaissance d'une telle attaque au sein du réseau entraînerait une grande perte de confiance dans le Bitcoin, et sans aucun doute un effondrement de la valeur monétaire de ce dernier. Néanmoins, tout a été mis en place pour éviter ce genre de situation. D'une part, la puissance de calcul qui serait nécessaire pour monopoliser 51% du réseau impliquerait des frais incommensurables, à tel point que lancer une telle attaque ne serait que pure folie. D'autre part, les récompenses mises en place pour les mineurs ont aussi pour but d'encourager les nœuds à se comporter honnêtement vis-à-vis du réseau. De plus, comme le soulignent très justement les auteurs de [33], il y a fort à parier qu'une telle attaque serait rapidement repérée par les développeurs du réseau, et que ceux-ci réagiraient rapidement en modifiant le protocole du Bitcoin afin de la contrer.

3.3.4 Focus sur les transactions

Pour clôturer notre voyage dans le réseau Bitcoin, regardons d'un peu plus près ce qui constitue une transaction. Nous savons déjà que, pour pouvoir être validée, une transaction doit être correctement signée. Il reste néanmoins des problèmes énoncés dans notre situation de départ que nous n'avons pas encore résolus. Parmi eux, par exemple, comment être sûr qu'Alice puisse envoyer 0,1 BTC à Bob ? Qu'est-ce qui nous assure qu'elle dispose bien des fonds nécessaires pour effectuer cette transaction ? C'est à ce genre de problèmes que nous allons nous intéresser dans cette section.

Valeurs d'entrée et de sortie

Chaque transaction est associée à un certain nombre d'entrées et de sorties (qui sont donc indexées afin de pouvoir y faire appel par la suite). De façon imagée, on peut en quelque sorte voir les valeurs d'entrée comme étant un nombre de Bitcoins que l'on mettrait dans une boîte, et les valeurs de sortie le nombre de Bitcoins que l'on retirerait de cette boîte. On impose alors que le montant total de sortie soit plus petit ou égal au montant total entré [32]. L'avantage de ce système est qu'une personne, Alice par exemple, peut décider de puiser des Bitcoins provenant de plusieurs adresses lui appartenant, pour les envoyer à potentiellement plusieurs adresses différentes. Une autre fonctionnalité rendue possible grâce à ce système est celle des paiments collectifs : si Alice et Carole souhaitent envoyer des Bitcoins à Bob pour son anniversaire, elles peuvent créer une transaction reprenant leurs adresses respectives en entrée, et celle de Bob en sortie. Notons qu'une telle transaction devra alors comporter deux signatures : celles d'Alice et Carole.

32. Si le montant de sortie est strictement plus petit que le montant d'entrée, alors la différence constitue les frais de transaction destinés au mineur dont nous avons parlé dans la section consacrée aux blocs.

Notons ici une particularité des transactions de Bitcoins. Si Alice a reçu 10 BTC à l'une de ses adresses, et qu'elle désire en envoyer 6 à Bob, elle va créer une transaction spécifiant une entrée (faisant appel aux 10 BTC, nous allons y revenir) et deux sorties : une vers l'adresse de Bob, spécifiant le montant de 6 BTC, et l'autre vers une de ses adresses personnelles, correspondant au reste de 4 BTC. Ainsi, lors d'une transaction, le montant total des Bitcoins disponibles à une adresse spécifiée en entrée est investi. Si le montant à transférer est inférieur au montant engagé en entrée, alors une sortie supplémentaire vers une des adresses de l'expéditeur est créée afin d'y retourner les fonds excédentaires.

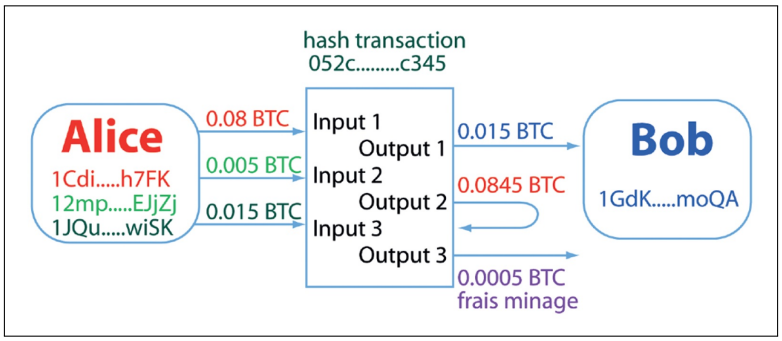

FIGURE 3.12 – Exemple d'une transaction constituée de différentes entrées et sorties.

A présent, il semble intéressant de préciser exactement ce que nous entendons par "entrées" et "sorties" d'une transaction :

- une ***entrée*** spécifie plusieurs éléments :
  1. la transaction passée tx (ou, plus précisément, le hash de cette transaction) dont proviennent les fonds qui vont être envoyés (cela permet à un nœud extérieur à la transaction de vérifier qu'un individu dispose bien des fonds nécessaires pour effectuer sa propre transaction),
  2. l'index n de la sortie de la transaction tx d'où provient le montant que l'on souhaite utiliser,
  3. une signature attestant l'autorisation d'utiliser les fonds réclamés,
- une ***sortie*** contient seulement deux éléments : un montant en Bitcoins ainsi qu'une adresse de destination, accompagnée d'une autre commande, appelée ***script***.

Scripts

Nous ne rentrerons pas ici dans les détails concernant le script du Bitcoin. Le but des quelques paragraphes suivants est essentiellement d'expliquer dans les grandes lignes ce qu'est un script, et à quoi il sert. Le lecteur intéressée pourra trouver des informations complémentaires concernant son fonctionnement dans le troisième chapitre de [33].

Commençons par le commencement : qu'est-ce qu'un ***script*** ? Dans le cadre du Bitcoin, on peut voir cela comme un petit programme qui est intégré à une transaction, et qui empêche n'importe qui d'utiliser les fonds en jeu, sauf à l'individu propriétaire de l'adresse à laquelle ils sont envoyés. On peut donc voir le script d'une transaction comme une sécurité supplémentaire mise en place pour éviter toute utilisation frauduleuse de Bitcoins.

Ainsi, si Alice désire envoyer des Bitcoins à Bob, elle va devoir spécifier d'une part son adresse et le montant de la transaction, et d'autre part un script qui fera en sorte que Bob, et seulement lui, puisse débloquer les fonds de la transaction. Dans la pratique, Alice crée un ***script de blocage*** dans la transaction, qui devra être concaténé avec un ***script de déblocage*** que seul Bob peut générer. En effet, celui-ci est construit par l'intermédiaire de sa clé secrète (dont seul Bob a connaissance), qui est liée à l'adresse à laquelle il a reçu le paiement.

La figure ci-dessous reprend en résumé toutes les notions importantes intervenant dans une transaction de Bitcoins.

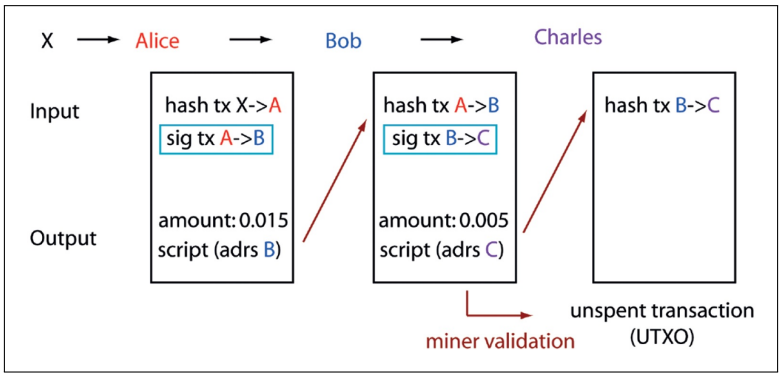

FIGURE 3.13 – Éléments constitutifs des transactions.

Dans cette situation, Alice a transféré 0,015 BTC à Bob. Pour ce faire, elle a dû spécifier en entrée la transaction qui atteste qu'elle dispose bien de ce montant (il s'agit de la transaction $X \to A$) et signer sa transaction vers Bob. En sortie, elle a alors ajouté l'adresse de Bob, ainsi qu'un script de blocage que seul Bob pourra débloquer grâce à sa clé secrète.

A son tour, Bob désire envoyer 0,005 BTC à Charles. Il effectue donc une procédure similaire à Alice : après avoir débloqué le script avec sa clé secrète, il renseigne la transaction $A \to B$ qui lui permet d'utiliser ce montant, il signe sa transaction et précise l'adresse de Charles, en incluant un script de blocage. Il diffuse alors les informations de cette transaction au sein du réseau afin de l'inclure dans la blockchain.

Avant de l'inclure dans un futur bloc, un mineur recevant cette transaction examinera plusieurs éléments. Entre autres, il vérifiera l'authenticité de la signature de Bob via sa clé publique, et il consultera dans la blockchain les informations de la transaction $A \to B$ pour s'assurer que Bob a effectivement le droit d'utiliser les fonds qu'il souhaite transférer. Une fois ces vérifications effectuées, le mineur ajoutera la transaction à sa liste, dans le but de l'introduire dans la chaîne par la suite. Tant qu'elle n'aura pas été utilisée par Charles, celle-ci restera marquée comme étant non dépensée.

On comprend ici à nouveau toute l'importance de conserver sa clé secrète à l'abri : comme elle permet de débloquer les scripts, la perdre ou se la faire voler reviendrait à ne plus pouvoir utiliser les Bitcoins stockés à l'adresse correspondante (et, dans le pire des cas, à se les faire dérober).

3.4 Et les courbes elliptiques dans tout ça ?

A présent que nous avons passé en revue le fonctionnement du réseau Bitcoin, il demeure une question à laquelle nous n'avons pas répondu, et non des moindres : quel est le rapport entre les courbes elliptiques et le Bitcoin ?

En fait, elles interviennent (notamment une en particulier, nous y reviendrons) à un moment que l'on pourrait qualifier de décisif dans le protocole : les signatures digitales. En effet, il s'agit du premier élément indispensable que nous avions pointé dans notre *"wishlist"* pour le Bitcoin : afin d'assurer la validité et l'authenticité d'une transaction, il faut que celle-ci soit signée par l'expéditeur. Les courbes elliptiques interviennent alors dans ce processus de signature. Elles permettent de générer les couples de clés secrète et publique des utilisateurs, et leurs propriétés permettent la vérification de l'authenticité d'une signature par un nœud extérieur. Tout cela repose sur un algorithme : l'***Elliptic Curve Digital Signature Algorithm***, ou ***ECDSA***.

Le but de la présente section sera donc d'expliquer le fonctionnement de cet algorithme de façon générale. Nous présenterons ensuite les paramètres utilisés dans le cadre de son application au Bitcoin.

3.4.1 Elliptic Curve Digital Signature Algorithm (ECDSA)

L'***Elliptic Curve Digital Signature Algorithm*** (qui sera simplement désigné par ECDSA dans la suite de ce travail) est l'analogue du ***Digital Signature Algorithm*** (généralement désigné par l'acronyme DSA). Avant d'aborder le fonctionnement de l'ECDSA, il semble intéressant de décrire celui du DSA, afin de pouvoir les comparer (tout comme nous l'avons fait dans le cadre du cryptosystème de Massey-Omura, cf. Section 2.4).

Fonctionnement du DSA

Cet algorithme, mis au point par le *National Institute of Standards and Technology* (NIST) aux États-Unis, a vu le jour en août 1991. Il repose sur le caractère supposé insoluble du problème du logarithme discret. A l'heure actuelle, il est de plus en plus délaissé au profit d'algorithmes plus efficaces et sécurisés tels que l'ECDSA, que nous détaillerons par après.

Paramètres de domaine. Pour cet algorithme comme pour l'ECDSA, la première chose à faire est de déterminer des ***paramètres de domaine***, qui sont définis comme étant des "paramètres utilisés dans un algorithme cryptographique et communs à un ensemble d'utilisateurs" par le NIST. Dans le cadre du DSA, ces paramètres sont déterminés comme suit. On choisit :

1. une fonction de hachage[33] H,
2. des nombres premiers p et q, respectivement d'une longueur[34] de L bits et de N bits, et tels que $q \mid p-1$,
3. un élément $h \in \mathbb{Z}_p^*$ (cfr. Définition 1.1.10), et on calcule $g \equiv h^{\frac{p-1}{q}} \bmod p$; si $g \equiv 1 \bmod p$, on sélectionne un autre h et on réitère l'opération.

Les paramètres de domaine sont alors les entiers p, q et g.

33. Initialement, c'est la fonction SHA-1 qui était utilisée. Cette fonction prend en entrée des mots d'une longueur maximale de 2^{64} bits et retourne des mots d'une longueur de 160 bits. Néanmoins, de relativement récentes recherches (2005) ont mené à la découverte de méthodes permettant de trouver des collisions pour cette fonction (or, rappelons-le, une fonction de hachage doit être résistante aux collisions, cf. Section 3.2.1), dont on a par la suite montré la faisabilité. Le lecteur intéressé peut consulter [56], qui décrit le fonctionnement de la toute première méthode trouvée. La fonction SHA-1 a alors été progressivement délaissée au profit d'autres fonctions de hachage plus sûres.

34. Celles-ci dépendent notamment des standards choisis. Le lecteur intéressé peut consulter [34] pour plus d'informations.

Générer une paire de clés. A présent, nous allons décrire comment un utilisateur du réseau (disposant donc des paramètres de domaine) peut générer ses clés secrète et publique, qui serviront d'une part à signer ses messages, et d'autre part aux autres utilisateurs de vérifier ces signatures. La procédure est la suivante :

1. sélectionner un entier x tel que $1 \leq x \leq q-1$,
2. calculer $y \equiv g^x \bmod p$.

Les entiers x et y constituent alors respectivement les clés secrète et publique de l'utilisateur qui les a générées.

Signer un message m. Supposons qu'Alice a généré une paire de clés (x, y). Voici les étapes qu'elle doit suivre afin de signer un message m qu'elle voudrait transmettre à Bob :

1. choisir au hasard un entier k tel que $1 \leq k \leq q-1$,
2. calculer $r \equiv \left(g^k \bmod p\right) \bmod q$; si $r \equiv 0 \bmod q$, recommencer à l'étape 1,
3. calculer $H(m)$, le hash du message m par la fonction H, et convertir [35] le mot obtenu en un entier e,
4. calculer $s \equiv k^{-1}(e + xr) \bmod q$; si $s \equiv 0 \bmod q$, recommencer à l'étape 1.

Le couple d'entiers (r, s) ainsi trouvé constitue alors la signature d'Alice pour le message m.

Vérifier une signature. A présent, imaginons que Bob veuille vérifier que le message m qu'il a reçu provient bien d'Alice, et n'a pas été envoyé ou falsifié par quelqu'un d'autre (Oscar n'est en effet jamais bien loin). Pour ce faire, il va vérifier l'authenticité de la signature (r, s) d'Alice au moyen de sa clé publique y, qu'elle a partagée au préalable. Bob peut alors procéder comme suit :

1. vérifier que $1 \leq r, s \leq q-1$,
2. calculer $H(m)$ et le convertir en l'entier e,
3. calculer $w \equiv s^{-1} \bmod q$,
4. calculer $u_1 \equiv ew \bmod q$ et $u_2 \equiv rw \bmod q$,
5. calculer $v \equiv (g^{u_1} y^{u_2} \bmod p) \bmod q$.

La proposition suivante permet alors de conclure.

Proposition 3.4.1. *En utilisant les mêmes notations que précédemment, une signature (r, s) est valide si et seulement si $v \equiv r \bmod q$.*

Démonstration. Si la signature (r, s) est valide, alors par définition

$$s \equiv k^{-1}(e + xr) \bmod q.$$

Dès lors,

$$\begin{aligned} k &\equiv (e + xr)\, s^{-1} \bmod q \\ &\equiv (e + xr)\, w \bmod q \\ &\equiv ew + xrw \bmod q \\ &\equiv u_1 + xu_2 \bmod q, \end{aligned}$$

par définition des entiers w, u_1 et u_2.

35. Le mot de sortie peut être converti en son écriture en base 2 (c'est d'ailleurs sous cette forme qu'un ordinateur le "lit"), puis en l'entier correspondant.

Or, g est d'ordre q. En effet, par définition, on a

$$g \equiv h^{\frac{p-1}{q}} \bmod p \Rightarrow g^q \equiv h^{p-1} \bmod p.$$

Comme p est premier et que h n'est pas divisible par p, le Petit Théorème de Fermet (Théorème 2.2.10) implique $h^{p-1} \equiv 1 \bmod p$, et donc $g^q \equiv 1 \bmod p$.

Par conséquent, vu ce qui précède, il vient

$$\begin{aligned} g^k &\equiv g^{u_1+xu_2} \bmod p \\ &\equiv g^{u_1} \left(g^x\right)^{u_2} \bmod p \\ &\equiv g^{u_1} y^{u_2} \bmod p. \end{aligned}$$

Il suffit alors de remarquer que

$$\begin{aligned} r &\equiv \left(g^k \bmod p\right) \bmod q \\ &\equiv \left(g^{u_1} y^{u_2} \bmod p\right) \bmod q \\ &\equiv v \bmod q. \end{aligned}$$

Pour l'autre implication, il suffit alors de considérer le raisonnement effectué ci-dessus dans l'autre sens. □

Fonctionnement de l'ECDSA

L'ECDSA, analogue du DSA utilisant les propriétés des courbes elliptiques, a été proposé en 1992 par Scott VANSTONE (1947 – 2014). Comme on peut s'y attendre, cet algorithme se base sur le problème du logarithme discret des courbes elliptiques, dont les avantages ont été explicités dans la Section 2.3.

Paramètres de domaine. Les paramètres de départ nécessaires à la génération de signatures dans le cadre de l'ECDSA sont les suivants :

1. une fonction de hachage H,
2. un champ $\mathbb{F}_q$ de caractéristique p, avec $q = p$ ou $q = 2^m$ ($m \in \mathbb{N}_0$),
3. une courbe elliptique E définie sur $\mathbb{F}_q$,
4. un ***point de base*** G de E, distinct du point à l'infini $\mathscr{O}$, et d'ordre premier n (*i.e.* $nG = \mathscr{O}$).

Il ne sera pas décrit ici précisément comment le champ et la courbe elliptique sont déterminés en pratique. On peut néanmoins retenir que l'on impose certaines contraintes au champ ainsi qu'aux coefficients de l'équation de Weierstrass de la courbe (afin d'éviter d'utiliser des valeurs trop "simples", dans le sens où elles pourraient facilement permettre des attaques bien connues). On a alors recours à des algorithmes permettant de générer une courbe elliptique relativement compliquée sur le champ déterminé. D'autres algorithmes sont alors utilisés pour vérifier la validité des paramètres générés. Le lecteur qui souhaite en connaître davantage sur le sujet peut consulter [22] et [34].

Générer une paire de clés. A présent que nous disposons des paramètres de domaine, un utilisateur peut générer son couple de clés secrète et publique. Pour ce faire, il faudra :

1. sélectionner un entier x tel que $1 \leq x \leq n-1$,
2. calculer $Q = xG$; si $Q = \mathscr{O}$, recommencer à l'étape 1.

L'entier x et le point Q de E constituent alors respectivement les clés secrète et publique de l'utilisateur qui les a générées. Afin de valider ce choix, il est également demandé de vérifier que $Q \in E$ et $nQ = \mathscr{O}$.

Signer un message m. Supposons à nouveau ici qu'Alice a généré une paire de clés (x, Q). Pour signer un message m qu'elle voudrait envoyer à Bob, elle va devoir :

1. choisir au hasard un entier k tel que $1 \leq k \leq n-1$,
2. calculer les coordonnées du point kG, que nous noterons (x_1, y_1),
3. calculer $r \equiv x_1 \bmod n$; si $r \equiv 0 \bmod n$, recommencer à l'étape 1,
4. calculer $H(m)$, le hash du message m par la fonction H, et convertir le mot obtenu en un entier e,
5. calculer $s \equiv k^{-1}(e + xr) \bmod n$; si $s \equiv 0 \bmod n$, recommencer à l'étape 1.

Le couple d'entiers (r, s) ainsi trouvé constitue alors la signature d'Alice pour le message m.

Vérifier une signature. Comme pour l'algorithme précédent, Bob veut vérifier que le message m qu'il a reçu provient bien d'Alice et non de quelqu'un d'autre, potentiellement malveillant. Pour valider la signature (r, s) d'Alice, il utilisera ici aussi sa clé publique Q, en procédant comme suit :

1. vérifier que $1 \leq r, s \leq n-1$,
2. calculer $H(m)$ et le convertir en l'entier e,
3. calculer $w \equiv s^{-1} \bmod n$,
4. calculer $u_1 \equiv ew \bmod n$ et $u_2 \equiv rw \bmod n$,
5. calculer les coordonnées du point $X = u_1 G + u_2 Q$, que nous noterons (x_2, y_2),
6. si $X = \mathscr{O}$, rejeter la signature ; sinon, calculer $v \equiv x_2 \bmod n$.

Le résultat suivant, analogue de la Proposition 3.4.1, permet alors de conclure.

Proposition 3.4.2. *En utilisant les mêmes notations que précédemment, une signature (r, s) est valide si et seulement si $v \equiv r \bmod n$.*

Démonstration. Si la signature (r, s) est valide, alors par définition

$$s \equiv k^{-1}(e + xr) \bmod n.$$

Dès lors,

$$\begin{aligned} k &\equiv (e + xr)\, s^{-1} \bmod n \\ &\equiv (e + xr)\, w \bmod n \\ &\equiv ew + xrw \bmod n \\ &\equiv u_1 + xu_2 \bmod n, \end{aligned}$$

par définition des entiers w, u_1 et u_2.

Par conséquent,

$$X = u_1 G + u_2 Q = u_1 G + u_2 xG = (u_1 + u_2 x)\, G \equiv kG \bmod n.$$

En particulier, $x_2 \equiv x_1 \bmod n$, *i.e.* $v \equiv r \bmod n$, ce qu'on voulait.

A nouveau, pour l'autre implication, il suffit alors de considérer le raisonnement effectué ci-dessus dans l'autre sens [36]. □

Remarque 3.4.3. Que ce soit pour le DSA ou l'ECDSA, le fait de "sélectionner au hasard" un entier dans un certain intervalle est régi par des méthodes et algorithmes particuliers. Les documents [34] et [35] en fournissent notamment des exemples.

36. Il faut néanmoins faire attention à ne considérer l'équivalence $X \equiv kG \bmod n$ qu'en termes de première composante et non de point, étant donné que rien n'affirme que $y_2 \equiv y_1 \bmod n$.

3.4.2 Dans le cadre du Bitcoin

Dans le protocole Bitcoin, l'ECDSA est utilisé pour générer les clés secrète et publique d'un utilisateur, qui lui serviront à signer ses transactions. En particulier, le protocole du Bictoin fait appel à des paramètres de domaine bien définis (que l'on peut retrouver dans [12]) :

- le champ $\mathbb{F}_p$ est défini par le nombre premier $p = 2^{256} - 2^{32} - 2^9 - 2^8 - 2^7 - 2^6 - 2^4 - 1$,
- la courbe elliptique secp256k1 définie sur $\mathbb{F}_p$, qui a pour équation

$$y^2 = x^3 + 7,$$

- les coordonnées [37] du point de base G sont données par

$x =$ 79BE667E F9DCBBAC 55A06295 CE870B07 029BFCDB 2DCE28D9 59F2815B 16F81798,
$y =$ 483ADA77 26A3C465 5DA4FBFC 0E1108A8 FD17B448 A6855419 9C47D08F FB10D4B8.

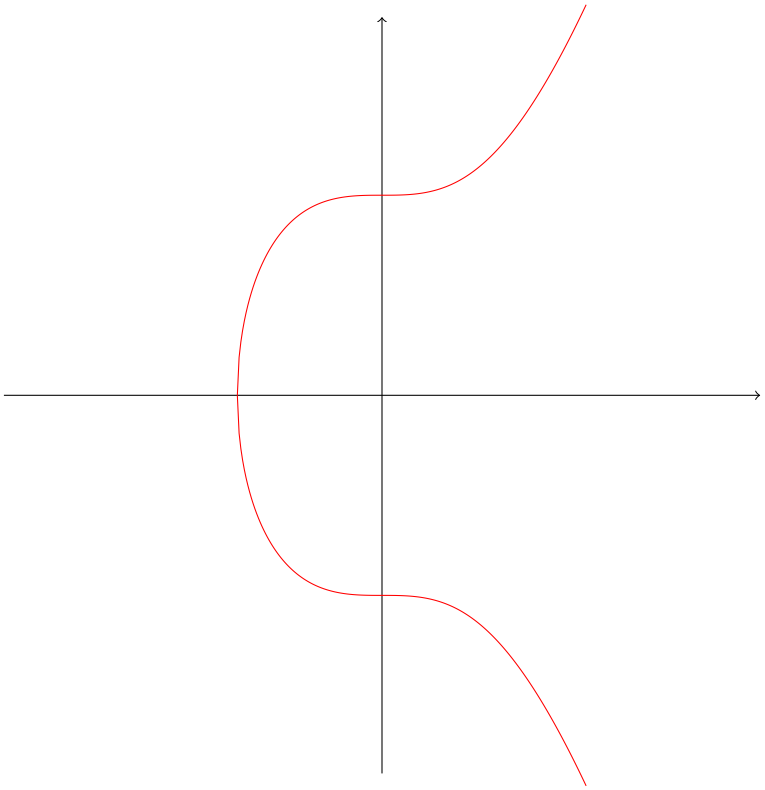

FIGURE 3.14 – Courbe secp256k1

Notons que la courbe est ici représentée dans le plan réel, mais rappelons que pour le Bitcoin, elle est vue comme une courbe du champ fini $\mathbb{F}_p$ spécifié précédemment.

Comme on peut le remarquer d'emblée, la courbe elliptique choisie est loin d'avoir des coefficients compliqués. C'est notamment ce qui en fait sa force, puisque la simplicité de ces paramètres permet des calculs plus rapides, sans pour autant altérer la sécurité du système (rappelons que c'est notamment pour cela que la cryptographie sur les courbes elliptiques s'est développée ces dernières années).

37. Celles-ci sont données selon leur représentation dans le système hexadécimal. La plupart du temps, on ne donne que la ***forme compressée*** du point de base, qui consiste uniquement en la coordonnée x du point (la coordonnée y pouvant être calculée à partir de l'équation de la courbe).

C'est donc par l'intermédiaire de la courbe secp256k1 qu'est générée la clé publique d'un utilisateur du réseau Bitcoin, à partir de sa clé secrète. Or, comme expliqué dans la description de l'ECDSA, cette dernière ne dépend pas de la courbe elliptique choisie : il s'agit d'un entier s choisi aléatoirement, strictement positif et plus petit que l'ordre n du point de base G. Nous ne décrirons pas son fonctionnement ici, mais précisons simplement que cet entier s est déterminé par un ***cryptographically secure pseudo-random number generator*** ou ***CSPRNG***, un algorithme permettant de générer des nombres ayant l'air aléatoires, mais qui en réalité sont déterminés à l'avance. Le but d'un tel algorithme est de se rapprocher au plus des propriétés d'une suite aléatoire.

Bibliographie

[1] Chaîne YouTube 3Blue1Brown, *But how does bitcoin actually work ?*, vidéo publiée le 7 juillet 2017, disponible via l'URL <https://youtu.be/bBC-nXj3Ng4>, consultée le 25 avril 2022.

[2] Sven Erick Alm, *Simple random walk*, Université d'Uppsala, Suède, 2006, disponible via l'URL <http://www2.math.uu.se/~sea/kurser/stokprocmn1/slumpvandring_eng.pdf>.

[3] Antoine Chambert-Loir, *Résultants*, Préparation à l'agrégation, Université Paris-Diderot, 2016, disponible via l'URL <https://webusers.imj-prg.fr/~antoine.chambert-loir/enseignement/2015-16/agreg/resultant.pdf>.

[4] Fred Cohen, *A Short History of Cryptography*, ch. 2 - Cryptographic Protection, 1995, disponible via l'URL <http://all.net/books/IP/Chap2-1.html>.

[5] Olivier Collin, *Notes de cours sur les courbes algébriques*, support du cours d'Olivier Collin sur les courbes algébriques (MAT3550) dispensé en 2019 et 2021, Université du Québec à Montréal, disponible via l'URL <https://maths.dur.ac.uk/users/mark.a.powell/MAT4030%20notes%20version%206-3-20112.pdf>.

[6] Sophia Crossen, *The Mathematics of Bitcoin*, Mémoire de fin d'études, Emporia State University, Kansas, 2015.

[7] Joan Daemen and Vincent Rijmen, *The Design of Rijndael : AES – The Advanced Encryption Standard*, Information Security and Cryptography, Springer-Verlag, Berlin, Heidelberg, 2002.

[8] Ivan Bjerre Damgärd, *A Design Principle for Hash Functions*, Advances in Cryptology – CRYPTO '89 Proceedings (G. Brassard, ed.), Lecture Notes in Computer Science, vol. 435, Springer, New York, 1990, pp. 416 – 427.

[9] Donald Davies, *A Brief History of Cryptography*, Information Security Technical Report **2** (1997), no. 2, pp. 14 – 17.

[10] Pascal Dupont, *Introduction à la géométrie : géométrie linéaire & géométrie différentielle*, Bibliothèque des universités. Mathématiques, De Boeck Université, Bruxelles, 2002.

[11] David Eisenbud, Mark Green, and Joe Harris, *Cayley-Bacharach Theorems and Conjectures*, Bulletin of the American Mathematical Society **33** (1996), no. 3, pp. 295 – 324.

[12] Standarts for Efficient Cryptography, *SEC 2 : Recommended Elliptic Curve Domain Parameters*, 2000, disponible via l'URL <https://www.secg.org/sec2-v2.pdf>.

[13] Damien Gorissen, *Cryptographie sur les courbes elliptiques*, Mémoire de fin d'études, Université de Liège, 2009.

[14] Mercedes Haiech and Aude Le Gluher, *Lectures dirigées de recherche, Théorème de Bezout*, Rapport de recherche, Université de Rennes 1, 2015, disponible via l'URL <http://perso.eleves.ens-rennes.fr/people/Mercedes.Haiech/stages/Lecture_dirige.pdf>.

[15] Darrel Hankerson, Alfred Menezes, and Scott Vanstone, *Guide to Elliptic Curve Cryptography*, Springer-Verlag, 2004.

[16] Georges Hansoul, *ALGÈBRE II (en ce compris des compléments d'algèbre linéaire)*, Université de Liège, 2010.

[17] ———, *ALGÈBRE III, Cours obligatoire au bachelier en sciences mathématiques*, Université de Liège, 2016.

[18] Robin Hartshorne, *Algebraic Geometry*, Graduate Texts in Mathematics, no. 52, Springer-Verlag, New York, 1977.

[19] Michael Hägler, *Courbes elliptiques et cryptographie*, Rapport de recherche, Université Clermont Auvergne, 2006, disponible via l'URL <https://lmbp.uca.fr/~rebolledo/page-fichiers/projetMichael.pdf>.

[20] Liao Hung-Zih and Shen Yuan-Yuan, *On the Elliptic Curve Digital Signature Algorithm*, Tunghai Science **8** (2006), pp. 109 – 126.

[21] Dale Husemoller, *Elliptic curves*, Graduate texts in mathematics, no. 111, Springer, New York, 2010.

[22] Don Johnson, Alfred Menezes, and Scott Vanstone, *The Elliptic Curve Digital Signature Algorithm (ECDSA)*, International Journal of Information Security **1** (2001), no. 1, pp. 36 – 63.

[23] Neal Koblitz, *Elliptic Curve Cryptosystems*, Mathematics of Computation **48** (1987), no. 117, pp. 203 – 209.

[24] ———, *A Course in Number Theory and Cryptography*, second ed., Graduate Texts in Mathematics, no. 114, Springer-Verlag, New York, 1994.

[25] ———, *Random Curves, Journeys of a Mathematician*, Springer, Berlin, Heidelberg, 2008.

[26] Alain Kraus, *Chapitre VII – Courbes elliptiques*, Chapitre issu du cours de cryptographie (MM067), Université Pierre et Marie Curie, 2009, disponible via l'URL <https://www.math.univ-paris13.fr/~boyer/enseignement/crypto/Chap7.pdf>.

[27] Aude Le Gluher, *Courbes cubiques projectives et tores complexes*, Rapport de stage, Institut de Mathématiques de Toulouse, 2016, disponible via l'URL <http://perso.eleves.ens-rennes.fr/people/Aude.Legluher/fr/rapportm1.pdf>.

[28] James L. Massey and Jimmy K. Omura, *Method and appartus for maintaining the privacy of digital messages conveyed by public transmission*, 1986, United States Patent and Trademark Office, disponible via l'URL <https://patents.google.com/patent/US4567600>.

[29] Alfred Menezes, Scott Vanstone, and Tatsuaki Okamoto, *Reducing Elliptic Curve Logarithms to Logarithms in a Finite Field*, IEEE Transactions on Information Theory **39** (1993), no. 5, pp. 1639 – 1646.

[30] Ralph C. Merkle, *A Certified Digital Signature*, Advances in Cryptology – CRYPTO '89 Proceedings (G. Brassard, ed.), Lecture Notes in Computer Science, vol. 435, Springer, New York, 1990, pp. 218 – 238.

[31] Victor S. Miller, *Use of Elliptic Curves in Cryptography*, Advances in Cryptology – CRYPTO '85 Proceedings (Hugh C. Williams, ed.), Lecture Notes in Computer Science, vol. 218, Springer, Berlin, Heidelberg, 1986, pp. 417 – 426.

[32] Satoshi Nakamoto, *Bitcoin : A Peer-to-Peer Electronic Cash System*, 2008, disponible via l'URL <https://bitcoin.org/bitcoin.pdf>.

[33] Arvind Narayanan, Joseph Bonneau, Edward Felten, Andrew Miller, and Steven Goldfeder, *Bitcoin and Cryptocurrency Technologies, A Comprehensive Introduction*, Princeton University Press, 2016.

[34] National Institute of Standards and Technology, *Digital Signature Standard (DSS)*, Federal Information Processing Standard Publications (2013), no. 186-4, disponible via l'URL <https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf>.

[35] ———, *Digital Signature Standard (DSS)*, Federal Information Processing Standard Publications (2019), no. 186-5, disponible via l'URL <https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-5-draft.pdf>.

[36] Elisabeth Oswald, *Introduction to Elliptic Curve Cryptography*, Institute for Applied Information Processing and Communication, 2002.

[37] A. Pinar Ozisik and Brian Neil Levine, *An Explanation of Nakamoto's Analysis of Double-spend Attacks*, Computing Research Repository (2017), disponible via l'URL <http://arxiv.org/abs/1701.03977>.

[38] Ilarion Pavel, *Introduction à la blockchain*, Annales des Mines – Réalités industrielles **4** (2019), pp. 98 – 104.

[39] Daniel Perrin, *Cours d'algèbre*, CAPES/AGREG Mathématiques, Ellipses, Paris, 1996.

[40] Luc Poitras, *Origines algébrique et géométrique des nombres complexes et leur extension aux quaternions ; Fondements de la géométrie*, Mémoire de fin d'étude, Université du Québec, Montréal, Août 2007.

[41] Leonid Reyzin, *Some Notions of Entropy for Cryptography*, Information Theoretic Security (Serge Fehr, ed.), Lecture Notes in Computer Science, vol. 6673, Springer, Berlin, Heidelberg, 2011, pp. 138 – 142.

[42] Michel Rigo, *Algèbre linéaire*, Université de Liège, notes de cours destinées aux étudiants de première année de bachelier en sciences mathématiques, 2009.

[43] ———, *Mathématiques discrètes*, Université de Liège, notes de cours destinées aux étudiants de master en sciences mathématiques, 2009.

[44] Ron L. Rivest, Adi Shamir, and Leonard Adleman, *A Method for Obtaining Digital Signatures and Public-Key Cryptosystems*, Communications of the ACM **21** (1978), no. 2, pp. 120 – 126.

[45] Robert Rolland, *Géométrie projective*, Publications de l'IREM de l'Académie d'Aix-Marseille, no. 30, IREM d'Aix-Marseille, Aix-Marseille, 2004.

[46] Thomas R. Shemanske, *Modern Cryptography and Elliptic Curves, A Beginner's Guide*, Student Mathematical Library, no. 83, Amercian Mathematical Society, Providence, Rhode Island, 2017.

[47] Joseph H. Silverman, *The arithmetic of elliptic curves*, Graduate Texts in Mathematics, no. 106, Springer-Verlag, New York, 1986.

[48] Stéphane Sordet, Jules Sandoz, Sylvain Arnaud, and Lucas Cuennet, *Le Bitcoin, un monstre énergivore*, Médiamorphoses (2021), no. 4931, revue publiée par les étudiants de 1ère année en ingénierie des médias, Haute École d'Ingénierie et de Gestion du Canton de Vaud, Suisse, disponible via l'URL <https://blog.comem.ch/wp-content/uploads/2022/02/mediamorphoses_4931.pdf>.

[49] Frank Olaf Wagner, *Chapitre 1 – Courbes elliptiques*, Chapitre issu du cours de Master 2 Professionnel portant sur la cryptographie sur les courbes elliptiques dispensé de 2011 à 2013, Institut Camille Jordan, Université Lyon I, disponible via l'URL <http://math.univ-lyon1.fr/~wagner/coursDelaunay.pdf>.

[50] Wikipédia, *Cryptographie asymétrique – Wikipédia, l'encyclopédie libre*, 2022, en ligne, disponible via l'URL <https://fr.wikipedia.org/w/index.php?title=Cryptographie_asym%C3%A9trique&oldid=192980984>, consulté le 22 avril 2022.

[51] ———, *Cryptographie symétrique – Wikipédia, l'encyclopédie libre*, 2022, en ligne, disponible via l'URL <http://fr.wikipedia.org/w/index.php?title=Cryptographie_sym%C3%A9trique&oldid=192305015>, consulté le 22 avril 2022.

[52] _____, *Histoire de la cryptologie – Wikipédia, l'encyclopédie libre*, 2022, en ligne, disponible via l'URL <https://fr.wikipedia.org/w/index.php?title=Histoire_de_la_cryptologie&oldid=192906419>, consulté le 22 avril 2022.

[53] Wikipedia, *Neal Koblitz – Wikipedia, The Free Encyclopedia*, 2022, en ligne, disponible via l'URL <https://en.wikipedia.org/w/index.php?title=Neal_Koblitz&oldid=1061156898>, consulté le 22 avril 2022.

[54] _____, *Victor S. Miller – Wikipedia, The Free Encyclopedia*, 2022, en ligne, disponible via l'URL <https://en.wikipedia.org/w/index.php?title=Victor_S._Miller&oldid=1073978998>, consulté le 22 avril 2022.

[55] Richard Winton, *Enhancing the Massey-Omura Cryptosystem*, Journal of Mathematical Sciences & Mathematics Education **2** (2007), no. 1, pp. 21 – 29.

[56] Wang Xiaoyun, Lisa Yin Yiqun, and Yu Hongbo, *Finding Collisions in the Full SHA-1*, Advances in Cryptology – CRYPTO 2005 Proceedings (Victor Shoup, ed.), Lecture Notes in Computer Science, vol. 3621, Springer, Berlin, Heidelberg, 2005, pp. 17 – 36.

[57] Taronisokhi Zebua, Rivalri Kristianto Hondro, and Eferoni Ndruru, *Message Security on Chat App based on Massey Omura Algorithm*, International Journal Of Information System & Technology **1** (2018), no. 2, pp. 16 – 23.