RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE

MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITÉ YAHIA FARÈS DE MÉDÉA

FACULTÉ DES SCIENCES

DÉPARTEMENT DE MATHÉMATIQUES ET INFORMATIQUE



MÉMOIRE DE FIN D'ÉTUDES

Présenté en vue de l'obtention du

DIPLÔME DE MASTER EN MATHÉMATIQUES

Spécialité : Analyse Combinatoire, Probabilités et Applications

Par

Khelladi Kheireddine

Aspect Algorithemique des Nombres Premiers

Soutenu publiquement le 29/06/2025, devant le jury composé de

Présidente :	Dr. Ait-Amrane N. R.	**M.C.B.**	Université de Médéa
Examinateur :	Mr. Krimi M.	**M.A.A.**	Université de Médéa
Encadrant :	Dr. Bendadou A.	**M.C.B.**	Université de Médéa

Année Universitaire : 2024-2025

Remerciement

Tout d'abord, je tiens à remercier **Allah**, le Miséricordieux et le Compatissant, de m'avoir donné la force et le courage nécessaires pour mener à bien ce modeste travail. De plus, c'est avec plaisir que je remercie toutes les personnes qui ont contribué à la réalisation de ce mémoire.

Tout d'abord. Je tiens à exprimer mes sincères remerciements à mon directeur de thèse, le Dr. **Bendadou A**. M.C.B. pour m'avoir guidé tout au long de cette étude et pour son soutien scientifique et moral.

Dr. **Ait-Amrane N.R**. M.C.B. à l'université Dr.Yahia Fares de Médéa, pour nous avoir fait l'honneur de présider le jury de ce mémoire.

Je tiens à remercier le Mr. **Krimi M**. M.A.A. à l'université Dr.Yahia Fares de Médéa, pour m'avoir fait l'honneur d'être membre du jury et pour avoir consacré une partie de son temps précieux à la lecture et à la correction de ce mémoire.

Enfin, je tiens à remercier toutes les personnes qui ont contribué, directement ou indirectement, à la réalisation de ce projet.

Dédicaces

Je dédie ce modeste travail à :

- Ma famille, ma source d'amour et de force,
- Ma chère mère, Tahraoui Malika, ton sacrifice et ton amour infini m'ont porté jusqu'à ce jour,
- Mon père, Khellad el Akdar, pour tes encouragements et ta sagesse,
- Mes frères et sœurs, pour votre soutien inconditionnel,
- Ma seconde mère, ma tante Tahraoui Bahia, pour ton affection et ta présence à chaque étape,
- Tous mes amis, ceux rencontrés sur les bancs de l'université et ailleurs,
- Mes professeurs, qui m'ont transmis leur savoir et guidé dans ce parcours.

Cette réalisation est le fruit de votre appui, de vos prières et de votre confiance. Qu'elle soit un hommage à votre bienveillance.

Avec tout mon amour et ma gratitude.

Résumé

Dans ce travail, nous nous sommes intéressés à l'aspect algorithmique des nombres premiers, en abordant diverses méthodes utilisées pour tester la primalité d'un nombre donné. Ces méthodes trouvent de nombreuses applications, notamment en cryptographie.

Nous avons étudié en particulier une méthode reconnue comme étant la seule basée sur une approche déterministe et de complexité polynomiale. Dans le but de réduire sa complexité, nous proposons une technique simple qui peut être appliquée à ses différentes variantes.

Notations

Dans ce mémoire, nous utiliserons les notations suivantes :

$a \mid b$: relation de divisibilité (a divise b).
$a \equiv b \mod c$ signifie que $a-b$ est multiple de c.
$\gcd(a,b)$: le plus grand commun diviseur de a et b.
$D(n)$ désigne l'ensemble des diviseurs de n .
$|a|$: la valeur absolue de a.
$\mathbb{Z}$: l'anneau des entiers relatifs $\{\ldots,-2,-1,0,1,2,\ldots\}$.
$\mathbb{Z}_n$: l'anneau des entiers modulo n .
$\mathbb{Z}_p^*$: le corps des entiers inversibles modulo p (pour p premier) .
$|\mathbb{Z}_p^*|$: le nombre des éléments de $\mathbb{Z}_p^*$.
$+_m$: l'opérateur addition dans l'anneau Z_m.
$\cdot_m$: l'opérateur multiplication dans l'anneau Z_m.
$R[X]$: l'anneau des polynômes à coefficients dans R.
$R[X]\backslash h$: l'anneau des polynômes à coefficients dans R et de degré inférieur à $deg(h)$.
$f \equiv r \mod h$ dans l'anneau $R[X]\backslash h$ signifie que $f = qh+r$ (q est le quotient de la division) .
$+_h$: l'opérateur addition dans l'anneau $R[X]\backslash h$.
$\cdot_h$: l'opérateur multiplication dans l'anneau $R[X]\backslash h$.
$\log x$ désigne le logarithme de base 2.
RSA : l'abréviation universellement reconnue pour l'algorithme de «Rivest, Shamir et Adleman».
AKS : l'abréviation universellement reconnue pour le test de primalité «Agrawal-Kayal-Saxena».
$\phi(n)$: la fonction indicatrice d'Euler comptant les entiers premiers avec n .
$\left(\frac{a}{b}\right)$: le symbole de Legendre pour les entiers a et b.
$ord_r(n)$: l'ordre multiplicatif de n modulo r.

Introduction générale

Les nombres premiers ont toujours intrigué les gens au fils des âges. De nombreuses traces en témoignent [2], comme l'os d'Ishago, découvert et issu d'une époque datant à plusieurs milliers années, qui semblerait contenir un tableau mystérieux de quelques nombres premiers. Ainsi, les anciens égyptiens ont montré l'intention pour les nombres premières dans le calcule fractionnaire qu'ils développaient. D'après leurs traces, il semblerait qu'ils éprouvaient des difficultés à décomposer la fraction $2/n$ (où n est un nombre premier compris entre 4 et 102) en une somme de deux fractions unitaires, notamment lorsque n était un nombre premier. On trouve également les anciens Grecs, reconnus comme les premiers avoir étudié les nombres premiers de manière systématique. Par exemple, Euclide a établi des propriétés essentielles, telles que l'infinité des nombres premiers, qui restent utilisées à ce jour. Enfin, les arabes ont également cités à travers leurs contributions notables à l'étude des nombres Thabit.
Récemment, les nombres premiers ont cessé d'être un concept purement abstrait. Leur importance est devenue de plus en plus manifeste avec l'exploration de leurs applications dans divers domaines, particulièrement en cryptographie. En effet, plusieurs méthodes cryptographiques développées sont fondées sur la génération de nombres premiers, une tâche reconnue comme difficile lorsqu'il s'agit de nombres de grande taille.
Ce modeste travail explore le rôle des nombres premiers en cryptographie, présente diverses méthodes de test de primalité et propose une technique susceptible d'accélérer ce test en utilisant la méthode A.K.S.
Dans le premier chapitre, nous présentons des notions préliminaires essentielles à l'algèbre et à la théorie des nombres. Nous commençons par un rappel sur la divisibilité, suivi d'une présentation de l'algorithme d'Euclide pour le calcul du plus grand commun diviseur (PGCD). Ensuite, nous abordons la caractérisation des nombres premiers et composés, ainsi que des méthodes classiques telles que l'algorithme d'Ératosthène. De plus, nous introduisons des concepts plus avancés comme les résidus quadratiques et le symbole de Legendre, ce qui nous permettra par la suite de présenter les témoins

des nombres composés et premiers. Enfin, nous rappelons les bases de la complexité algorithmique pour mieux comprendre l'efficacité des méthodes que nous étudierons.
Dans le deuxième chapitre, nous examinons l'importance des nombres premiers en cryptographie, en nous concentrant notamment sur l'algorithme RSA, qui repose sur la complexité de la factorisation de grands entiers. Nous présentons ensuite en détail des tests de primalité probabilistes : test de Fermat, de Miller-Rabin et de Solovay-Strassen. Bien qu'efficaces, ces méthodes ne garantissent pas une certitude absolue quant à la primalité d'un nombre.
Dans le chapitre trois, nous détaillons l'algorithme AKS, la seule méthode déterministe connue à ce jour pour tester la primalité d'un nombre en temps polynomial. Bien que cette méthode soit classée comme polynomiale, elle nécessite encore de nombreux calculs, surtout lorsqu'il s'agit de très grands nombres. Afin de réduire cette charge computationnelle, nous proposons une technique simple qui pourrait également être appliquée à d'autres variantes de l'algorithme AKS.

TABLE DES MATIÈRES

Résumé iii

Notations iv

Introduction générale v

1 Préliminaires 1

1.1 Divisibilité 1
1.2 Arithmétique modulaire 3
1.3 Nombres premiers et composés 6
1.4 Structure Algébrique 7
1.5 Résidus quadratiques 12
1.6 Symbole de Legendre 13
1.7 Témoins des Nombres Composés 15
1.8 Témoins des Nombres Premiers 16
1.9 Rappel sur la Complexité 17

2 Cryptographie et Tests de Primalité 20

2.1 Introduction à la Cryptographie 20
2.2 Cryptographie symétrique 22
2.3 Cryptographie asymétrique 23
2.4 Tests de primalité 27

3 Test de primalité A.K.S 33

3.1 Algorithme 33
3.2 Analyse de complexité temporelle et améliorations 35
3.3 Application 37

Chapitre 1

Préliminaires

Ce chapitre est dédié aux notions de base et aux concepts préliminaires essentiels pour bien appréhender les sujets qui seront abordés par la suite. Nous y établirons les définitions clés, les théorèmes fondamentaux et les outils nécessaires à la compréhension. Pour plus de détails consulter l'ouvrage [4] et l'article [1].

1.1 Divisibilité

Définition 1.1

On dit qu'n entier n non nul divise un entier m, si $n \times x = m$ pour un certain entier x. Si tel est le cas, on note $n \mid m$, et on dit également que n est un diviseur de m ou que m est un multiple de n.

Proposition 1.1

Soient n, m, k des entiers non nuls. Alors :

(a) Si $n \mid m$ et $n \mid k$, alors $n \mid (mu + kv)$ pour des entiers arbitraires u et v.

(b) Si $n, m > 0$ et $n \mid m$, alors $n \leq m$.

(c) Si $n \mid m$ et $m \mid k$, alors $n \mid k$.

(d) Si $n \mid m$, alors $(-n) \mid m$ et $n \mid (-m)$.

Démonstration. se référer à [4]. □

Définition 1.2

Pour un entier n, soit $D(n)$ l'ensemble des diviseurs non négatifs de n.

Définition 1.3

1. Si n et m sont des entiers, différents de 0, alors le plus grand entier qui divise n et qui divise m est appelé le plus grand commun diviseur de n et m et est

noté $\gcd(n,m)$. Par conséquent $\gcd(n,m) \in D(n) \cap D(m)$.

2. Il est pratique de définir (de manière quelque peu arbitraire) $\gcd(0,0) = 0$.
3. Si $\gcd(n,m) = 1$, alors on dit que n et m sont premiers entre eux.

Exemple 1.1

- *$D(0)$ comprend tous les entiers non négatifs.*
- $D(60) = D(-60) = \{1,2,3,4,5,6,10,12,15,20,30,60\}$.
- *Pour $n \neq 0$, $D(n)$ ne contient que des nombres positifs, dont* 1.
- *Nous avons* $\gcd(4,15) = 1$, $\gcd(4,6) = 2$, $\gcd(4,0) = \gcd(4,4) = 4$.
- *D'après la Proposition 1.1, nous avons que $D(n) = D(-n)$, que $D(n)$ est partiellement ordonné par la relation $|$, et qu'il est fermé vers le bas sous cette relation.*

Proposition 1.2

Si n est un entier et d un entier positif (le diviseur ou module), alors il existe des entiers uniques q (le quotient) et r (le reste) tels que

$$n = dq + r, \quad \text{et} \quad 0 \leq r < d.$$

Démonstration. se référer à [4]. □

Il existe des algorithmes assez efficaces pour trouver ces nombres q et r à partir de n et d (voir [4]).

Définition 1.4

Pour un entier n et un entier positif d, nous posons

$$n \mod d = r, \quad \text{et}$$
$$n \operatorname{div} d = q,$$

où r et q sont les nombres uniques qui satisfont $n = qd + r$, $0 \leq r < d$, comme dans la Proposition 1.2. Notez que $n \operatorname{div} d = \lfloor n/d \rfloor$, qui désigne le plus grand entier ne dépassant pas n/d.

Proposition 1.3

Si $m \geq 1$, alors $\gcd(n,m) = \gcd(m, (n \mod m))$ pour tout entier $n \geq m$.

Démonstration. se référer à [4]. □

Exemple 1.2

- $16 \operatorname{div} 4 = 4, \qquad 16 \mod 4 = 0,$
  $9 \operatorname{div} 4 = 2, \qquad 9 \mod 4 = 1,$
- $\gcd(16,6) = 2 = \gcd(6,2), \qquad \gcd(12,9) = 3 = \gcd(9,3).$

1.1.1 Algorithme d'Euclide

L'algorithme d'Euclide est une méthode efficace pour calculer le plus grand commun diviseur de deux nombres entiers naturels. Il est basé sur la répétition de la règle notée dans la proposition 1.3.

Algorithme 1 Algorithme d'Euclide

1: **Entrée :** Deux entiers n, m.
2: Initialiser a, b : entier ;
3: **si** $|n| \geq |m|$ **alors**
4: $\quad a \leftarrow |n|$; $b \leftarrow |m|$;
5: **sinon**
6: $\quad b \leftarrow |m|$; $a \leftarrow |n|$;
7: **fin si**
8: **tant que** $b > 0$ **faire**
9: $\quad (a, b) \leftarrow (b, a \pmod b)$;
10: **fin tant que**
11: **retourner** a ;

1.2 Arithmétique modulaire

Soit $m \geq 2$ un entier (fixé le module). Dans le calcule modulaire, on s'intéresse pas réellement à a mais seulement au reste $a \mod m$. Ainsi, tous les nombres qui correspond au même reste lorsqu'ils sont divisés par m sont considérés comme similaires.

Définition 1.5

Soit $m \geq 2$ un entier donné. Pour tous entiers a et b, on dit que a est congru à b modulo m, noté

$$a \equiv b \pmod m,$$

si et seulement si $a \mod m = b \mod m$.

Lemme 1.1. Si $a \equiv a' \mod m$ et $b \equiv b' \mod m$, alors pour tous $a, a', b, b' \in \mathbb{Z}$:

$$a + b \equiv a' + b' \mod m,$$
$$a \cdot b \equiv a' \cdot b' \pmod m.$$

Par conséquent, si $a \equiv a' \pmod m$ et $n \geq 0$ est arbitraire, alors :

$$a^n \equiv (a')^n \pmod m.$$

Démonstration. se référer à [4]. □

Définition 1.6

Pour $m \geq 2$, on définit $\mathbb{Z}_m = \{0, 1, \ldots, m-1\}$. Sur cet ensemble, on définit deux opérations :

$$a +_m b = (a + b) \mod m,$$
$$a \cdot_m b = (a \cdot b) \mod m.$$

(L'indice m peut être omis lorsqu'il n'y a pas d'ambiguïté.)

Lemme 1.2. Les opérations $+_m$ et $\cdot_m$ vérifient les propriétés suivantes pour tous $a, b, c \in \mathbb{Z}_m$:

(a) Commutativité :

$$a +_m b = b +_m a,$$
$$a \cdot_m b = b \cdot_m a.$$

(b) Associativité :

$$(a +_m b) +_m c = a +_m (b +_m c),$$
$$(a \cdot_m b) \cdot_m c = a \cdot_m (b \cdot_m c).$$

(c) Distributivité :

$$(a +_m b) \cdot_m c = (a \cdot_m c) +_m (b \cdot_m c).$$

(d) Éléments neutres :

$$a +_m 0 = 0 +_m a = a,$$
$$a \cdot_m 1 = 1 \cdot_m a = a.$$

(e) Inverse additif :

$$a +_m (m - a) = (m - a) +_m a = 0.$$

Démonstration. voir [4]. □

Exemple 1.3

Considérons $m = 5$, *donc* $\mathbb{Z}_5 = \{0,1,2,3,4\}$.

- ***Addition modulo 5*** $(+_5)$ *:*

$$3 +_5 4 = (3+4) \mod 5 = 7 \mod 5 = 2.$$
$$2 +_5 4 = (2+4) \mod 5 = 6 \mod 5 = 1.$$

- ***Multiplication modulo 5*** $(\cdot_5)$ *:*

$$3 \cdot_5 4 = (3 \times 4) \mod 5 = 12 \mod 5 = 2.$$
$$2 \cdot_5 3 = (2 \times 3) \mod 5 = 6 \mod 5 = 1.$$

- ***Inverse additif*** *:*
  - *L'inverse de 2 est 3 car* $2 +_5 3 = 0$.
  - *L'inverse de 4 est 1 car* $4 +_5 1 = 0$.
- ***Puissance modulo 5*** *:*

$$2^3 \mod 5 = 8 \mod 5 = 3.$$
$$4^2 \mod 5 = 16 \mod 5 = 1.$$

Définition 1.7

Pour $m \geq 1$, on définit :

$$\mathbb{Z}_m^* = \{a \mid 1 \leq a < m, \gcd(a,m) = 1\} \text{ et } \varphi(m) = |\mathbb{Z}_m^*|.$$

La fonction φ est appelée fonction indicatrice d'Euler ou fonction totient d'Euler.

Proposition 1.4

Soit $m \geq 1$:

(a) $1 \in \mathbb{Z}_m^*$.

(b) Si $a, b \in \mathbb{Z}_m^*$, alors $a \cdot_m b \in \mathbb{Z}_m^*$.

(c) $a \in \mathbb{Z}_m^*$ si et seulement s'il existe $b \in \mathbb{Z}_m$ tel que $a \cdot_m b = 1$.

Démonstration. se référer à [4]. □

Exemple 1.4

- *Dans* $\mathbb{Z}_7^* = \{1,2,3,4,5,6\}$, *on a :*

$$3 \cdot_7 3 = 2 \in \mathbb{Z}_7^*, \text{ et } 3 \cdot_7 5 = 1 \in \mathbb{Z}_7^*.$$

- $\varphi(7) = 6$.

Définition 1.8

Soient a et b deux entiers premiers entre eux. L'ordre multiplicatif de a modulo b, noté $ord_b(a)$, est le plus petit entier positif k tel que $(a^k - 1)|b$.

1.3 Nombres premiers et composés

Définition 1.9

Un entier positif n est appelé nombre premier (ou simplement premier), si $n > 1$ et s'il n'existe aucun entier divisant n autre que 1 et n lui-même. Si n est divisible par un entier a avec $1 < a < n$, alors n est appelé nombre composé.

Theorem 1.1. (Théorème fondamental de l'arithmétique) Tout entier $n \geq 1$ peut s'écrire comme un produit de nombres premiers d'une unique manière (à l'ordre des facteurs).

Démonstration. se référer à [4]. □

Exemple 1.5

- *Voici la liste des 25 nombres premiers entre 1 et 100 :*

$$2,3,5,7,11,13,17,19,23,29,31,37,41,43,47,53,59,61,67,71,73,79,83,89,97.$$

- $6 = 2 \cdot 3$.
- $60 = 2 \cdot 2 \cdot 3 \cdot 5$.

Theorem 1.2. Il existe une infinité de nombres premiers

Démonstration. consulter [4]. □

1.3.1 Crible d'Eratosthène

Le Crible d'Ératosthène est la plus simple méthode fréquemment utilisée pour déterminer les nombres premiers inférieurs à un certain entier n. Il s'agit d'un algorithme

ancien décrit comme suit :
On considère la liste de tous les nombres dans $\{2,\ldots,n\}$. Initialement, tous les nombres sont non marqués. Pour $j = 2,3,\ldots,\sqrt{n}$, on fait : si j est non marqué, alors on marque tous les multiples de j entre j^2 et n. Par conséquent, les nombres qui restent non marqués sont exactement les nombres premiers dans l'intervalle $[2,n]$.

1.4 Structure Algébrique

Définition 1.10

On appelle *loi de composition interne* sur un ensemble non vide A toute application $\Psi : A \times A \to A$.

Définition 1.11

Un *groupe* $(G,*)$ est un ensemble G muni d'une loi de composition interne $*$ sur G qui vérifie les propriétés suivantes :

(a) (**Associativité**) Pour tous $a,b,c \in G$, $(a*b)*c = a*(b*c)$.

(b) (**Élément neutre**) Il existe un élément $e \in G$ qui vérifie $a*e = e*a = a$ pour chaque $a \in G$.

(c) (**Élément inverse**) Pour chaque $a \in G$, il existe un élément $b \in G$ tel que $a*b = b*a = e$, où e est l'élément neutre.

Définition 1.12

Un ensemble R muni de deux opérations, notées $+$ et $\cdot$, est dit un anneau si les propriétés suivantes sont satisfaites :

- **(Élément Neutre Additif)** Il existe un élément neutre additif $0 \in R$ tel que $0+a = a+0 = a$ pour tout $a \in R$.
- **(Opposé Additif)** Pour tout élément $a \in R$, il existe un inverse additif $b \in R$ tel que $a+b = b+a = 0$.
- **(Associativité de l'Addition)** Pour tous $a,b,c \in R$, nous avons $a+(b+c) = (a+b)+c$.
- **(Commutativité de l'Addition)** Pour tous $a,b \in R$, nous avons $a+b = b+a$.
- **(Élément Neutre Multiplicatif)** Il existe un élément neutre multiplicatif $1 \in R$ tel que $1 \cdot a = a \cdot 1 = a$ pour tout $a \in R$.
- **(Associativité de la Multiplication)** Pour tous $a,b,c \in R$, nous avons $a \cdot (b \cdot c) = (a \cdot b) \cdot c$.
- **(Commutativité de la Multiplication)** Pour tous $a,b \in R$, nous avons $a \cdot b = b \cdot a$.

- **(Distributivité de la Multiplication sur l'Addition)** Pour tous $a, b, c \in R$, nous avons $a \cdot (b + c) = a \cdot b + a \cdot c$.

Définition 1.13

Un ensemble R muni de deux opérations, notées $+$ et $\cdot$, est dit un corps s'il est un anneau dans lequel la multiplication est commutative, et si de plus, chaque élément non nul de R possède un inverse multiplicatif.

Proposition 1.5

Pour tout entier $m \geq 2$, la structure $\mathbb{Z}_m = \{0, 1, \ldots, m-1\}$, munie des opérations binaires :

$$a +_m b = (a + b) \mod m \text{ et } a \cdot_m b = (a \cdot b) \mod m,$$

où les nombres 0 et 1 sont respectivement les éléments neutres, forme un anneau.

Démonstration. se référer à [4]. □

Proposition 1.6

Soit $m \geq 1$ un entier.

(a) Si $a_1 \equiv a_2 \mod m$ et $b_1 \equiv b_2 \mod m$, alors :

$$a_1 \pm b_1 \equiv a_2 \pm b_2 \mod m \text{ et } a_1 \cdot b_1 \equiv a_2 \cdot b_2 \mod m.$$

(b) Pour tout entier a, il existe un entier b tel que :

$$a \cdot b \equiv 1 \pmod m,$$

si et seulement si $\gcd(a, m) = 1$. Un tel b est appelé l'inverse multiplicatif de a modulo m (il est unique modulo m).

Démonstration. se référer à [4]. □

Proposition 1.7

Pour tout entier $m \geq 2$, les assertions suivantes sont équivalentes :

1. L'anneau $\mathbb{Z}_m = \{0, 1, \ldots, m-1\}$ est un corps.
2. m est un nombre premier.

Démonstration. se référer à [4]. □

Définition 1.14

Pour $m \geq 1$, on définit :

$$\mathbb{Z}_m^* = \{a \mid 1 \leq a < m, \gcd(a,m) = 1\}, \text{ et } \varphi(m) = |\mathbb{Z}_m^*|.$$

La fonction φ est appelée indicatrice d'Euler ou fonction totient d'Euler.

Theorem 1.3. (Théorème d'Euler) Pour tout entier $m \geq 2$, tous les éléments $a \in \mathbb{Z}_m^*$ vérifient :

$$a^{\varphi(m)} \equiv 1 \mod m.$$

où $\varphi(m)$ désigne l'indicatrice d'Euler.

Démonstration. voir [4]. □

Définition 1.15

Soit $(R, +, \cdot, 0, 1)$ un anneau. On définit $R[X]$ comme l'ensemble de toutes les suites (formellement infinies) :

$$(a_0, a_1, \ldots), \text{ avec } a_0, a_1, \ldots \in R,$$

où seul un nombre fini de a_i est non nul. Les éléments de $R[X]$ sont appelés polynômes sur R (plus précisément : polynômes en une variable sur R). On note généralement les polynômes par $f, g, h, \ldots$

Définition 1.16

Soit $(R, +, \cdot, 0, 1)$ un anneau. Pour $f = (a_0, a_1, \ldots)$ et $g = (b_0, b_1, \ldots)$ dans $R[X]$, on définit :

(a) L'addition :

$$f + g = (a_0 + b_0, a_1 + b_1, \ldots).$$

(b) La multiplication :

$$f \cdot g = (c_0, c_1, \ldots), \text{ où } c_i = \sum_{k=0}^{i} a_k \cdot b_{i-k} \text{ pour } i = 0, 1, \ldots$$

(Seul un nombre fini de c_i peut être non nul.)

Définition 1.17

Pour $f=(a_0,a_1,\ldots)\in R[X]$, on définit le degré :

$$\deg(f)=\begin{cases}\max\{i\mid a_i\neq 0\} & \text{si } f\neq 0,\\ -\infty & \text{si } f=0.\end{cases}$$

Proposition 1.8

Soit R un anneau et soit $h\in R[X]$ un polynôme unitaire (ou plus généralement un polynôme non nul dont le coefficient dominant est inversible dans R). Alors, pour tout polynôme $f\in R[X]$, il existe des polynômes uniques $q,r\in R[X]$ tels que :

$$f=h\cdot q+r,\ \text{avec}\ \deg(r)<\deg(h).$$

Démonstration. se référer à [4]. □

Exemple 1.6

Exemple de division euclidienne dans $\mathbb{Z}_5[X]$. Considérons les polynômes suivants dans $\mathbb{Z}_5[X]$:

$$f(X)=3X^3+2X^2+4X+1,\quad h(X)=X^2+2X+3.$$

Nous voulons effectuer la division euclidienne de f par h, c'est-à-dire trouver $q(X)$ et $r(X)$ tels que :

$$f(X)=h(X)\cdot q(X)+r(X),\ \textit{avec}\ \deg(r)<\deg(h)=2.$$

***Étapes de la division** :*

1. *Division du terme dominant : On divise le terme de plus haut degré de f par celui de h :*

$$\frac{3X^3}{X^2}=3X.$$

On pose donc $q_1(X)=3X$, et on calcule :

$$h(X)\cdot 3X=(X^2+2X+3)\cdot 3X=3X^3+6X^2+9X\equiv 3X^3+X^2+4X \mod 5.$$

On soustrait ce résultat à $f(X)$:

$$f(X)-h(X)\cdot 3X=(3X^3+2X^2+4X+1)-(3X^3+X^2+4X)=X^2+1.$$

2. *Division du reste partiel : On divise le terme de plus haut degré du reste partiel par celui de h :*

$$\frac{X^2}{X^2}=1.$$

On pose donc $q_2(X) = 1$, *et on calcule :*

$$h(X) \cdot 1 = X^2 + 2X + 3.$$

On soustrait ce résultat au reste partiel :

$$(X^2 + 1) - (X^2 + 2X + 3) = -2X - 2 \equiv 3X + 3 \mod 5.$$

3. *Résultat final : Le quotient est* $q(X) = 3X + 1$. *Le reste est* $r(X) = 3X + 3$ *(vérifiant* $\deg(r) = 1 < 2$*).*

Vérification *: Calculons* $h(X) \cdot q(X) + r(X)$ *:*

$$(X^2 + 2X + 3)(3X + 1) + (3X + 3).$$

Développons $h \cdot q$ *:*

$$3X^3 + X^2 + 6X^2 + 2X + 9X + 3 = 3X^3 + 7X^2 + 11X + 3 \equiv 3X^3 + 2X^2 + X + 3 \mod 5.$$

Ajoutons $r(X)$ *:*

$$3X^3 + 2X^2 + X + 3 + 3X + 3 = 3X^3 + 2X^2 + 4X + 6 \equiv 3X^3 + 2X^2 + 4X + 1 \mod 5,$$

ce qui redonne bien $f(X)$. *La division est donc correcte. La division euclidienne de* $f(X)$ *par* $h(X)$ *dans* $\mathbb{Z}_5[X]$ *donne :*

$$f(X) = (X^2 + 2X + 3)(3X + 1) + (3X + 3).$$

Définition 1.18

Soient $f, h \in R[X]$. Nous disons que h divise f (ou h est un diviseur de f, ou f est divisible par h) si $f = h \cdot q$ pour un certain $q \in R[X]$. Si $0 < \deg(h) < \deg(f)$, alors h est appelé un diviseur propre de f.

Définition 1.19

Soit $h \in R[X]$ un polynôme dont le coefficient dominant est une unité. Pour $f, g \in R[X]$, nous disons que f et g sont congrus modulo h, en symboles $f \equiv g \mod h$, si et seulement si $f - g$ est divisible par h.

Lemme 1.3. Supposons que $h \in R[X] \setminus \{0\}$ est un polynôme unitaire, $f_1 \equiv f_2 \mod h$, et $g_1 \equiv g_2 \mod h$. Alors :

(a) $f_1 + g_1 \equiv f_2 + g_2 \mod h$.

(b) $f_1 \cdot g_1 \equiv f_2 \cdot g_2 \pmod{h}$.

(c) $f(g_1) \equiv f(g_2) \pmod{h}$ pour tout $f \in R[X]$.

Démonstration. consulter [4]. □

Définition 1.20

Si $(R,+,\cdot,0,1)$ est un anneau, et $h \in R[X]$ est un polynôme unitaire de degré $d \geq 0$, nous désignons par $R[X]/(h)$ l'ensemble de tous les polynômes de $R[X]$ de degré strictement inférieur à d, muni des opérations suivantes $+_h$ et $\cdot_h$: $f +_h g = (f+g) \bmod h$, et $f \cdot_h g = (f \cdot g) \bmod h$, pour $f, g \in R[X]/(h)$.

Proposition 1.9

Si R et h sont définis comme dans la définition précédente, alors $(R[X]/(h), +_h, \cdot_h, 0, 1)$ est un anneau. De plus, nous avons :

(a) $f \bmod h = f$ si et seulement si $\deg(f) < d$.

(b) $(f+g) \bmod h = ((f \bmod h)+(g \bmod h)) \bmod h$ et $(f \cdot g) \bmod h = ((f \bmod h)\cdot(g \bmod h)) \bmod h$, pour tous $f, g \in R[X]$.

(c) Si $g_1 \equiv g_2 \pmod{h}$, alors $f(g_1) \bmod h = f(g_2) \bmod h$, pour tous $f, g_1, g_2 \in R[X]$.

Démonstration. se référer à [4]. □

1.5 Résidus quadratiques

Définition 1.21

Pour $m \geq 2$ et $a \in \mathbb{Z}$, avec $\gcd(a,m) = 1$, nous disons que a est un résidu quadratique modulo m si $a \equiv x^2 \pmod{m}$ pour un certain $x \in \mathbb{Z}$. Si a satisfait $\gcd(a,m) = 1$ et n'est pas un résidu quadratique modulo m, on l'appelle un non-résidu quadratique.

Il est clair que le fait d'être ou non un résidu quadratique est une propriété de la classe de congruence de a. Souvent, mais pas toujours, nous limitons notre attention au groupe $\mathbb{Z}_m^*$ qui contient un représentant de chaque classe de congruence en question. Dans ce contexte, -1 représente toujours l'inverse additif de 1, c'est-à-dire $m-1$. Notez que les nombres a avec $\gcd(a,m) > 1$ ne sont considérés ni comme des résidus quadratiques ni comme des non-résidus.

Exemple 1.7

Pour $m = 13$, les carrés modulo 13 de $1, 2, \ldots, 12$ sont $1, 4, 9, 3, 12, 10, 10, 12, 3, 9, 4, 1$, c'est-à-dire que les résidus quadratiques sont $1, 3, 4, 9, 10, 12$. Pour $m = 26$, les résidus quadratiques

sont $1,3,9,17,23,25$*; pour* $m=27$*, ils sont* $1,4,7,10,13,16,19,22,25$*. Nous observons que dans le cas* $m=13$*, il y a 6 résidus et 6 non-résidus. Ce comportement est typique pour* $m=p$ *un nombre premier. Si nous élevons au carré les nombres* $1,\dots,p-1$*, nous obtenons au plus* $\frac{1}{2}(p-1)$ *valeurs distinctes, puisque* $x^2 \equiv (p-x)^2 \mod p$*. D'autre part, les carrés de* $1,\dots,\frac{1}{2}(p-1)$ *sont tous distincts : si* $x^2 \equiv y^2 \pmod p$ *pour* $1 \le x \le y < \frac{1}{2}p$*, alors* p *divise* $y^2-x^2=(x+y)(y-x)$*. Puisque* $0 \le y-x < x+y < p$*, cela ne peut être le cas que si* $y=x$ *ou* $y=p-x$*. En d'autres termes, tout* $a \in \mathbb{Z}_p^*$ *qui est un résidu quadratique modulo* p *a exactement deux « racines carrées » modulo* p*, c'est-à-dire des nombres* $x \in \mathbb{Z}_p^*$ *avec* $x^2=a$ *dans* $\mathbb{Z}_p^*$*. Dans le cas particulier où* p *est un nombre premier, il n'est pas difficile de déterminer si* $a \in \mathbb{Z}_p^*$ *est un résidu quadratique ou non.*

1.6 Symbole de Legendre

Définition 1.22

Soient $p \ge 3$ un nombre premier et $a \in \mathbb{Z}$ un entier. On définit le symbole de Legendre $\left(\frac{a}{p}\right)$ par :

$$\left(\frac{a}{p}\right) = \begin{cases} 0 & \text{si } p \mid a, \\ 1 & \text{si } a \text{ est un résidu quadratique modulo } p, \\ -1 & \text{si } a \text{ est un non-résidu quadratique modulo } p. \end{cases}$$

Ce symbole indique si l'équation $x^2 \equiv a \mod p$ admet une solution.

Le symbole de Legendre satisfait plusieurs propriétés simples : Soit $p \ge 3$ un nombre premier. Pour tous entiers $a, b, c \in \mathbb{Z}$, on a :

(a) Multiplicativité :

$$\left(\frac{ab}{p}\right) = \left(\frac{a}{p}\right)\left(\frac{b}{p}\right).$$

(b) Invariance par multiplication par un carré (si $p \nmid b$) :

$$\left(\frac{b^2 a}{p}\right) = \left(\frac{a}{p}\right).$$

(c) Réduction modulo p :

$$\left(\frac{a+pc}{p}\right) = \left(\frac{a}{p}\right).$$

(d) Cas particulier : $a = -1$:

$$\left(\frac{-1}{p}\right) = (-1)^{\frac{p-1}{2}} = \begin{cases} 1 & \text{si } p \equiv 1 \mod 4, \\ -1 & \text{si } p \equiv 3 \mod 4. \end{cases}$$

1.7 Témoins des Nombres Composés

Theorem 1.4. [4] (Petit théorème de Fermat) Soit p un nombre premier et a un entier tel que $1 \leq a < p$. Alors :

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

(Par conséquent, pour tout entier a avec $0 \leq a < p$, on a $a^p \equiv a \mod p$.)

Démonstration. Considérons les $p-1$ premiers multiples de a :

$$a, 2a, 3a, \ldots, (p-1)a.$$

Étape 1 : Montrons que ces multiples sont deux à deux distincts modulo p. Supposons que pour $1 \leq i < j \leq p-1$, on ait :

$$ia \equiv ja \mod p.$$

Alors p divise $(j-i)a$. Comme p est premier et ne divise pas a (par hypothèse), p doit diviser $(j-i)$. Mais $0 < j-i < p-1$, ce qui est impossible.
Étape 2 : Les résidus modulo p forment une permutation de $\{1, 2, \ldots, p-1\}$. Puisque les $p-1$ multiples sont distincts modulo p et non nuls modulo p, ils correspondent exactement aux entiers $1, 2, \ldots, p-1$ dans un certain ordre.
Étape 3 : Produit des éléments. En multipliant tous ces multiples :

$$a \cdot 2a \cdot 3a \cdots (p-1)a \equiv 1 \cdot 2 \cdot 3 \cdots (p-1) \pmod p,$$

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

Comme $(p-1)!$ est premier avec p, on peut simplifier :

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

□

Lemme 1.4. [4] Soit p un nombre premier et a un entier tel que $1 \leq a < p$. Si $a^2 \equiv 1 \mod p$, alors $a = 1$ ou $a = p-1$.

Démonstration. L'hypothèse $a^2 \equiv 1 \pmod p$ implique que :

$$p \mid (a^2 - 1) = (a+1)(a-1).$$

Comme p est premier, il doit diviser au moins un des facteurs :

- Si $p \mid (a-1)$, alors $a \equiv 1 \pmod{p}$. Comme $1 \leq a < p$, on a $a = 1$.
- Si $p \mid (a+1)$, alors $a \equiv -1 \pmod{p}$. Comme $1 \leq a < p$, on a $a = p-1$.

□

Lemme 1.5. Soit $n \geq 3$ un nombre impair, tels que $n = u \cdot 2^k$, où u est impair et $k \geq 1$. Soit a un nombre tels que $1 \leq a < n$, si pour tout i, $0 \leq i < k$: $a^u \mod n \neq 1$ et $a^{u \cdot 2^i} \mod n \neq n-1$, alors le nombre n est composé.

Démonstration. se référer à [4]. □

Lemme 1.6. (Critère d'Euler) Soit p un nombre premier impair. Alors :

1. L'ensemble des résidus quadratiques forme un sous-groupe de $\mathbb{Z}_p^*$ de taille $(p-1)/2$.
2. Pour tout $a \in \mathbb{Z}_p^*$, on a :

$$a^{(p-1)/2} \equiv \begin{cases} 1 & \text{si } a \text{ est un résidu quadratique modulo } p \\ -1 & \text{si } a \text{ est un non-résidu modulo } p \end{cases} \mod p.$$

Démonstration. se référer à [4]. □

Lemme 1.7. Pour tout nombre premier impair p et tout $a \in \{1, \ldots, p-1\}$, on a :

$$a^{(p-1)/2} \cdot \left(\frac{a}{p}\right) \equiv 1 \mod p,$$

où $\left(\frac{a}{p}\right)$ désigne le symbole de Legendre.

Démonstration. se référer à [4]. □

1.8 Témoins des Nombres Premiers

Lemme 1.8. [1] Soient $a \in \mathbb{Z}$, $n \in \mathbb{N}$ avec $n \geq 2$, et $\gcd(a, n) = 1$. Alors n est premier si et seulement si :

$$(X+a)^n \equiv X^n + a \mod n.$$

Démonstration. Pour $0 < i < n$, le coefficient de X^i dans $((X+a)^n - (X^n + a))$ est $\binom{n}{i} a^{n-i}$. Supposons que n est premier. Alors $\binom{n}{i} \equiv 0 \mod n$ pour $0 < i < n$, et par conséquent tous les coefficients sont nuls. Supposons que n est composé. Considérons un nombre premier q qui est un facteur de n et soit $q^k \parallel n$. Alors q^k ne divise pas $\binom{n}{q}$ et est premier avec a^{n-q}. Par conséquent, le coefficient de X^q n'est pas nul $\mod n$. Ainsi, $((X+a)^n - (X^n + a))$ n'est pas identiquement nul sur $\mathbb{Z}_n$. □

1.9 Rappel sur la Complexité

Pour des raisons de convention, il est bien de noter que le logarithme utilisé dans ce qui suit est celui de base 2.

1.9.1 Opérations Arithmétiques et binaires

Définition 1.23

Pour une fonction $f : \mathbb{N} \to \mathbb{R}^+$, on définit :

- La notation $O(f)$ comme l'ensemble des fonctions :

$$O(f) = \{g : \mathbb{N} \to \mathbb{R}^+ \mid \exists C > 0 \exists n_0 \forall n \geq n_0 : g(n) \leq C \cdot f(n)\}.$$

- La notation $\Omega(f)$ comme l'ensemble des fonctions :

$$\Omega(f) = \{g : \mathbb{N} \to \mathbb{R}^+ \mid \exists c > 0 \exists n_0 \forall n \geq n_0 : g(n) \geq c \cdot f(n)\}.$$

- La notation $\Theta(f)$ comme l'intersection :

$$\Theta(f) = O(f) \cap \Omega(f).$$

On peut aussi caractériser ces ensembles par :

- $O(f)$ contient les fonctions g telles que $\limsup_{n\to\infty} \frac{g(n)}{f(n)} < \infty$.
- $\Omega(f)$ contient les fonctions g telles que $\liminf_{n\to\infty} \frac{g(n)}{f(n)} > 0$.
- $\Theta(f)$ contient les fonctions g vérifiant les deux conditions simultanément.

Lemme 1.9. (a) Si $g_1(n) = O(f_1(n))$ et $g_2(n) = O(f_2(n))$, alors

$$g_1(n) + g_2(n) = O(\max\{f_1(n), f_2(n)\}).$$

(b) Si $g_1(n) = O(f_1(n))$ et $g_2(n) = O(f_2(n))$, alors

$$g_1(n) \cdot g_2(n) = O(f_1(n) \cdot f_2(n)).$$

Démonstration. se référer à [4]. □

Exemple 1.8

Si $g_1(n) = O((\log n)^2)$ *et* $g_2(n) = O((\log n)(\log\log n))$, *alors*

$$g_1(n) + g_2(n) = O((\log n)^2),$$

et

$$g_1(n) \cdot g_2(n) = O((\log n)^3(\log\log n)).$$

Définition 1.24

Pour une fonction $f : \mathbb{N} \to \mathbb{R}^+$ avec $\lim_{n\to\infty} f(n) = \infty$, on définit :

$$\tilde{O}(f) = \{g : \mathbb{N} \to \mathbb{R}^+ \mid \exists C > 0 \exists n_0 \exists k \forall n \geq n_0 : g(n) \leq C \cdot f(n)\log(f(n))^k\}.$$

On écrit $g(n) = \tilde{O}(f(n))$ lorsque $g \in \tilde{O}(f)$.

Exemple 1.9

Considérons un algorithme de multiplication de deux grands entiers a *et* b *de taille* $n = \log(\max(a,b))$ *bits. Supposons que sa complexité temporelle soit donnée par :*

$$T(n) = 5n(\log n)^3(\log\log n)^2 + 2n^2 \log n.$$

Analyse avec O ***et*** Θ *: Le terme dominant est* $2n^2 \log n$ *pour* n *grand. Ainsi,* $T(n) = \Theta(n^2 \log n)$. *On pourrait aussi écrire* $T(n) = O(n^{2.1})$ *(moins précis).*
Analyse avec $\tilde{O}$ *: On ignore les facteurs logarithmiques poly-logarithmiques. Ici,* $T(n)$ *est dominé par* $n^2 \log n$, *qui est un* $\tilde{O}(n^2)$. *Ainsi,* $T(n) = \tilde{O}(n^2)$.

Propriété 1. Soient n, m des entiers naturels.

(a) L'addition ou la soustraction de n et m nécessite $O(\ell(n) + \ell(m)) = O(\log n + \log m)$ opérations binaires, où $\ell(k)$ désigne la longueur binaire de k.

(b) La multiplication de m et n nécessite $O(\ell(n) \cdot \ell(m)) = O(\log n \cdot \log m)$ opérations binaires.

(c) Le calcul du quotient $n \div m$ et du reste $n \mod m$ nécessite $O((\ell(n) - \ell(m) + 1) \cdot \ell(m))$ opérations binaires.

Lemme 1.10. Le calcul de $a^n \mod m$ nécessite :

- $O(\log n)$ multiplications et divisions de nombres dans $\{0, \ldots, m^2 - 1\}$.
- $O((\log n)(\log m)^2)$ opérations binaires.

Démonstration. se référer à [4]. □

Lemme 1.11. Tester si n est une puissance parfaite ne dépasse pas :

- $O((\log n)^2 \log\log n)$ multiplications de nombres dans $\{1,\ldots,n\}$.
- $O((\log n)^4 \log\log n)$ opérations binaires.

Démonstration. se référer à [4]. □

La complexité d'un algorithme est une estimation du temps de calcul nécessaire à son exécution en fonction de la taille de son entrée. Elle est généralement exprimée à l'aide des notations asymptotiques identifié dans la définition 1.21. Selon la fonction f utilisée dans la notation O, on distingue trois types de complexité :

- **Polynomiale** : lorsque la fonction f est un polynôme par rapport à la taille de l'entrée.
- **Exponentielle** : lorsque la fonction f est une fonction exponentielle par rapport à la taille de l'entrée.
- **Factorielle** : lorsque la fonction f est une fonction factorielle par rapport à la taille de l'entrée.

Chapitre 2

Cryptographie et Tests de Primalité

Ce chapitre s'articule autour de deux axes essentiels : premièrement, une introduction à la cryptographie, un domaine fortement lié à l'utilisation des nombres premiers, illustré par la présentation d'un algorithme de cryptographie largement utilisé, basé sur la sélection de grands nombres premiers, afin de mettre la lumière sur leur importance cruciale. Deuxièmement, une analyse de quelques algorithmes développés pour tester la primalité d'un nombre, dont le but d'introduire l'algorithme A.K.S exposé dans le chapitre ultérieur. Pour plus de détails consulter les ouvrages ([4] et [5])

2.1 Introduction à la Cryptographie

La cryptographie, nommée art du secret, est une discipline scientifique regroupant un ensemble de techniques appliquées aux systèmes cryptographiques, ou *cryptosystèmes*, définissant un cadre d'éléments et de propriétés, conçues pour sécuriser les communications générées entre deux ou plusieurs personnes (même s'ils ne sont jamais rencontrés) dans un espace public, en mettant l'information inaccessible aux tiers indésirables. Elle s'appuie considérablement sur des principes mathématiques ou logiques, dont la complexité permet de garantir une sécurité accrue. Par ailleurs, les opérations effectuées dans un cryptosystème incluent principalement le chiffrement, qui consiste à transformer un message clair destiné à être envoyé par l'expéditeur en un message incompréhensible à l'aide d'une clé secrète, et l'opération inverse appelée déchiffrement, qui permet de retransformer à l'aide d'une clé, le message chiffré et reçu par le destinataire, en sa forme originale. D'autre part, les éléments et les propriétés définis dans tout système de cryptographie peuvent être décrits comme suit :

- Espace des messages, noté par $\mathscr{M}$.
- Espace des messages chiffrés, noté par $\mathscr{C}$.
- Espace des clés, noté par $\mathscr{K}$.
- Fonctions de chiffrement $\{e_k : k \in \mathscr{K}\}$, avec $e_k : \mathscr{M} \to \mathscr{C}$, dépendent généralement des clés utilisées.

- Fonctions de déchiffrement $\{d_k : k \in \mathcal{K}\}$, dépendent généralement des clés utilisées.
- $\forall k_1 \in \mathcal{K}, \exists k_2 \in \mathcal{K}$, tel que $d_{k_2}(e_{k_1}(m)) = m, \forall m \in \mathcal{M}$.

2.1.1 Bref Historique

De nombreuses traces, telles que des manuscrits, des tablettes, et d'autres documents anciens, indiquent que l'utilisation de la cryptographie remonte à des temps anciens [5], en effet, son nom, d'origine grecque ancienne, signifie « écrire et cacher ». Parmi les techniques les plus anciennes de cryptographie, figure le chiffrement de César, réputé dater de l'époque de l'empereur Jules César, elle repose sur l'idée de décaler les lettres alphabétiques d'un certain nombre de positions comme indiqué dans le tableau suivant.

Table 2.1 – Table de chiffrement par substitution simple

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c

La première ligne indique les positions standards des lettres selon l'ordre alphabétique français, tandis que la seconde reflète leurs nouvelles positions après un décalage de trois places vers l'avant. Par conséquent, en appliquant cette astuce, la lettre 'a' est chiffrée en 'x' et déchiffrée en 'd' et de même pour les autres lettres. Il était évident que ce chiffrement était peu sécurisé, car toute personne pouvait décrypter tout message en effectuant les 25 décalages possibles.

Par la suite, de nombreuses techniques ont été élaborées, cependant, elles n'étaient pas considérées comme des cryptosystèmes, car elles souffraient de certaines vulnérabilités, comme c'est le cas pour le chiffrement par transposition, qui consiste à modifier les positions des caractères du texte sans les altérer.

En 1883 Kerckoffs von Nieuwenhoff a pu établir les règles fondamentales à respecter dans la conception d'un système cryptographique, elles se répartissent comme suit :

- Le système doit être mathématiquement ou matériellement indéchiffrable.
- Il faut qu'il n'exige pas le secret, et qu'il puisse sans inconvénient tomber entre les mains d'un tiers.
- La clé doit être facilement communicable et mémorisable sans support écrit, et modifiable à volonté.
- Le système doit être compatible avec les communications télégraphiques.
- Il doit être portable et utilisable par une seule personne.
- Enfin, il doit être facile à utiliser, sans nécessiter de procédures complexes ou de longues instructions.

2.1.2 Cryptographie moderne

À l'instar de la révolution scientifique reconnue en informatique, les intéressés ont commencé à chercher à développer des méthodes plus efficaces et exécutables, permettant de concevoir un bon cryptosystème selon les critères de Kerckoffs, assurant une sécurité plus élevée, en reliant la décryption des messages par des tiers à des problèmes difficiles à résoudre. Ces méthodes constituent actuellement la cryptographie moderne, elles sont largement utilisées, et dans divers domaines en raison de l'efficacité et la sécurité qu'elles présentent.

Les méthodes de la cryptographie moderne se ressemblent ou se différencient selon le type de chiffrement adopté, en voici les plus fréquents :

- **Chiffrement par blocs** : qui consiste à diviser le message en blocs, et à chiffrer chacun individuellement.
- **Chiffrement en flux** : dans lequel, le message est chiffré en continu, sans nécessiter de découpage préalable.

Dans les systèmes cryptographiques dont la sécurité est basée sur la résolution de problèmes mathématiques complexes, le message à chiffrer est souvent représenté par un nombre, en associant à chaque caractère un nombre binaire de huit bits. Par exemple : `a=00000000, b=00000001,...` Cela comprend tous types de caractères : lettres, chiffres, symboles, etc. Mêmes les lettres majuscules sont prises en compte, et pour les différencier, on leur attribue des nombres binaires distincts.

Enfin, il est nécessaire de noter que les systèmes cryptographiques de la cryptographie moderne sont souvent classés en deux grandes catégories : symétriques et asymétriques.

2.2 Cryptographie symétrique

Supposons que Bob et Alice veuillent communiquer en toute sécurité. Avec les systèmes cryptographiques symétriques, ils doivent partager une seule clé secrète k. En utilisant cette clé secrète, ils peuvent tous deux crypter et décrypter les messages, de sorte que Bob et Alice ont les mêmes connaissances et capacités (ou symétriques). C'est pourquoi les chiffrements de ce type sont connus sous le nom de chiffrements symétriques.

Mathématiquement, un chiffrement symétrique utilise une clé k choisie dans un espace de clés possibles $\mathscr{K}$ pour chiffrer un message en clair m choisi dans un espace de messages possibles $\mathscr{M}$, et le résultat du processus de chiffrement est un texte chiffré c appartenant à un espace de textes chiffrés possibles $\mathscr{C}$.

2.3 Cryptographie asymétrique

La cryptographie asymétrique, également appelée cryptographie à clé publique, se distingue radicalement de la cryptographie symétrique. Pour illustrer cette différence, imaginez la cryptographie symétrique comme un coffre-fort : seules les personnes possédant la clé peuvent y accéder. En revanche, la cryptographie asymétrique fonctionne comme une boîte aux lettres. N'importe qui peut y déposer un message, mais seule la personne possédant la clé privée peut l'ouvrir pour en consulter le contenu. Dans cette analogie, la boîte aux lettres représente la clé publique, accessible à tous, tandis que la clé pour l'ouvrir correspond à la clé privée, gardée secrète. Ainsi, la cryptographie asymétrique repose sur une paire de clés : une clé publique pour chiffrer et une clé privée pour déchiffrer.

2.3.1 Cryptosystème RSA

Le cryptage RSA est une méthode cryptographique, dont le nom dérive des lettres initiales des noms de ses concepteurs (Rivest, Shamir et Adleman), constitue un cryptosystème asymétrique reposant sur l'utilisation d'une clé publique pour le chiffrement et d'une clé privée, détenue par le destinataire, pour le déchiffrement. La méthode est désormais adoptée dans divers domaines liés au chiffrement des informations, tels que les transactions en ligne, la sécurisation des courriers électroniques, ainsi que l'authentification et les signatures numériques.

Dans ce système, les opérations sont mises en œuvre au sein d'un ensemble des entiers, en particulier, cela inclut la génération de nombres premiers et la création des clés en appliquant la congruence. Plus précisément, le processus suit les étapes suivantes [5] :

Création des clés

Un couple de clés RSA se génère ainsi :

1. Générer deux très grands nombres premiers p et q distincts.
2. Calculer $n = p \times q$ et $\phi(n) = (p-1)(q-1)$.
3. Choisir un entier aléatoire e entre 1 et $\phi(n)$, tel que $\gcd(e, \phi(n)) = 1$.
4. Utiliser l'algorithme d'Euclide pour calculer l'entier d (l'inverse modulaire de e modulo $\phi(n)$) compris entre 1 et $\phi(n)$ tel que :

$$e \times d \equiv 1 \mod \phi(n).$$

5. La clé publique est le couple (e, n).

6. La clé privée est le couple (d, n).

Chiffrement et déchiffrement

Pour chiffrer un message M, il faut d'abord le diviser en blocs de chiffres plus petits que n. Si p et q ont tous deux 100 chiffres, alors n aura un peu moins de 200 chiffres décimaux, et chaque bloc de chiffres devra avoir un peu moins de 200 chiffres.

- **Fonction de chiffrement :**
  - Entrée : Un message clair M, une clé publique $\{e, n\}$.
  - Sortie : Un message chiffré C.
  - Calcul :

$$C \equiv M^e \mod n.$$

- **Fonction de déchiffrement :**
  - Entrée : Un message chiffré C, une clé privée $\{d, n\}$.
  - Sortie : Le message clair M.
  - Calcul :

$$M \equiv C^d \mod n.$$

Exemple 2.1

Alice souhaite envoyer un message chiffré à Bob en utilisant le cryptosystème RSA.

1. Génération des clés par Bob

- ***Étape 1*** *: Choix des nombres premiers*
  Bob sélectionne deux nombres premiers distincts :

$$p = 47, \quad et \quad q = 71.$$

- ***Étape 2*** *: Calcul du module*

$$n = p \times q = 47 \times 71 = 3337.$$

- ***Étape 3*** *: Calcul de l'indicatrice d'Euler*

$$\phi(n) = (p-1)(q-1) = 46 \times 70 = 3220.$$

- ***Étape 4*** *: Choix de l'exposant public*
  Bob choisit $e = 79$ *qui vérifie :*

$$\gcd(79, 3220) = 1.$$

- ***Étape 5*** *: Calcul de l'exposant privé*
  Bob doit trouver d tel que :

$$79 \times d \equiv 1 \mod 3220.$$

Calcul détaillé avec l'algorithme d'Euclide étendu :

$$\begin{aligned} 3220 &= 40 \times 79 + 60, \\ 79 &= 1 \times 60 + 19, \\ 60 &= 3 \times 19 + 3, \\ 19 &= 6 \times 3 + 1, \\ 3 &= 3 \times 1 + 0. \end{aligned}$$

En remontant les calculs :

$$\begin{aligned} 1 &= 19 - 6 \times 3 \\ &= 19 - 6 \times (60 - 3 \times 19) \\ &= 19 \times 19 - 6 \times 60 \\ &= 19 \times (79 - 1 \times 60) - 6 \times 60 \\ &= 19 \times 79 - 25 \times 60 \\ &= 19 \times 79 - 25 \times (3220 - 40 \times 79) \\ &= 1019 \times 79 - 25 \times 3220. \end{aligned}$$

On obtient donc :

$$d = 1019.$$

Clés finales *:*

- *Clé publique :* $(n = 3337, e = 79)$.
- *Clé privée :* $d = 1019$.

2. Chiffrement par Alice *Message original :*

$$688\ 232\ 687\ 966\ 668\ 003.$$

Calcul du chiffrement : Pour chaque bloc M, Alice calcule :

$$C \equiv M^e \mod n.$$

Pour $M_1 = 688$:

$$\begin{aligned}
688^2 &\equiv 688 \times 688 \equiv 473344 \equiv 2820 \mod 3337,\\
688^4 &\equiv 2820^2 \equiv 7952400 \equiv 3157 \mod 3337,\\
688^8 &\equiv 3157^2 \equiv 9966649 \equiv 2521 \mod 3337,\\
688^{16} &\equiv 2521^2 \equiv 6355441 \equiv 1234 \mod 3337,\\
688^{32} &\equiv 1234^2 \equiv 1522756 \equiv 296 \mod 3337,\\
688^{64} &\equiv 296^2 \equiv 87616 \equiv 2224 \mod 3337,\\
688^{79} &= 688^{64} \times 688^8 \times 688^4 \times 688^2 \times 688^1,\\
&\equiv 2224 \times 2521 \times 3157 \times 2820 \times 688 \mod 3337,\\
&\equiv 1570 \mod 3337.
\end{aligned}$$

Le message chiffré est :

1570 2756 2091 2276 2423 158.

3. Déchiffrement par Bob *Calcul du déchiffrement : Pour chaque bloc C, Bob calcule :*

$$M \equiv C^d \mod n.$$

Pour $C_1 = 1570$:

$$1570^{1019} \equiv 688 \mod 3337.$$

D'où le message original est retrouvé :

688 232 687 966 668 003.

Principes du fonctionnement de RSA

Le principe des étapes de la méthode RSA découle de la proposition suivante :

Proposition 2.1: [5]

Soient p et q deux nombres premiers distincts, et soit $e \geq 1$ un entier vérifiant $\gcd(e, \phi(n)) = 1$. Alors la congruence

$$x^e \equiv c \mod n,$$

a pour unique solution

$$x \equiv c^d \mod n,$$

où d est l'inverse de e modulo $\phi(n)$.

Démonstration. Si $\gcd(c,n)=1$, la preuve utilise la formule d'Euler. La congruence $de \equiv 1 \pmod{\phi(n)}$ signifie qu'il existe un entier k tel que $de = 1 + k\phi(n)$.

Nous vérifions que c^d est une solution :

$$\begin{aligned}(c^d)^e &\equiv c^{de} \mod n \\ &\equiv c^{1+k\phi(n)} \mod n \\ &\equiv c \cdot (c^{\phi(n)})^k \mod n \\ &\equiv c \cdot 1^k \mod n \quad \text{le théorème d'Euler (Théorème 1.3)} \\ &\equiv c \mod n.\end{aligned}$$

L'unicité se démontre en supposant deux solutions et en montrant qu'elles sont congrues modulo n. □

Remarque 2.1. L'existence d'une solution unique pour la congruence ($x = c^d \mod n$) constitue l'idée principale sur laquelle repose l'exactitude de la méthode.

Sécurité de RSA

La sécurité du système RSA repose sur la difficulté pratique de factoriser de grands nombres entiers. La connaissance de la factorisation de n (i.e., connaître p et q) permet de calculer $\phi(n)$ et donc de déterminer la clé privée d à partir de la clé publique e.

Les attaques potentielles incluent :

- La factorisation de n.
- Le calcul de $\phi(n)$ sans factoriser n.

La taille recommandée pour n est actuellement d'au moins 2048 bits pour une sécurité raisonnable.

2.4 Tests de primalité

La section précédente a clairement illustré le rôle essentiel des nombres premiers dans les algorithmes de cryptographie, notamment dans l'algorithme RSA. Ce dernier requiert premièrement la sélection de deux grands nombres premiers, une opération qui s'avère exigeante avec la méthode de factorisation classique. On va examiner dans cette section quelques algorithmes fréquemment utilisés, développés pour la recherche de tels nombres.

2.4.1 Test de Fermat

Le test de Fermat est un algorithme probabiliste permettant de vérifier si un nombre n est probablement premier. Il repose sur le théorème de Fermat (Théorème 1.4), il est

rapide mais présente des limites, notamment avec les nombres de Carmichael.

Description de l'algorithme

Les étapes suivantes expliquent comment déterminer si le nombre n est un nombre premier ou non à l'aide de l'algorithme de Fermat.

Algorithme 2 Test de Fermat

Entrée: Un entier impair $n \geq 3$ à tester.

1: Laisser a être choisi au hasard parmi $\{2, \ldots, n-2\}$;
2: si $a^{n-1} \mod n \neq 1$
3: alors retourner 1 ;
4: sinon retourner 0 ;

Exemple 2.2

Testons $n = 340$ *avec* $a = 2$ *:*

$$2^{339} \equiv 8 \mod 340.$$

Par conséquent, le nombre 340*est composé. Pourtant,* $341 = 11 \times 31$ *est composé.*

Principe de fonctionnement

L'algorithme de Fermat s'appuie sur le théorème de Fermat (Théorème 1.4). Pour un entier a, il vérifie si la condition $a^{n-1} \mod n = 1$ est satisfaite. Si ce n'est pas le cas, l'algorithme confirme avec certitude que n est un nombre composé. Dans le cas contraire, il indique que n est probablement premier, avec une probabilité moins de $1/2$, comme le démontre le théorème suivant.

Theorem 2.1. Si $n \geq 3$ est un nombre composé et impair, alors le test de Fermat retourne 1 avec une probabilité égal à $\frac{1}{2}$.

Démonstration. se référer à [4]. □

Analyse de complexité

Le calcul coûteux est l'exponentiation modulaire $a^{n-1} \mod n$: selon le lemme 1.10, cela nécessite $O(\log n)$ opérations arithmétiques et $O((\log n)^3)$ opérations binaires. De plus, il est clair que si l'algorithme renvoie 1, c'est qu'il a détecté ce qui garantit que n est composé.

Efficacité et Limites

Le test de Fermat est un algorithme simple et rapide pour vérifier la primalité d'un nombre. Son principal avantage est sa facilité d'implémentation et son efficacité pour des entiers de taille modérée. Cependant, il présente une faille majeure : il est trompé par les nombres de Carmichael (des nombres composés qui satisfont le test pour toutes les bases premières avec eux) conduisant à des faux positifs. Ainsi, bien qu'il soit rapide, sa fiabilité est limitée, ce qui le rend peu adapté aux applications cryptographiques exigeantes.

2.4.2 Test de Miller-Rabin

Le test de Miller-Rabin est un test de primalité qui repose sur une approche probabiliste. Il permet de déterminer si un nombre donné est probablement premier ou certainement composé. Contrairement à d'autres tests, il est fiable même pour les nombres de Carmichael, ce qui le rend particulièrement utile.

Description de l'algorithme

Les étapes de l'algorithme de Miller-Rabin sont détaillées ci-après :

Algorithme 3 Test de Miller-Rabin

Entrée: Un entier impair $n \geq 3$ à tester.

1: Trouver u impair et k tels que $n-1 = u \cdot 2^k$;
2: Choisir aléatoirement $a \in \{2, \ldots, n-2\}$;
3: Calculer $b \leftarrow a^u \mod n$;
4: Si $b \in \{1, n-1\}$ alors retourner 0 ;
5: Répéter $k-1$ fois :
6: (a). $b \leftarrow b^2 \mod n$;
7: (b). Si $b = n-1$ alors retourner 0 ;
8: (c). Si $b = 1$ alors retourner 1 ;
9: Retourner 1 ;

Exemple 2.3

Soit $n = 221$, *alors* $n-1 = 220 = 2^2 \times 55$, *d'où* $k = 2$ *et* $u = 55$.

On teste avec $a = 5$:

$$x_0 \equiv 5^{55} \equiv 112 \mod 221,$$
$$x_1 \equiv 112^2 \equiv 168 \mod 221,$$
$$x_2 \equiv 168^2 \equiv 157 \mod 221,$$
$$x_3 \equiv 157^2 \equiv 118 \mod 221,$$
$$x_4 \equiv 118^2 \equiv 1 \mod 221 \text{ (\textit{égal à} 1)}.$$

On s'arrête : 221 est composé. Sa factorisation est :

$$221 = 13 \times 17.$$

Principe de fonctionnement

L'algorithme cherche à trouver une racine quadratique non triviale de a modulo n. Si une telle racine est détectée, cela prouve que n est composé, et l'algorithme retourne alors 1. En revanche, si après l application, il retourne toujours 0, on constate que n est premier avec une probabilité plus que $1-4^{-l}$, comme détaillé dans le lemme et la proposition ci-après.

Proposition 2.2

Lorsqu'on applique l'algorithme de Miller-Rabin l fois à un nombre d'entrée n, cela nécessite $O(l \log n)$ opérations arithmétiques et $O(l(\log n)^3)$ opérations binaires (méthodes simples) ou $\tilde{O}(l(\log n)^2)$ opérations binaires (meilleures méthodes). Si n est un nombre premier, la sortie est 0. Si n est composé, la probabilité que la sortie 0 soit donnée est inférieure à 4^{-l}.

Démonstration. se référer à [4]. □

Lemme 2.1. Si l'algorithme de Miller-Rabin retourne 1 alors n est composé.

Démonstration. Une conséquence directe du lemme 1.5 (voir [4]) □

Efficacité et Limites

Plus sophistiqué que le test de Fermat, le test de Miller-Rabin améliore significativement la fiabilité en évitant les écueils des nombres de Carmichael. Il utilise plusieurs bases et une analyse probabiliste rigoureuse, permettant d'atteindre une très haute précision en augmentant le nombre d'itérations. Son principal avantage est son excellente balance entre vitesse et fiabilité, ce qui en fait le choix privilégié dans de nombreuses applications cryptographiques. Toutefois, il reste techniquement probabiliste (bien que les erreurs soient extrêmement rares avec un choix judicieux de bases) et légèrement plus coûteux en calculs que le test de Fermat.

2.4.3 Test de Solovay-Strassen

Le test de Solovay-Strassen utilise le critère d'Euler et les propriétés du symbole de Jacobi. C'est un test probabiliste.

Description de l'algorithme

Voici les étapes de l'algorithme de Solovay-Strassen :

Algorithme 4 Test de Solovay-Strassen

Entrée: Un entier impair $n \geq 3$ à tester.

1: Choisir aléatoirement un entier $a \in \{2, \ldots, n-2\}$;
2: Calculer $x \equiv a^{(n-1)/2} \mod n$;
3: Calculer le symbole de Jacobi $y = \left(\frac{a}{n}\right)$;
4: Si $x \equiv 0$ ou $x \not\equiv y \pmod{n}$ Alors Retourner 1 ;
5: Sinon Retourner 0 ;

Exemple 2.4

Testons le nombre $n = 561$ *avec* $a = 2$ *:*

$$2^{280} \equiv 67 \mod 561.$$

$$\left(\frac{2}{561}\right) = 1 \ (car\ 561 \equiv 1 \mod 8).$$

$$67 \not\equiv 1 \mod 561 \Rightarrow composé.$$

Principe de fonctionnement

S'appuyant sur le lemme 1.7, le test de Solovay-Strassen examine si $\left(\frac{a}{n}\right)$ est l'inverse multiplicatif modulo n de $a^{(n-1)/2}$ pour un entier a inférieur à $n-1$. Une non correspondance indique que n est composé ; dans le cas contraire, n est considéré comme probablement premier (voir la proposition qui suit).

Proposition 2.3

L'algorithme de Solovay, lorsqu'il est appliqué à un nombre d'entrée n, nécessite $O(\log n)$ opérations arithmétiques sur des nombres plus petits que n^2, ce qui équivaut à $O((\log n)^3)$ opérations binaires (méthodes naïves) ou $\widetilde{O}((\log n)^2)$ opérations binaires (meilleures méthodes).
Si n est un nombre premier, la sortie est 0. Si n est composé, la probabilité que la sortie soit 0 est inférieure à $\frac{1}{2}$.

Démonstration. voir [4]. □

Efficacité et Limites

Basé sur le symbole de Jacobi et les propriétés de réciprocité quadratique, le test de Solovay-Strassen offre une approche mathématiquement élégante pour la primalité probabiliste. Il est plus fiable que le test de Fermat et permet de détecter efficacement

les nombres composés. Cependant, il est plus compliqué à mettre en œuvre en pratique que le test de Miller-Rabin, car il implique des calculs un peu compliquées.

Chapitre 3

Test de primalité A.K.S

Le présent chapitre constitue l'élément essentiel de ce travail. Il fournit une description complète du test AKS, renommé pour être la seule méthode capable de déterminer la primalité d'un nombre de façon déterministe. De surcroît, une technique est proposée dans le but d'améliorer sa performance.

3.1 Algorithme

Les étapes clés de l'algorithme A.K.S sont exposées dans le tableau suivant :

Algorithme 5 Algorithme AKS

Entrée: $n \geq 2$ entier.
Sortie: 1 si n est premier, 0 sinon.

1: Vérifier si n est une puissance parfaite. Si oui, retourner 0 ;
2: Trouver le plus petit r tel que $\text{ord}_r(n) > (\log n)^2$;
3: Vérifier si n est premier avec tout entier inférieur à r. Sinon, retourner 0 ;
4: Si $n < r$ alors retourner 1 ;
5: **pour** $a = 2$ à $\lfloor \sqrt{\phi(r)} \log n \rfloor$ **faire**
6: **si** $(X+a)^n \not\equiv X^n + a \mod (X^r - 1, n)$ **alors**
7: Retourner 0 ;
8: **fin si**
9: **fin pour**
10: Retourner 1 ;

Exemple 3.1

Considérons l'application de l'algorithme AKS au nombre $n = 21$.

Étapes clés pour $n = 21$ ***:***

Étape 1 : *Vérification des puissances parfaites*

$21 = 3 \times 7 \Rightarrow$ *Pas d'une puissance parfaite.*

Étape 2 : *Recherche de* r *minimal avec* $o_r(21) > \log^2(21) \approx 19.27$

Trouvé $r = 23$ *avec* $o_{23}(21) = 22$.

Étape 3 : *Test de divisibilité pour* $a = 2$ *à* 23
Détection que $\gcd(3,21) = 3$.
composé
Détails des calculs :

- *Calcul de l'ordre multiplicatif :*

$$\begin{aligned} 21^1 &\equiv 21 \mod 23, \\ 21^2 &\equiv 4 \mod 23, \\ 21^{11} &\equiv 12 \mod 23, \\ 21^{22} &\equiv 1 \mod 23. \end{aligned}$$

- *Test de divisibilité :*

$$\gcd(3,21) = 3 \quad (\textit{détecté dès } a = 3).$$
$$1 < 3 < 21 \quad \Rightarrow \textit{Critère satisfait.}$$

Résultat : *L'algorithme termine à l'étape 3 et retourne* composé.

3.1.1 Principe de fonctionnement

L'algorithme AKS repose sur l'idée exposée dans le lemme 1.8, et implique des calculs sur l'anneau $\mathbb{Z}_n[X]/(X^r-1)$, où r est le plus petit entier tel que $\mathrm{ord}_n(r) > \log^2(n)$. Plus précisément, après avoir vérifié que n n'est pas une puissance parfaite, l'algorithme détermine la valeur de r. Ensuite, il compare $(X+a)^n \bmod (X^r-1, n)$ avec $X^n+a \bmod (X^r-1, n)$ pour tout entier a dans l'intervalle $\left[1, \lfloor\sqrt{\phi(r)}\log(n)\rfloor\right]$. Si une inégalité est détectée pour un de ces entiers a, alors n est composé. Dans le cas contraire, l'algorithme conclut que n est premier. L'existence d'un tel r est toujours garantie.

L'algorithme retourne que n est premier si et seulement si n est un nombre premier, comme démontré par les lemmes subséquents.

Lemme 3.1. Il existe un entier $r \leq \max\{3, \lfloor\log^5 n\rfloor\}$ tel que :

$$\mathrm{ord}_r(n) > \log_2^2 n.$$

Démonstration. voir [1] □

Lemme 3.2. [1] Si n est premier, l'algorithme retourne premier.

Démonstration. Si n est premier, alors les étapes 1 et 3 ne peuvent pas retourner composé. D'après le Lemme 1.8, la boucle pour ne peut pas non plus retourner composé. Par conséquent, l'algorithme identifiera n comme premier soit à l'étape 4, soit à l'étape 10. □

Lemme 3.3. Soit n un entier supérieur à 2. Si pour tout entier a premier avec n dans l'intervalle $\left[1, \lfloor\sqrt{\phi(r)}\log(n)\rfloor\right]$:

$$(X+a)^n \equiv X^n + a \pmod{X^r - 1, n},$$

alors n est premier.

La démonstration du lemme s'appuie sur les concepts des nombres introceptifs et des polynômes cyclotomiques. Pour une explication plus détaillée, consulter [1].

3.2 Analyse de complexité temporelle et améliorations

Theorem 3.1. La complexité temporelle asymptotique de l'algorithme est $\tilde{O}(\log^{15/2} n)$.

Démonstration. se référer à [1] □

3.2.1 Amélioration de la Complexité

Dans ce qui précède, on a présenté la version classique du test AKS, celle qui provient directement de l'article original où il a été exposé [1]. Cependant, il est important de savoir que de nombreuses recherches ont été dédiées à la réduction de la complexité de cet algorithme, en se basant sur d'autres concepts et résultats mathématiques. À titre d'exemple, [3] propose une variante de l'algorithme AKS pour une classe de nombres spécifique, montrant une complexité égale à $O((\log n)^4)$. Par ailleurs, [6] introduit une modification pour l'algorithme AKS qui peut générer une complexité de $O(\log(n)^6)$. D'autre part, il est également possible d'améliorer davantage l'algorithme en adoptant les idées et le raisonnement illustrées dans ce qui suit. Dans la deuxième étape de l'algorithme visant à trouver le plus petit entier r tel que $ord_r(n) > \log^2(n)$, la méthode initialement envisagée est une approche directe. Elle consiste à tester séquentiellement les entiers r dans l'intervalle $[\log^2(n), \max\{3, \lfloor\log^5 n\rfloor\}]$. Pour chaque entier r, la vérification de $n \mod r$ implique une division, ce qui, d'après la propriété 1, entraîne une complexité d'environ $O(\log n \log r)$.

Cependant, cette complexité peut être réduite en adoptant une technique de division optimisée. Au lieu de recalculer $n \mod r+1$ de manière indépendante, nous pouvons exploiter les résultats de la division précédente par r.

Plus précisément, si nous avons déjà calculé le quotient q et le reste s de la division de n par r (c'est-à-dire $n = rq+s$), nous pouvons dériver les valeurs pour $r+1$ comme suit : Puisque $n = rq+s$, nous pouvons réécrire l'expression en fonction de $r+1$:

$$n = (r+1-1)q+s = (r+1)q-q+s = (r+1)q-(q-s).$$

À partir de cette relation, nous pouvons déterminer le nouveau reste et le nouveau quotient pour $r+1$:

- Le reste de n par $r+1$ est égal : à $n \pmod{r+1} = r+1+((q-s) \pmod{r+1})$ si $(q-s) \geq (r+1)$, à $r+1+(q-s)$ si $0 < (q-s) < (r+1)$, et à $(q-s)$ si $(q-s) < 0$.
- Le quotient de n par $r+1$ est égal : à $n \operatorname{div}(r+1) = q-((q-s)\operatorname{div}(r+1))$ si $(q-s) \geq (r+1)$, à $(q-1)$ si $0 < (q-s) < (r+1)$, et à q si $(q-s) < 0$.

Cette approche permet de ramener la complexité de chaque étape de test à $O((\log n - \log(r+1))\log(r+1))$, ce qui est une amélioration notable par rapport à la complexité $O(\log n \log(r+1))$ d'une division standard.
Dans la troisième étape de l'algorithme AKS, l'objectif est de vérifier si la congruence polynomiale $(X+a)^n \equiv X^n + a \mod (X^r-1, n)$ est satisfaite pour chaque entier a appartenant à l'intervalle $[1, \lfloor\sqrt{\phi(r)}\log(n)\rfloor]$. On peut simplifier et accélérer significativement cette vérification. Il est clair que la condition est remplie pour un entier a si et seulement si :

$$(X+a)^n \mod (X^r-1, n) = X^{n \bmod r} + a.$$

En examinant les étapes de la division euclidienne de $(X+a)^n$ par (X^r-1), on constate que :

$$\begin{aligned}(X+a)^n \mod (X^r-1, n) = &\left(C_n^1 a + C_n^{r+1} a^{r+1} + \ldots + C_n^{kr+1} a^{kr+1}\right) \mod nX + \\ &\left(C_n^2 a^2 + \ldots + C_n^{kr+2} a^{kr+2}\right) \mod nX^2 + \ldots \\ &\left(C_n^s a^s + \ldots + C_n^{kr+s} a^{kr+s}\right) \mod nX^s + \ldots \\ &\left(C_n^{s+1} a^{s+1} + \ldots + C_n^{(k-1)r+s+1} a^{(k-1)r+s+1}\right) \mod nX^{s+1} + \ldots \\ &\left(C_n^{r-1} a^{r-1} + \ldots + C_n^{(k-1)r+r-1} a^{(k-1)r+r-1}\right) \mod nX^{r-1}.\end{aligned}$$

La dernière formule montre clairement comment déterminer les coefficients du polynôme résiduel. Par conséquent, il ne reste plus qu'à vérifier s'ils sont nuls pour confirmer que la condition est satisfaite pour a. Pour ce faire, on peut utiliser les résultats établis dans la proposition 1.9, en commençant par une faible puissance et en augmentant et simplifiant progressivement jusqu'à atteindre la puissance n (Comme

c'est le cas dans le calcul d'exponentiation modulaire).

3.3 Application

Pour illustrer les semi-méthodes dont nous avons discuté, voyons comment leurs différentes étapes s'appliquent dans le cadre du test de primalité AKS pour le nombre $n = 461$, déjà connu comme un nombre premier

- **Vérifier si n est une puissance parfaite**

  Le nombre $n = 461$ ne constitue pas une puissance parfaite, car : $\ln(461) \simeq 6.13$ et la division de $\ln(461)$ par $\ln(2)$, $\ln(3)$, ou $\ln(5)$ ne donne pas un entier.

- **Rechercher le r approprié**

  On cherche à déterminer l'entier r tel que $\text{ord}_r(461) > \log_2(461) \simeq 78.29$. Évidemment, $r \geq 80$. Pour commencer, déterminons $\text{ord}_{80}(461)$. En divisant 461 par 80, on obtient $461 = 80 \cdot 5 + 61$, d'où

  $$\begin{aligned} 461 &\equiv 61 \mod 80, \\ 461^2 &\equiv 41 \mod 80, \\ 461^3 &\equiv 21 \mod 80, \\ 461^4 &\equiv 1 \mod 80. \end{aligned}$$

  Ensuite, on passe au nombre suivant 81. Au lieu de diviser 461 directement par 81, on reprend le quotient précédent 5, et on le divise par 81, on obtient 0, par conséquent, $461 = 81 \cdot 5 + 56$. Et on continue de la même manière.

- **Vérifier si n est premier avec tout nombre inférieur à r**

  On applique l'algorithme d'Euclide.

- **Vérifier si** $n \leq r$

  On passe à la dernière vérification.

- **Congruence polynomiale**

  Supposons que $r = 83$. Pour illustrer cela, examinons ce qui se passe avec le coefficient de X lorsque $a = 2$. D'après la formule discutée précédemment, ce coefficient est égal à

  $$C_{461}^{1} 2 + C_{461}^{84} 2^{84} + C_{461}^{167} 2^{167} + C_{461}^{250} 2^{250} + C_{461}^{333} 2^{333} + C_{461}^{416} 2^{416}.$$

Conclusion

Les nombres premiers jouent un rôle essentiel en mathématiques grâce à leurs propriétés intrigantes, qui en font un sujet d'étude fondamental. Leurs applications s'étendent à plusieurs domaines, notamment en cryptographie, où leur caractéristique de n'être divisibles que par un et par eux-mêmes a été exploitée pour développer des algorithmes efficaces offrant une sécurité accrue.

Dans des méthodes cryptographiques comme le RSA, la capacité à déterminer de très grands nombres premiers est cruciale. Ce travail a présenté diverses méthodes utilisées pour tester la primalité d'un nombre, incluant celles de Fermat, Miller-Rabin, et Solovay-Strassen.

Nous nous sommes particulièrement focalisés sur l'algorithme AKS, car c'est le seul test de primalité reconnu comme déterministe et de complexité polynomiale. Après l'avoir étudié en détail, nous avons proposé une technique visant à réduire les calculs, laquelle pourrait être appliquée aux différentes variantes de la méthode AKS.

Bibliographie

[1] Agrawal, M., Kayal, N., and Saxena, N. (2004) *Primes is in P*. Annals of Mathematics, 781–793.

[2] Barrow-Green, J., Gray, J., Wilson, R. (2019). *The History of Mathematics : A Source-Based Approach : Volume 1*. American Mathematical Society.

[3] Berrizbeitia, P. (2005). *Sharpening "Primes is in P" for a large family of numbers*. Mathematics of Computation **74**, 252, 2043–2059.

[4] Dietzfelbinger, M. (2004). *Primality Testing in Polynomial Time : From Randomized Algorithms to "PRIMES Is in P"*. Lecture Notes in Computer Science. Springer Berlin Heidelberg.

[5] Hoffstein, J., Pipher, J., Silverman, J. (2008). *An Introduction to Mathematical Cryptography*. Undergraduate Texts in Mathematics. Springer New York.

[6] Lenstra, H. W., Pomerance, C. (2019). *Primality testing with Gaussian periods*. Journal of the European Mathematical Society **21(4)**, 1229–1269.