N d'ordre : 184/2024-C/MT

RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE
MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET
DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITÉ DES SCIENCES ET DE LA TECHNOLOGIE
HOUARI BOUMEDIENE
FACULTÉ DE MATHÉMATIQUES

THÈSE DE DOCTORAT

Présentée pour l'obtention du grade de Docteur
En MATHÉMATIQUES
Spécialité : Mathématiques et Applications

Par
Rachid CHERGUI

Sujet

Les propriétés algébriques des suites récurrentes linéaires.

Soutenue publiquement le : 13/06/2024 à 10h00 , devant le jury composé de :

M. REZAOUI Mohammed Salem	Maître de conférences A	à l'USTHB	Président
M. BEHLOUL Djilali	Professeur	à l'USTHB	Directeur de thèse
M. SELT Omar	Professeur	à Univ. Djelfa	Examinateur
M. HAMTAT Abdelkader	Maître de conférences A	à CU-Tipaza	Examinateur

Remerciements

Louange à Allah, Clément et
Miséricordieux, sans lui rien de tout
cela n'aurait pu être

Je remercie très chaleureusement mon directeur de thèse, Professeur BEHLOUL Djilali, qui a accepté de prendre la direction de cette thèse, transformant les difficultés rencontrées en une expérience enrichissante. Je lui suis également reconnaissante de m'avoir assuré un encadrement rigoureux tout au long de ces années. Monsieur BEHLOUL a su diriger mes travaux avec beaucoup de disponibilité, de tact et d'intérêt, ses qualités scientifiques, pédagogiques et surtout humaines.

Mes vifs remerciements sont également adressés au professeur REZAOUI Mohammed Salem qui a eu l'amabilité de bien vouloir présider le jury. Je remercie très chaleureusement les professeurs : SELT Omar et HAMTAT Abdelkader qui ont accepté sans réserve aucune, d'évaluer cette thèse à sa juste valeur, et de me faire part de leur remarques sûrement pertinentes qui, contribueront, sans nul doute, au perfectionnement du présent travail.

Je tiens à exprimer mes plus sincères remerciements à mes parents, mon épouse et mon trésor MOHAMED YANIS pour leurs encouragements et contributions d'une manière ou d'une autre, à la réalisation de cette thèse de doctorat.

Je remercie également tous les membres du laboratoire Algèbre et Théorie des Nombres de l'USTHB pour la bonne humeur et la convivialité qui y régnaient. J'exprime ma profonde gratitude à tous mes enseignants, à la responsable de la bibliothèque Taous OULMANE, à tous les employés, et mes amis.

Table des matières

Table des figures

Liste des tableaux

Introduction

Depuis la découverte des suites de Fibonacci et de Lucas, les chercheurs ne cessent de découvrir de nouvelles applications de ces suites. Cette Thèse donne une application des suites de Fibonacci et de Lucas dans la théorie du codage, utilisée essentiellement dans la compression sans perte de données. On distinguera deux codes basés sur les nombres de Fibonacci : le premier défini par Apostolico et Frankel en 1987 dans [1], nommé le code universel de Fibonacci qui repose essentiellement sur le théorème de Zeckendorf, ce dernier propose une représentation des entiers en utilisant les nombres de Fibonacci, suivit par un préfixe. Le deuxième propose par Stankhov en 1999 dans [29], qui utilise la Qp-matrice de Fibonacci. Dans le premier chapitre de cette Thèse, on va commencer par donner quelques définitions liées à la théorie des codes et de la cryptographie dont on va décrire quelques propriétés essentielles du codage puis on va citer quelques méthodes de compression de données et on terminera avec un exemple sur un Cryptosystème basé sur les codes linéaires (Mc Eliece). Dans le deuxième chapitre, on va donner quelques définitions et propriétés liées aux suites récurrentes linéaires (Fibonacci et Lucas) qui seront utilisées dans la suite de la Thèse. Nous passons après à la présentation du Théorème de Zeckendorf avec ces deux versions (Fibonacci et Lucas) et à sa généralisation dans le chapitre trois, dans le chapitre quatre, on s'intéressera à son arithmétique. Dans le chapitre cinq, intitulé Suites Récurrentes Linéaires et Applications au Codage, on va développer la partie du codage en utilisant le théorème de Zeckendorf où on montrera comment coder et décoder un entier en utilisant la suite de Fibonacci classique, la suite de Fibonacci d'ordre supérieur et la suite de Gopala-Hemachandra. Enfin, on va développer la partie codage en utilisant la Qp-matrice de Fibonacci (simple ou généralisée) où on va Établir les hautes capacités de détection et de correction basées sur le déterminant de la matrice du code.

Chapitre 1

Théorie des codes et cryptographie

1.1 Généralités sur les codes linéaires

Dans ce qui va suivre, nous avons tenté de résumer quelques notions fondamentales sur les codes, en incluant un bref aperçu sur un Cryptosystème Mc Eliece basé sur les codes linéaires . Le lecteur peut consulter [9], [10], [11] , pour de plus amples informations sur ce sujet .

Définition 1.1. Soit A un alphabet (souvent on considère des corps finis), et soit $\mathcal{A}$ l'ensemble des vecteurs formés à partir de A. Un code C est un sous ensemble de $\mathcal{A}$. Un élément de C est appelé mot de code.

Exemple 1.1. On considère le corps $F_2 = \{0, 1\}$. L'ensemble

$$C = \{000, 111, 011, 101, 001\}$$

est un code de longueur 3, il contient 5 mots. Ce genre de codes est appelé code en bloque.

1.1.1 Définitions et propriétés

Considérons le corps de base F_q où $q = p^m$ est une puissance d'un nombre premier.

Définition 1.2. Soit k, $n \in \mathbb{N}^*$ avec $k \leq n$.

Un code linéaire de longueur n et de dimension k est un sous espace vectoriel de dimension k de F_q^n.

On note un tel code par $[n,k]-code$.

Définition 1.3. Soit C un $[n,k]-code$ sur F_q. Soit $(c_1,...,\ c_k)$ une base de C. Alors la matrice

$$G = \begin{pmatrix} \boxed{c_1} \\ \boxed{c_2} \\ \boxed{\begin{matrix}.\\.\end{matrix}} \\ \boxed{c_k} \end{pmatrix}$$

est appelée une matrice génératrice de C.

Définition 1.4. Soit C un $[n,k]-code$ et G une matrice génératrice de C. On dit que G est sous forme systématique si elle est de la forme

$$G = (I_k \mid R) \text{ où } R \subset M_{k,n-k}(F_q).$$

Définition 1.5. Soit C un $[n,k]-code$ sur F_q. On appelle la matrice de contrôle de parité de C une matrice H qui vérifie :

$$C = \{x \in F_q^n \mid H\ x^t = 0\}.$$

Remarque 1.1. Si $G = (I_k \mid R)$, alors $H = (-R^t \mid I_{n-k})$.

Définition 1.6. On définit sur F_q^n une métrique appelée distance de Hamming de deux mots x et y de longueur fixe est :

$$d(x,y) = \mid \{i\ /\ x_i \neq y_i\} \mid .$$

Le poids de Hamming d'un mot x, qu'on note $\omega(x)$, est le nombre de coordonnées non nulles de x, i.e. $\omega(x) = d(x,0)$.
La distance minimale d'un code C

$$d(c) = \min\{d(x,y)/\ x,\ y \in C,\ x \neq y\}.$$

Le poids minimal d'un code C

$$\omega(c) = \min\{\omega(x)/\ x \in C,\ x \neq 0\}.$$

Théorème 1.1. *La distance de Hamming définit une distance sur* F_q^n.

Preuve 1. *Preuve 1. –* $d(x,y) = 0 \Leftrightarrow card\{i \in (1,.....,n), x_i \neq y_i\} = 0 \Leftrightarrow x_i = y_i, \forall i \in \{1,....n\} \Leftrightarrow x = y$

2. – La propriété : $\forall x, y$ *dans* $F_q^n, d(x,y) = d(y,x)$ *découle directement de la définition,* $(x_i \neq y_i) \Leftrightarrow (y_i \neq x_i)$.

3. – Soit $x, y \in F_q^n$ *et montrons que* $d(x,z) \leq d(x,y) + d(y,z)$.

Notons $A = \{i \in (1,...n), x_i \neq z_i\}, B = \{i \in (1,...n), x_i \neq y_i\}, C = \{i \in (1,...n), y_i \neq z_i\}$

Montrons que : $A \subset B \cup C$

On utilise la contraposée : si $i \notin B \cup C \Rightarrow i \in \overline{B} \cap \overline{C} \Rightarrow i \in \overline{B}$ *et* $i \in \overline{C} \Rightarrow x_i = y_i$ *et* $y_i = z_i \Rightarrow x_i = z_i$.

Donc, $A = card\ (B \cup C) = card\ A + card\ B - card(B \cap C) \leq card\ A + card\ B$. *D'ou* $d(x,y) \leq d(x,z) + d(z,y)$.

Alors la distance de Hamming définie une distance .

Théorème 1.2. *Soit* C *un* $[n,k,d] - code$, *où* $d = 2t+1$, *alors* C *détecte et corrige jusqu'à* t *erreurs.*

Démonstration. Soit x le mot envoyé et y le mot reçu. Alors $y = x + e$, où e est le vecteur erreur. On suppose que $w(e) \leq t$. Alors, on peut affirmer que x est le seul mot du code C qui correspond à y.

En effet, supposons que z est un autre mot du code C, vérifiant $d(z,t) \leq t$, alors on aura $2t+1 = d \leq d(x,z) \leq d(x,y) + d(y,z) \leq 2t$. Contradiction. □

1.2 Les opérations d'encodage et de décodage

1.2.1 L'étape d'encodage

Soit C un code $[n, k]$ linéaire sur F_q de matrice génératrice G alors il contient q^k mots et peut être utilisé pour envoyer q^k messages. L'étape d'encodage consiste à ajouter une certaine redondance au message de tels sorte à le rendre cohérent avec le code. En effet, pour encoder un message x il suffit de le multiplier par la matrice génératrice G, on obtient alors un mot du code.

Exemple 1.2. Soit C un code linéaire de matrice génératrice G donnée par :

$$G = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{pmatrix}.$$

$u_1 = 011$ correspond au message $m_1 = (011011)$.

$u_2 = 101$ correspond au message $m_2 = (101101)$.

Remarque 1.2. Se donner une matrice génératrice sous forme systématique s'avère être très intéressant en pratique.

1.2.2 L'étape de décodage

La deuxième étape de la transmission consiste à envoyer le message à travers un canal. Puisque le canal ajoute un bruit, le message arrive déformé (un certain nombre d'erreurs a été ajouté au message initial) et là, le décodeur doit corriger les erreurs pour retrouver le message initial avant de le donner à l'utilisateur. C'est ce qu'on appelle l'étape de décodage.

1.2.3 Le décodage par syndrôme

Définition 1.7. Soit C un $[n, k]$ code linéaire, à toute matrice de parité H de C on associe l'application linéaire suivante dite application syndrôme

$$\begin{aligned} S_H : F_q^n &\longrightarrow F_q^{n-k} \\ y &\longmapsto S_H(y) = Hy^t \end{aligned}$$

Définition 1.8. Le translaté d'un vecteur $y \in F_q^n$ est défini comme étant $\{y + c \mid c \in C\}$. À tous les translatés on peut associer un mot du plus petit poids contenu dans le translaté (mot non nécessairement unique), ce mot est appelé coset leader et est noté cl.

L'algorithme du décodage par syndrôme pour un code de matrice de contrôle H consiste à décoder tout mot reçu y en $y - cl$ pour cl le coset leader associé au syndrôme Hy^t de y. Autrement, cet algorithme consiste à faire les étapes suivantes :

1- Calculer le syndrôme $s = Hy^t$ de y.

2- Dans la liste $S_H^{-1}(s) + C$ trouver le coset leader cl.

3- Décoder $y \mapsto y - cl$.

Exemple 1.3. On considère le code

$$C = \{000000, 100110, 010101, 110011, 001011, 101101, 011110, 111000\}.$$

C'est un code $[6, 3, 3]$ de matrice de contrôle

$$H = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \end{pmatrix}.$$

Supposons qu'on a reçu le vecteur $y = 101011$. Le vecteur syndrôme de y est $s = Hy^t = (110)^t$. On a

$$S_H^{-1}(s) = 101011 + C = \{101011, 001101, 111110, 011000, 100000, 000111, 010011, 011111\},$$

le coset leader $cl(s) = 10000$, et donc on décode par $y - cl(s) = 001011 \in C$.

1.3 Cryptographie

1.3.1 Définitions et propriétés

La cryptographie est une technique mathématique liée à la sécurité de l 'information, afin de protéger un message on lui applique une transformation qui le rend incompréhensible, c'est ce qu'on appelle le chiffrement, qui à partir d'un texte en clair, donne un texte

chiffré. Inversement, le déchiffrement est l'action qui permet de reconstruire le texte en clair à partir du texte chiffré.

Un système cryptographie est la donnée de :

P : l'ensemble fini des clairs.

C : l'ensemble fini des chiffrés.

K : l'ensemble fini des clés.

Pour toute clé k de K, il existe une fonction de chiffrement $e_k : P \to C$ et une fonction de déchiffrement $d_k : C \to P$ telle que $d_k \circ e_k = Id_p$.

1.4 Principes de la cryptographie

1.4.1 Cryptographie symétrique

La cryptographie symétrique a été utilisée par la première fois par Jules César. Sur la base d'une seule clef secrète, il s'agit de réaliser une transformation capable de rendre illisible puis de restituer une information. Dans la cryptographie symétrique, les clefs de chiffrement et de déchiffrement sont identiques : c'est la clef secrète, qui doit être connue des tiers communiquant et d'eux seuls. Le procédé de chiffrement est dit symétrique.

Exemple 1.4. (L'algorithme de Jules César)

Algorithme : Remplacer chaque lettre par une autre lettre.

Clé : permutation

Avec la clé :

$$k = \begin{pmatrix} ABCDEFGHIJKLMNOPQSRSTUVWXYZ \\ UZSAGYECTLPBXOVDWJNKFRHMQI \end{pmatrix}$$

L'ALGERIE EST UN BEAU PAYS

devient

B'UBEGJTG CNK FO ZGUF DUQN

Exemple 1.5. DES (Data Encryption Standard)

Le gouvernement américain a adopté, en 1977, la fonction DES (Data Encryption Standard) comme algorithme de chiffrement standard officiel. Le DES est un algorithme de chiffrement par blocs qui agit sur des blocs de 64 bits. La longueur de la clef est de 56 bits.

1.4.2 Cryptographie asymétrique

On utilise deux clés différentes (une publique et une secrète) respectivement pour les opérations de cryptage et de décryptage.

Exemple 1.6. RSA (Rivest, Shamir, Adleman)

En 1976, Diffie et Hellman publièrent un article "*New Directions in Cryptography*" dans lequel ils introduisent le concept de cryptographie à clef publique qui constituait une réelle alternative à celui des cryptosystèmes à clef secrète.

Le plus bel exemple de cryptosystème à clef publique et sans conteste le plus simple apparut juste deux ans plus tard en 1978. Il est dû à Rivest, Shamir et Adleman d'où le nom RSA.

Dans le système RSA, un utilisateur crée son couple (clef publique, clef privée), en utilisant la procédure suivante :

1. – Choisir au hasard deux grands nombres premiers p et q (plus de 100 chiffres).
2. – Calculer $n = pq$
3. – Choisir un entier e qui est premier avec $(p-1)(q-1)$.

Le $p\,\text{gcd}\ (e,(p-1)(q-1)\ ) = 1$, e est inversible dans $\dfrac{\mathbb{Z}}{(p-1(q-1)\mathbb{Z}}$.

4. – $\exists$ un d unique $(1 \leq d \leq (p-1)(q-1))$ est vérifier $ed = 1 \mod (p-1)(q-1)$
5. – Calculer d.
6. – Publier la paire (e, n) comme la clef publique RSA
7. – Garder secrète la paire (d, n) qui sera clef privée RSA.

Le chiffrement RSA consiste à effectuer $c = m^e \mod n$.

Pour Le déchiffrement RSA on a

$$c = m^e \mod n$$
$$c^d = m^{ed} \mod n$$
$$c^d = m^{1 \mod (p-1)(q-1)} \mod n = m \mod n$$

Donc, on trouve $m = c^d \mod n$

Bien que le cryptosystème RSA soit le plus utilisé aujourd'hui, il présente l'inconvénient comme la majorité des cryptosystèmes à clef publique d'être une centaine de fois plus lent qu'un cryptosystème à clef secrète ordinaire

1.5 Cryptosystème basés sur les codes linéaires (Mc Eliece)

En 1978, un cryptosystème à clef publique nommé cryptosystème de Mc Eliece fut inventé par Robert Mc Eliece. Ce cryptosystème repose sur un problème difficile issu de la théorie des codes et se présente comme une alternative au système inventé deux ans auparavant qui est le RSA et cela grâce à des propriétés majeures que le cryptosystème de Mc Eliece possède et qui font défaut à RSA.
La première qualité qu'on peut trouver à ce cryptosystème est sa vitesse du chiffrement.
Un deuxième avantage tout aussi important que le premier est qu'il repose sur un problème très différent des algorithmes asymétriques usuels. Cela signifie qu'une percée théorique dans le domaine de la factorisation, qui ruinerait RSA par exemple, n'affecterait en rien ce système.

1.6 Matrice de permutation

Définition 1.9. Une matrice de permutation d'ordre n est une matrice carrée d'ordre n obtenue en permutant les lignes (ou les colonnes) de la matrice identité. Chaque ligne et chaque colonne de P contient exactement un seul élément égal à 1 et tous les autres éléments sont 0.

Remarque 1.3. Il y a un seul 1 par ligne et (n-1) zéro.

Il y a un seul 1 par colonne et (n-1) zéro.

Exemple 1.7. La matrice donnée ci-dessous est une matrice de permutation.

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

1.7 Cryptosystème de Mc Eliece

Le cryptosystème de Mc Eliece est un schéma de chiffrement asymétrique, ce cryptosystème est basé sur l'utilisation des codes correcteurs d'erreur. On utilise les différentes matrices nécessaires pour générer les clefs :

G : Matrice génératrice du code C de taille $k \times n$.

S : Matrice inversible d'ordre k.

P : Matrice de permutation d'ordre n.

1.7.1 Génération des clefs

On sélectionne d'abord aléatoirement un $[n, k] - code$ linéaire C capable de corriger t erreurs. On construit ensuite la matrice G de taille $k \times n$ génératrice du code C. Puis on sélectionne aléatoirement une matrice binaire S d'ordre k et inversible.

On tire aléatoirement une matrice de permutation P d'ordre n.

On calcule enfin la matrice $G\prime = SGP$

Ainsi, la clef publique est $(G\prime, t)$; la clef privée est (S, G, P)

1.7.2 Chiffrement

Pour chiffrer un message m binaire de longueur k. On calcule d'abord le vecteur $c' = mG'$. On génère ensuite un vecteur erreur e de poids t. Enfin, on calcule le chiffré y tel que $y = c' + e$

1.7.3 Déchiffrement

Pour déchiffrer y, on calcule $y\prime$ tel que $y' = yP^{-1}$.

On a

$$y = c' + e = m\ G' + e = m\ S\ G\ P + e.$$

On multiplie par p^{-1} a droite on trouve

$$y = m\ S\ G + ep^{-1}.$$

Comme $e\ p^{-1}$ est poids t, alors l'algorithme de décodage permet de retrouver $m\ S\ G$, donc aussi $m\ S$, puis m et le multipliant par S^{-1}.

Exemple 1.8. Pour notre exemple, nous utiliseront le $[7, 4] - code$ de Hamming qui corrige toutes les erreurs simples. Un générateur de code de cette matrice est donné par

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Rachid choisit la matrice inversible

$$S = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 \end{bmatrix}$$

et la matrice de permutation :

$$P = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

Rachid rend public le générateur de la matrice

$$G' = SGP = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \end{bmatrix}$$

Si Asma souhaite envoyer le message $m = (1101)$ à Rachid, elle choisit au hasard un vecteur erreur de poids de 1, par exemple $e = (0\ 0\ 0\ 0\ 1\ 0\ 0)$ et calcule :

$$\begin{aligned} y = mG' + e &= (0\ 1\ 1\ 0\ 0\ 1\ 0) + (0\ 0\ 0\ 0\ 1\ 0\ 0) \\ &= (0\ 1\ 1\ 0\ 1\ 1\ 0) \end{aligned}$$

qu' elle envoie à Rachid. A la réception de y, Rachid calcule $y' = yp^{-1}$, où

$$P^{-1} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

et obtient $y\prime = (1\ 0\ 0\ 0\ 1\ 1\ 1)$. Maintenant, rachid décode y' par l'algorithme de décodage. L'erreur se produit en position 7. rachid a maintenant le mot de code $y'' = (1000110)$. Rachid sait que $mS = (1000)$, et il peut désormais obtenir m en multipliant par la matrice

$$S^{-1} = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \end{bmatrix}$$

et on obtient $m = (1101)$.

Chapitre 2

Suites Récurrentes Linéaires

Dans ce chapitre on va introduire les suites de Fibonacci, Lucas et les suites de Fibonacci et Lucas généralisées ainsi que certaines propriétés dont on aura besoin dans la suite de cette Thèse.

2.1 Suite de Fibonacci

Définition 2.1. Soit n un entier positif, on définit la suite de Fibonacci $(F_n)_n$ par

$$\begin{cases} F_0 = 0, \\ F_1 = 1, \\ F_{n+1} = F_n + F_{n-1} \quad (n \geq 1), \end{cases}$$

2.1.1 Suite de Fibonacci généralisée

Dans cette section, nous allons présenter deux extensions distinctes de la suite de Fibonacci : la Multibonacci et la p-Fibonacci .

Multibonacci

Définition 2.2. On dit que (g_n) est une suite de Fibonacci généralisée d'ordre k (multibonacci) si elle satisfait la récurrence :

$$\begin{cases} g_n^{(k)} = g_{n-1}^{(k)} + g_{n-2}^{(k)} + \ldots + g_{n-k}^{(k)} \text{ , } \forall n \geq 2 \\ g_0^{(k)} = 0, g_1^{(k)} = 1. \\ g_{-n}^{(k)} = \ldots = g_{-2}^{(k)} = g_{-1}^{(k)} = 0 \end{cases}$$

Ci-dessous, nous présentons les valeurs de ces nombres pour les premières valeurs k et n.

k	Nom	Premiers termes non nuls ($n \geqslant 1$)
2	Fibonacci	$1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, \ldots$
3	Tribonacci	$1, 1, 2, 4, 7, 13, 24, 44, 81, 149, 274, 504, 927, 1705, 3136, \ldots$
4	Tetranacci	$1, 1, 2, 4, 8, 15, 29, 56, 108, 208, 401, 773, 1490, 2872, 5536, \ldots$
5	Pentanacci	$1, 1, 2, 4, 8, 16, 31, 61, 120, 236, 464, 912, 1793, 3525, 6930, \ldots$
6	Hexanacci	$1, 1, 2, 4, 8, 16, 32, 63, 125, 248, 492, 976, 1936, 3840, 7617, \ldots$
7	Heptanacci	$1, 1, 2, 4, 8, 16, 32, 64, 127, 253, 504, 1004, 2000, 3984, 7936, \ldots$
8	Octanacci	$1, 1, 2, 4, 8, 16, 32, 64, 128, 255, 509, 1016, 2028, 4048, 8080, \ldots$
9	Nonanacci	$1, 1, 2, 4, 8, 16, 32, 64, 128, 256, 511, 1021, 2040, 4076, 8144, \ldots$
10	Decanacci	$1, 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1023, 2045, 4088, 8172, \ldots$

Table 2.1- Premiers nombres k-Fibonacci non nuls

La p-Fibonacci

Définition 2.3. En 1977, les p-nombres de Fiobnacci ont été introduits par Mc Laughlin et Sury (voir [18]). Pour un entier donné $p = 0, 1, 2, 3, \ldots$, les p-nombres de Fibonacci sont donnés par la relation de recurrence suivante :

$$\begin{cases} F_p(n) = F_p(n-1) + F_p(n-p-1), \text{ si } n > p+1 \\ F_p(1) = F_p(2) = \ldots = F_p(p) = F_p(p+1) = 1 \end{cases} \tag{2.1.1}$$

Quelques propriétés

Si on prend $p = 0$ la relation (2.1.1) se réduit à :

$$\begin{cases} F_0(n) = F_0(n-1) + F_0(n-1), n > 1 \\ F_0(1) = 1 \end{cases}$$

Cette relation de récurrence génère les nombres binaires $1, 2, 4, \ldots, 2^{n-1}, ..$

Pour $p = 1$ la relation (2.1.1) s'écrit de cette forme :

$$\begin{cases} F_1(n) = F_1(n-1) + F_1(n-2), \text{ si } n > 2 \\ F_1(1) = F_1(2) = 1 \end{cases}$$

Cette relation génère les nombres de Fibonacci classiques $1, 1, 2, 3, 5, 8, 13, \ldots$

Comme les nombres de Fibonacci classiques ; les p-nombres de Fibonacci permettent leur extension aux valeurs négatives pour l'argument n, pour calculer ces nombres $F_p(0), F_p(-1), F_p(-2), \ldots,$ utilise la relation (2.1.1), et en représentant le nombre $F_p(p+1)$ de la forme (2.1.1) on aura :

$$F_p(p+1) = F_p(p) + F_p(0)$$

et puisque $F_p(p) = F_p(p+1) = 1$ donc $F_p(0) = 0$. En suivant ce processus, la représentation des p-nombres de Fibonacci $F_p(p), F_p(p-1), \ldots, F_p(2)$ de la forme (2.1.1) on aura

$$F_p(0) = F_p(-1) = F_p(-2) = \ldots = F_p(-p+1) = 0$$

Maintenant, on représente le $F_p(1)$ de la forme

$$F_p(1) = F_p(0) + F_p(-p)$$

Comme $F_p(1) = 1$ et $F_p(0) = 0$, on déduit que $F_p(-p) = 1$. On peut aussi trouver que $F_p(-p-1) = F_p(-p-2) = \ldots = F_p(-2p+1) = 0$. Ce tableau donne quelques valeurs de $F_p(n)$

n	5	4	3	2	1	0	-1	-2	-3	-4	-5
$F_1(n)$	5	3	2	1	1	0	1	-1	2	-3	5
$F_2(n)$	3	2	1	1	1	0	0	1	0	-1	1
$F_3(n)$	2	1	1	1	1	0	0	0	1	0	0
$F_4(n)$	1	1	1	1	1	0	0	0	0	1	0
$F_5(n)$	1	1	1	1	1	0	0	0	0	0	1

La propriété suivante des p-nombres de Fibonacci a été prouvé dans [30] :

$$F_p(1) + F_p(2) + F_p(3) + \ldots F_p(n) = F_p(n + p + 1) - 1$$

Cette formule inclut un nombre remarquable de formules de mathématiques discrètes, par exemple, pour le cas $p = 0$ cette formule va se réduire à la suivante qui est très connue pour les nombres binaires :

$$2^0 + 2^1 + 2^2 + \ldots + 2^{n-1} = 2^n - 1$$

Comme il à été mentionné déjà, pour $p = 1$ les p-nombres de Fibonacci coïncides avec les nombres de Fibonacci classiques et on aura la formule suivante :

$$F_1 + F_2 + F_3 + \ldots + F_n = F_{n+2} - 1$$

2.1.2 Q-matrice de Fibonacci

Dans les derniers temps, la théorie des nombres de Fibonacci a été complétée par une théorie appelée Q-matrice de Fibonacci.

La Q-matrice représente la matrice carrée d'ordre deux suivante :

$$Q = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \tag{2.1.2}$$

Le déterminant de cette matrice est égal à -1 .

La relation entre cette matrice et les nombres de Fibonacci est la matrice Q^n définit comme suit :

$$Q^n = \begin{pmatrix} F_{n+1} & F_n \\ F_n & F_{n-1} \end{pmatrix} \tag{2.1.3}$$

où F_{n+1}, F_n, F_{n-1} sont des nombres de Fibonacci.

Quelques propriétés de la Q-matrice de Fibonacci

O sait que $\mathrm{Det}\,(A^n) = \mathrm{Det}(A)^n$, ceci implique que :

$$\mathrm{Det}\, Q^n = (-1)^n \tag{2.1.4}$$

où n est un entier.

Mais si on calcule Det Q^n en utilisant (2.1.3) et (2.1.4) on aura cette identité :

$$\text{Det}\, Q^n = F_{n-1} + F_{n+1} - F_n^2 = (-1)^n.$$

Cette égalité représente une des plus importantes propriétés des nombres de Fibonacci. Maintenant, représentons Q^n de cette forme :

$$\begin{aligned} Q^n &= \begin{pmatrix} F_{n+1} & F_n \\ F_n & F_{n-1} \end{pmatrix} = \begin{pmatrix} F_n + F_{n-1} & F_{n-1} + F_{n-2} \\ F_{n-1} + F_{n-2} & F_{n-2} + F_{n-3} \end{pmatrix} \\ &= \begin{pmatrix} F_n & F_{n-1} \\ F_{n-1} & F_{n-2} \end{pmatrix} + \begin{pmatrix} F_{n-1} & F_{n-2} \\ F_{n-2} & F_{n-3} \end{pmatrix} \\ &= Q^{n-1} + Q^{n-2}. \end{aligned} \tag{2.1.5}$$

ou autrement

$$Q^{n-2} = Q^n - Q^{n-1}. \tag{2.1.6}$$

Il a été prouvé par [31] que $Q^n Q^m = Q^m Q^n = Q^{n+m}$.

La formule explicite des matrices $Q^n, (n = 0, \pm1, \pm2, \pm3, \pm4\pm, \pm5)$ obtenu par les formules récurrentes (2.1.5) et (2.1.6) est donnée par la table suivante :

	0	1	2	3	4	5
Q^n	$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$	$\begin{pmatrix} 2 & 1 \\ 1 & 1 \end{pmatrix}$	$\begin{pmatrix} 3 & 2 \\ 2 & 1 \end{pmatrix}$	$\begin{pmatrix} 5 & 3 \\ 3 & 2 \end{pmatrix}$	$\begin{pmatrix} 8 & 5 \\ 5 & 3 \end{pmatrix}$
Q^{-n}	$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 \\ 1 & -1 \end{pmatrix}$	$\begin{pmatrix} 1 & -1 \\ -1 & 2 \end{pmatrix}$	$\begin{pmatrix} -1 & 2 \\ 2 & -3 \end{pmatrix}$	$\begin{pmatrix} 2 & -3 \\ -3 & 5 \end{pmatrix}$	$\begin{pmatrix} -3 & 5 \\ 5 & -8 \end{pmatrix}$

FIGURE 2.1 – La formule explicite des matrices Q^n

Ce tableau donne directement les matrices Q^n. On remarque qu'il est facile de trouver la matrice inverse Q^{-n}, on a

$$Q^{-n} = \begin{cases} \begin{pmatrix} F_{n-1} & -F_n \\ -F_n & F_{n+1} \end{pmatrix}, \text{ si } n = 2k \\ \begin{pmatrix} -F_{n+1} & F_n \\ F_n & -F_{n+1} \end{pmatrix}, \text{ si } n = 2k+1 \end{cases}$$

2.1.3 Les matrices de Fibonacci généralisées

Il a été introduit dans [31] que les matrices de Fibonacci généralisées sont désignées par Q_p pour un p donné ($p = 0, 1, 2, 3, \ldots$).

Définition 2.4. On définit la Q_p-matrice est une matrice carrée $(p+1) \times (p+1)$:

$$Q_p = \begin{pmatrix} 1 & 1 & 0 & . & . & . & . & . & 0 \\ 0 & 0 & 1 & . & . & . & . & . & 0 \\ . & . & . & . & & & & . & \\ . & . & . & . & . & & & & . \\ . & . & . & . & . & . & & & . \\ . & . & . & & . & . & . & & . \\ . & . & . & & & . & . & . & . \\ 0 & 0 & 0 & . & . & . & . & 0 & 1 \\ 1 & 0 & 0 & . & . & . & . & 0 & 0 \end{pmatrix}$$

Pour $p = 1, 2, 3, 4$ voici les Q_p-matrices qui leurs corresponds :

$$Q_0 = (1); \quad Q_1 = Q = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}; Q_2 = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}, Q_3 = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{pmatrix}, Q4 = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Les théorèmes suivants qui concernes les Q_p- matrices sont prouvés dans [31] :

Théorème 2.1. *Pour un entier donné* $p = 0, 1, 2, 3, \ldots$ *on a*

$$\operatorname{Det} Q_p = (-1)^p$$

Théorème 2.2. *Pour un entier donné* $p = 0, 1, 2, 3, \ldots$ *on a*

$$Q_p^n = \begin{pmatrix} F_p(n+1) & F_p(n) & \cdots & F_p(n-p+2) & F_p(n-p+1) & F_p(n-p) \\ F_p(n-p+1) & F_p(n-p) & \cdots & & F_p(n-2p+2) & F_p(n-2p+1) \\ \vdots & \vdots & & & \vdots & \vdots \\ \vdots & \vdots & & & \vdots & \vdots \\ \vdots & \vdots & \ddots & & \vdots & \vdots \\ \vdots & \vdots & & & \vdots & \vdots \\ & \vdots & & \ddots & \vdots & \vdots \\ F_p(n-1) & F_p(n-2) & \cdots & \vdots & F_p(n-p) & F_p(n-p-1) \\ F_p(n) & F_p(n-1) & \cdots & \vdots & F_p(n-p+1) & F_p(n-p) \end{pmatrix}$$

FIGURE 2.2 – Matrice Q_p^n

Où $F_p(n)$ est le p-nombre de Fibonacci, $n = 0, 1, 2, 3, \ldots$

Théorème 2.3. *Pour un entier donné* $p = 0, 1, 2, 3, \ldots$ *on a*

$$Q_p^n Q_p^m = Q_p^m Q_p^n = Q_p^{n+m}$$

et

$$Q_p^n = Q_p^{n-1} + Q_p^{n-p-1}.$$

Théorème 2.4. *Pour un entier donné* $p = 0, 1, 2, 3, \ldots$ *on a* $\operatorname{Det} Q_p^n = (-1)^{pn} = (\operatorname{Det} Q_p)^n$, *où* $n = 0, \pm 1, \pm 2, \pm 3, \ldots$

2.2 Suite de Lucas

2.2.1 Suite de Lucas

Définition 2.5. Les nombres de Lucas, $(L_n)_n$, sont définis par la formule de récurrence :

$$\begin{cases} L_{n+2} = L_{n+1} + L_n, \quad n \geq 0 \\ L_0 = 2, \quad L_1 = 1 \end{cases}$$

et pour tout $n \geq 1$, on a l'identité bien connue

$$L_n = F_{n+1} + F_{n-1}.$$

2.2.2 Suite de Lucas généralisée

Définition 2.6. On dit que (g_n) est une suite de Lucas généralisée d'ordre k si elle satisfait la récurrence :

$$\begin{cases} g_n^{(k)} = g_{n-1}^{(k)} + g_{n-2}^{(k)} + \ldots + g_{n-k}^{(k)} \text{ , } \forall n \geq 2 \\ g_0^{(k)} = 2, g_1^{(k)} = 1. \\ g_{-n}^{(k)} = \ldots = g_{-2}^{(k)} = g_{-1}^{(k)} = 0 \end{cases}$$

Ci-dessous, nous présentons les valeurs de ces nombres pour les premières valeurs k et n.

k	Nom	Premiers termes non nuls ($n \geqslant 0$)
2	Lucas	$2, 1, 3, 4, 7, 11, 18, 29, 47, 76, 123, 199, 322, 521, 843, 1364, \ldots$
3	3 -Lucas	$2, 1, 3, 6, 10, 19, 35, 64, 118, 217, 399, 734, 1350, 2483, 4567, \ldots$
4	4 -Lucas	$2, 1, 3, 6, 12, 22, 43, 83, 160, 308, 594, 1145, 2207, 4254, 8200, \ldots$
5	-Lucas	$2, 1, 3, 6, 12, 24, 46, 91, 179, 352, 692, 1360, 2674, 5257, 10335, \ldots$
6	6 -Lucas	$2, 1, 3, 6, 12, 24, 48, 94, 187, 371, 736, 1460, 2896, 5744, 11394, \ldots$
7	7 -Lucas	$2, 1, 3, 6, 12, 24, 48, 96, 190, 379, 755, 1504, 2996, 5968, 11888, \ldots$
8	8 -Lucas	$2, 1, 3, 6, 12, 24, 48, 96, 192, 382, 763, 1523, 3040, 6068, 12112, \ldots$
9	9 -Lucas	$2, 1, 3, 6, 12, 24, 48, 96, 192, 384, 766, 1531, 3059, 6112, 12212, \ldots$
10	10-Lucas	$2, 1, 3, 6, 12, 24, 48, 96, 192, 384, 768, 1534, 3067, 6131, 12256, \ldots$

Table 2.2- Premiers nombres k-Lucas non nuls

2.2.3 La p-Lucas

En 2006, les p-nombres de Lucas ont été introduits par Stakhov et Rozin (voir [33]). Pour un entier donné $p = 0, 1, 2, 3, \ldots$, les p-nombres de Lucas sont donnés par la relation de récurrence suivante :

$$\begin{cases} L_p(n) = L_p(n-1) + L_p(n-p-1), \text{ si } n > p+1 \\ L_p(1) = L_p(2) = \cdots = L_p(p) = 1, L_p(p+1) = p+2 \end{cases} \tag{2.2.1}$$

Pour $p = 1, L_1(n) = L_n$ sont connus sous le nom de nombres classiques de Lucas.

2.2.4 $Q_{p,m}$-matrice de Lucas

Il a été introduit dans [27] que les matrices de Lucas généralisées sont désignées par $Q_{p,m}$ pour un p donné ($p = 0, 1, 2, 3, \ldots$) et $m > 0$.

Définition 2.7. On définit une nouvelle matrice appelée $Q_{p,m}$ matrice de Lucas d'ordre $(p+1)$ sur la m-extension des nombres p de Lucas où $p > 0$ est entier et $m > 0$.

$$Q_{p,m} = \begin{pmatrix} m & 1 & 0 & . & . & . & . & 0 & 0 \\ 0 & 0 & 1 & . & . & . & . & 0 & 0 \\ . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . \\ 0 & 0 & 0 & . & . & . & . & 0 & 1 \\ 1 & 0 & 0 & . & . & . & . & 0 & 0 \end{pmatrix}$$

FIGURE 2.3 – Matrice $Q_{p,m}$

En utilisant $L_{p,m}(n) = m^{n-1}, n = 1, 2, 3, 4, \ldots\ldots, p+1$ on peut écrire

$$Q_{p,m} = \begin{pmatrix} L_{p,m}(2) & L_{p,m}(1) & \cdots & L_{p,m}(3-p) & L_{p,m}(2-p) \\ L_{p,m}(2-p) & L_{p,m}(1-p) & \cdots & L_{p,m}(3-2p) & L_{p,m}(2-2p) \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ L_{p,m}(0) & L_{p,m}(-1) & \cdots & L_{p,m}(1-p) & L_{p,m}(-p) \\ L_{p,m}(1) & L_{p,m}(0) & \cdots & L_{p,m}(2-p) & L_{p,m}(1-p) \end{pmatrix}$$

Notons que la $Q_{p,m}$ est une matrice carrée $(p+1) \times (p+1)$.

voici les $Q_{p,m}$ matrices qui leurs corresponds :

$$Q_{1,m} = \begin{pmatrix} m & 1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} L_{1,m}(2) & L_{1,m}(1) \\ L_{1,m}(1) & L_{1,m}(0) \end{pmatrix}$$

$$Q_{2,m} = \begin{pmatrix} m & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} = \begin{pmatrix} L_{2,m}(2) & L_{2,m}(1) & L_{2,m}(0) \\ L_{2,m}(0) & L_{2,m}(-1) & L_{2,m}(-2) \\ L_{2,m}(1) & L_{2,m}(0) & L_{2,m}(-1) \end{pmatrix}$$

$$Q_{3,m} = \begin{pmatrix} m & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{pmatrix} = \begin{pmatrix} L_{3,m}(2) & L_{3,m}(1) & L_{3,m}(0) & L_{3,m}(-1) \\ L_{3,m}(-1) & L_{3,m}(-2) & L_{3,m}(-3) & L_{3,m}(-4) \\ L_{3,m}(0) & L_{3,m}(-1) & L_{3,m}(-2) & L_{3,m}(-3) \\ L_{3,m}(1) & L_{3,m}(0) & L_{3,m}(-1) & L_{3,m}(-2) \end{pmatrix}$$

$$Q_{4,m} = \begin{pmatrix} m & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$$= \begin{pmatrix} L_{4,m}(2) & L_{4,m}(1) & L_{4,m}(0) & L_{4,m}(-1) & L_{4,m}(-2) \\ L_{4,m}(-2) & L_{4,m}(-3) & L_{4,m}(-4) & L_{4,m}(-5) & L_{4,m}(-6) \\ L_{4,m}(-1) & L_{4,m}(-2) & L_{4,m}(-3) & L_{4,m}(-4) & L_{4,m}(-5) \\ L_{4,m}(0) & L_{4,m}(-1) & L_{4,m}(-2) & L_{4,m}(-3) & L_{4,m}(-4) \\ L_{4,m}(1) & L_{4,m}(0) & L_{4,m}(-1) & L_{4,m}(-2) & L_{4,m}(-3) \end{pmatrix}$$

Théorème 2.5. *[28] Pour un entier donné $p = 0, 1, 2, 3, \ldots$ on a*

$Q^n_{p,m} =$

$$\begin{pmatrix} L_{p,m}(n+1) & L_{p,m}(n) & \ldots & L_{p,m}(n-p+2) & L_{p,m}(n-p+1) \\ L_{p,m}(n-p+1) & L_{p,m}(n-p) & \ldots & L_{p,m}(n-2p+2) & L_{p,m}(n-2p+1) \\ \vdots & \vdots & \ldots & \vdots & \vdots \\ \vdots & \vdots & \ldots & \vdots & \vdots \\ \vdots & \vdots & \ldots & \vdots & \vdots \\ \vdots & \vdots & \ldots & \vdots & \vdots \\ \vdots & \vdots & \ldots & \vdots & \vdots \\ L_{p,m}(n-1) & L_{p,m}(n-2) & \ldots & L_{p,m}(n-p) & L_{p,m}(n-p-1) \\ L_{p,m}(n) & L_{p,m}(n-1) & \ldots & L_{p,m}(n-p+1) & L_{p,m}(n-p) \end{pmatrix}$$

FIGURE 2.4 – Matrice $Q^n_{p,m}$

où $L_{p,m}(n) = m^{n-1}$.

Preuve 2. *Quand $p = 1$, nous devons prouver*

$$Q^n_{1,m} = \begin{pmatrix} L_{1,m}(n+1) & L_{1,m}(n) \\ L_{1,m}(n) & L_{1,m}(n-1) \end{pmatrix} \tag{2.1}$$

Nous allons le prouver par induction mathématique.

Pour $n = 1$

$$Q_{1,m} = \begin{pmatrix} m & 1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} L_{1,m}(2) & L_{1,m}(1) \\ L_{1,m}(1) & L_{1,m}(0) \end{pmatrix}$$

ceci est vrai pour $n = 1$.

Pour $n = 2$

$$Q_{1,m}^2 = \begin{pmatrix} m^2+1 & m \\ m & 1 \end{pmatrix} = \begin{pmatrix} L_{1,m}(3) & L_{1,m}(2) \\ L_{1,m}(2) & L_{1,m}(1) \end{pmatrix}$$

ce qui est vrai pour $n = 2$.

Supposons que (2.1) soit vrai pour l'entier $n = k$, *alors*

$$Q_{1,m}^k = \begin{pmatrix} L_{1,m}(k+1) & L_{1,m}(k) \\ L_{1,m}(k) & L_{1,m}(k-1) \end{pmatrix}$$

Maintenant, nous pouvons écrire

$$\begin{aligned} Q_{1,m}^{k+1} = \left(Q_{1,m}^k\right)(Q_{1,m}) &= \begin{pmatrix} L_{1,m}(k+1) & L_{1,m}(k) \\ L_{1,m}(k) & L_{1,m}(k-1) \end{pmatrix} \begin{pmatrix} m & 1 \\ 1 & 0 \end{pmatrix} \\ &= \begin{pmatrix} L_{1,m}(k+2) & L_{1,m}(k+1) \\ L_{1,m}(k+1) & L_{1,m}(k) \end{pmatrix} \end{aligned}$$

Par induction, nous pouvons écrire.

$$Q_{1,m}^n = \begin{pmatrix} L_{1,m}(n+1) & L_{1,m}(n) \\ L_{1,m}(n) & L_{1,m}(n-1) \end{pmatrix}$$

Lorsque $p = 2$, *nous devons prouver*

$$Q_{2,m}^n = \begin{pmatrix} L_{2,m}(n+1) & L_{2,m}(n) & L_{2,m}(n-1) \\ L_{2,m}(n-1) & L_{2,m}(n-2) & L_{2,m}(n-3) \\ L_{2,m}(n) & L_{2,m}(n-1) & L_{2,m}(n-2) \end{pmatrix} \tag{2.2}$$

Nous allons le prouver par l'induction mathématique. Pour $n = 1$

$$Q_{2,m} = \begin{pmatrix} m & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} = \begin{pmatrix} L_{2,m}(2) & L_{2,m}(1) & L_{2,m}(0) \\ L_{2,m}(0) & L_{2,m}(-1) & L_{2,m}(-2) \\ L_{2,m}(1) & L_{2,m}(0) & L_{2,m}(-1) \end{pmatrix}$$

ce qui est vrai pour $n = 1$.

Pour $n = 2$

$$Q_{2,m}^2 = \begin{pmatrix} m^2 & m & 1 \\ 1 & 0 & 0 \\ m & 1 & 0 \end{pmatrix} = \begin{pmatrix} L_{2,m}(3) & L_{2,m}(2) & L_{2,m}(1) \\ L_{2,m}(1) & L_{2,m}(0) & L_{2,m}(-1) \\ L_{2,m}(2) & L_{2,m}(1) & L_{2,m}(0) \end{pmatrix}$$

ce qui est vrai pourn $= 2$.

Supposons que (2.2) soit vrai pour l'entier $n = k$, *alors*

$$Q_{2,m}^k = \begin{pmatrix} L_{2,m}(k+1) & L_{2,m}(k) & L_{2,m}(k-1) \\ L_{2,m}(k-1) & L_{2,m}(k-2) & L_{2,m}(k-3) \\ L_{2,m}(k) & L_{2,m}(k-1) & L_{2,m}(k-2) \end{pmatrix}$$

Maintenant, nous pouvons écrire

$$Q_{2,m}^{k+1} = \left(Q_{2,m}^k\right)(Q_{2,m}) = \begin{pmatrix} L_{2,m}(k+1) & L_{2,m}(k) & L_{2,m}(k-1) \\ L_{2,m}(k-1) & L_{2,m}(k-2) & L_{2,m}(k-3) \\ L_{2,m}(k) & L_{2,m}(k-1) & L_{2,m}(k-2) \end{pmatrix} \begin{pmatrix} m & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$$

$$= \begin{pmatrix} L_{2,m}(k+2) & L_{2,m}(k+1) & L_{2,m}(k) \\ L_{2,m}(k) & L_{2,m}(k-1) & L_{2,m}(k-2) \\ L_{2,m}(k+1) & L_{2,m}(k) & L_{2,m}(k-1) \end{pmatrix}$$

Par conséquent, par induction, nous pouvons écrire

$$Q_{2,m}^n = \begin{pmatrix} L_{2,m}(n+1) & L_{2,m}(n) & L_{2,m}(n-1) \\ L_{2,m}(n-1) & L_{2,m}(n-2) & L_{2,m}(n-3) \\ L_{2,m}(n) & L_{2,m}(n-1) & L_{2,m}(n-2) \end{pmatrix}$$

De même, par induction, il peut être prouvé pour toutes les valeurs de p

Théorème 2.6. *[28]* $Q^n_{p,m} = mQ^{n-1}_{p,m} + Q^{n-(p+1)}_{p,m}$

Preuve 3. *On a*

$$Q^n_{p,m} =$$

$$\begin{pmatrix} L_{p,m}(n+1) & L_{p,m}(n) & \dots & L_{p,m}(n-p+2) & L_{p,m}(n-p+1) \\ L_{p,m}(n-p+1) & L_{p,m}(n-p) & \dots & L_{p,m}(n-2p+2) & L_{p,m}(n-2p+1) \\ \vdots & \vdots & \dots & \vdots & \vdots \\ \vdots & \vdots & \dots & \vdots & \vdots \\ \vdots & \vdots & \dots & \vdots & \vdots \\ \vdots & \vdots & \dots & \vdots & \vdots \\ \vdots & \vdots & \dots & \vdots & \vdots \\ L_{p,m}(n-1) & L_{p,m}(n-2) & \dots & L_{p,m}(n-p) & L_{p,m}(n-p-1) \\ L_{p,m}(n) & L_{p,m}(n-1) & \dots & L_{p,m}(n-p+1) & L_{p,m}(n-p) \end{pmatrix}$$

Quand $p = 1$

$$Q^n_{1,m} = \begin{pmatrix} L_{1,m}(n+1) & L_{1,m}(n) \\ L_{1,m}(n) & L_{1,m}(n-1) \end{pmatrix}$$

$$= \begin{pmatrix} mL_{1,m}(n) + L_{1,m}(n-1) & mL_{1,m}(n-1) + L_{1,m}(n-2) \\ mL_{1,m}(n-1) + L_{1,m}(n-2) & mL_{1,m}(n-2) + L_{1,m}(n-3) \end{pmatrix}$$

$$= \begin{pmatrix} mL_{1,m}(n) & mL_{1,m}(n-1) \\ mL_{1,m}(n-1) & mL_{1,m}(n-2) \end{pmatrix}$$

$$+ \begin{pmatrix} L_{1,m}(n-1) & L_{1,m}(n-2) \\ L_{1,m}(n-2) & L_{1,m}(n-3) \end{pmatrix}$$

$$= mQ^{n-1}_{1,m} + Q^{n-2}_{1,m}$$

Quand $p = 2$

$$Q_{2,m}^n = \begin{pmatrix} L_{2,m}(n+1) & L_{2,m}(n) & L_{2,m}(n-1) \\ L_{2,m}(n-1) & L_{2,m}(n-2) & L_{2,m}(n-3) \\ L_{2,m}(n) & L_{2,m}(n-1) & L_{2,m}(n-2) \end{pmatrix}$$

$$= \begin{pmatrix} mL_{2,m}(n) + L_{2,m}(n-2) & mL_{2,m}(n-1) + L_{2,m}(n-3) & mL_{2,m}(n-2) + L_{2,m}(n-4) \\ mL_{2,m}(n-2) + L_{2,m}(n-4) & mL_{2,m}(n-3) + L_{2,m}(n-5) & mL_{2,m}(n-4) + L_{2,m}(n-6) \\ mL_{2,m}(n-1) + L_{2,m}(n-3) & mL_{2,m}(n-2) + L_{2,m}(n-4) & mL_{2,m}(n-3) + L_{2,m}(n-5) \end{pmatrix}$$

$$= \begin{pmatrix} mL_{2,m}(n) & mL_{2,m}(n-1) & mL_{2,m}(n-2) \\ mL_{2,m}(n-2) & mL_{2,m}(n-3) & mL_{2,m}(n-4) \\ mL_{2,m}(n-1) & mL_{2,m}(n-2) & mL_{2,m}(n-3) \end{pmatrix}$$

$$+ \begin{pmatrix} L_{2,m}(n-2) & L_{2,m}(n-3) & L_{2,m}(n-4) \\ L_{2,m}(n-4) & L_{2,m}(n-5) & L_{2,m}(n-6) \\ L_{2,m}(n-3) & L_{2,m}(n-4) & L_{2,m}(n-5) \end{pmatrix}$$

$$= mQ_{2,m}^{n-1} + Q_{2,m}^{n-3}$$

De même, nous pouvons montrer que,

$$Q_{p,m}^n = mQ_{p,m}^{n-1} + Q_{p,m}^{n-(p+1)}$$

Chapitre 3

LE THÉORÈME DE ZECKENDORF

3.1 Théorème de Zeckendorf pour les nombres de Fibonacci

Le Théorème de Zeckendorf, nommé de son fondateur belge Edouard Zeckendorf en 1972, est un théorème pour la représentation des entiers comme somme de nombres de Fibonacci.

Théorème 3.1. *[36] Chaque entier strictement positif n peut être représenté d'une manière unique comme somme de nombres de Fibonacci F_i avec $i \geq 2$ distincts et non consécutifs deux à deux.*
Autrement dit, si n est un entier strictement positif

$$n = \sum_{r=1}^{\infty} e_r F_r$$

où $e_r \in \{0, 1\}, e_r = 1 \Rightarrow e_{r+1} = 0$ et pour un nombre fini de coefficients e_r sont non nuls.

Exemple 3.1. On a $F_2 = 1, F_3 = 2, F_4 = 3, , F_5 = 5, F_6 = 8, F_7 = 13, F_8 = 21,$
$F_9 = 34, F_{10} = 55, alors$
$1 = F_2, 2 = F_3, 3 = F_4, 4 = f_2 + F_4, 5 = F_5, 6 = F_2 + F_5, 7 = F_3 + F_5, 8 = F_6, ...$

Preuve. le Théorème de Zeckendorf a deux parties :

Existence : chaque entier positif n a une représentation de Zeckendorf.

Unicité : chaque entier positif n'a pas deux représentations différentes de Zeckendorf.

La première partie du Théorème de Zeckendorf(existence) peut être prouvée par induction. Pour $n = 1, 2, 3$ elle est vraie, pour $n = 4$ on a $4 = 3 + 1$. Maintenant, supposons que chaque entier a une représentation de Zeckendorf. Si $k+1$ est un nombre de Fibonacci alors c'est fini, sinon il existe j tel que $F_j < k + 1 < F_{j+1}$. Considérons $a = k + 1 - F_j$.

Comme $a < k, a$ a une représentation de Zeckendorf ; de plus $F_j + a < F_{j+1}$ donc $a < F_{j-1}$ donc la représentation de Zeckendorf de a ne contient pas F_{j-1}. Alors $k + 1$ peut être représenté comme $a + F_j$. De plus, il est clair que chaque représentation de Zeckendorf correspond à un seul entier. La seconde partie du Théorème de Zeckendorf (unicité) requiert le lemme suivant :

Lemme 3.2. *La somme des éléments d'un ensemble fini et non vide S de nombres de Fibonacci non-consécutifs deux à deux F_i avec $i \geq 2$,dont le plus grand nombre est F_j strictement inférieur à F_{j+1}.*

Exemple 3.2. Soit $S = \{F_2, F_4, F_7, F_9\}$, alors on a $F_2 + F_4 + F_7 + F_9 < F_{10}$

Preuve. Maintenant, on prend deux ensembles de nombres de Fibonacci distincts, non-consécutifs S et T qui ont la même somme. Éliminons les nombres communs pour former des ensembles S' et T' qui n'ont aucun nombre commun. On veut montrer que S' et T' sont vides i.e $S = T$.

Premièrement, on montre qu'au moins un des ensembles S' et T' est vide. Supposons le contraire, soit F_s le plus grand nombre de S' et F_t le plus grand nombre de T' sans perte de généralité, supposons que $F_s < F_t$. Alors par Lemme 13, la somme de S' est strictement inférieur à F_{s+1}, et donc strictement inférieur à F_t, mais il est claire que la somme de T' est au moins F_t.

Cela veut dire que S' et T' ne peuvent pas avoir la même somme, et alors S et T ne peuvent pas avoir la même somme.

Donc notre supposition est fausse.

Si S' est vide et T' est non vide alors S est un sous ensemble de T, et alors S et T ne peuvent pas avoir la même somme. Similairement on peut éliminer le cas ou S' est non vide et T' est vide. Le seul cas qui reste est que S' et T' sont vides, donc $S = T$.

On conclut que quelque soient deux représentations de Zeckendorf qui ont la même somme doivent être identiques.

3.2 Théorème de Zeckendorf pour les nombres de Lucas

Il est connu que les nombres de Lucas sont complets [6] au sens que chaque entier positif peut être représenté comme somme de nombres distincts de Lucas. En général, ses représentation ne sont pas uniques par exemple $4 = L_3 = L_1 + L_2, 12 = L_1 + L_3 + L_4 = L_0 + L_2 + L_4$.

Avant d'énoncer le Théorème, certains Lemmes sont utiles :

Lemme 3.3. *$L_n - 1 = L_{n-1} + L_{n-3} + L_{n-5} + \ldots + L_{1,2}(n)$, pour $n \geq 2$ où*

$$L_{1,2}(n) = \begin{cases} 2L_1 \text{ si } n \text{ pair} \\ L_2 \text{ si } n \text{ impair} \end{cases}$$

Preuve. on va monter ce Lemme par récurrence sur n :

Si n est pair, posons $n = 2k$

on a $L_2 - 1 = 3 - 1 = 2 = 2L_1$ vérifie le lemme.

Supposons que l'égalité est vraie pour l'ordre n et montrons qu'elle est vraie pour l'ordre $n + 2$. On a

$$L_{2k} - 1 = L_{2k-1} + L_{2k-3} + L_{2k-5} + \ldots + L_3 + 2L_1$$

on rajoutant L_{2k+1} des deux cotés on aura

$$L_{2k+1} + L_{2k} - 1 = L_{2k+1} + L_{2k-1} + L_{2k-3} + \ldots + L_3 + 2L_1$$

or $L_{2k+1} + L_{2k} = L_{2k+2}$ donc

$$L_{2k+2} = L_{2k+1} + L_{2k-1} + L_{2k-3} + \ldots + L_3 + 2L_1$$

ce ci est égale à

$$L_{n+2} = L_{(n+2)-1} + L_{(n+2)-3} + L_{(n+2)-5} + \ldots + L_3 + 2L_1$$

et on a bien l'égalité du lemme.

si n est impaire, posons $n = 2k + 1$.

on a $L_3 - 1 = 4 - 1 = 3 = L_2$, vérifie le lemme.

Supposons que l'égalité est vraie pour l'ordre n et montrons qu'elle est vraie pour l'ordre $n + 2$. On a

$$L_{2k+1} - 1 = L_{2k} + L_{2k-2} + L_{2k-4} + \ldots + L_4 + L_2$$

on rajoutant L_{2k+2} des deux cotés on aura

$$L_{2k+2} + L_{2k+1} - 1 = L_{2k+2} + L_{2k} + L_{2k-2} + L_{2k-4} + \ldots + L_4 + L_2$$

or $L_{2k+2} + L_{2k+1} = L_{2k+3}$, donc

$$L_{2k+3} = L_{2k+2} + L_{2k} + L_{2k-2} + L_{2k-4} + \ldots + L_4 + L_2$$

en remplaçons n par $2k + 1$ dans l'égalité précédente on aura le résultat

$$L_{n+2} = L_{(n+2)-1} + L_{(n+2)-3} + L_{(n+2)-5} + \ldots + L_4 + L_2$$

Donc le lemme est vrai dans les deux cas.

Lemme 3.4. *$L_{n+2} = 1 + \sum_{i=0}^{n} L_i$ pour $n \geq 0$.*

Preuve. On va démontrer ce Lemme par récurrence sur n.

pour $n = 0$ on a $L_2 = 1 + L_1 = 1 + 2 = 3$. L'égalité est vraie.

Supposons qu'elle est vraie pour n est montrons qu'elle est vraie pour $n + 1$.

on rajoutant L_{n+1} des deux cotés de l'égalité on aura

$$L_{n+2} + L_{n+1} = 1 + \sum_{i=0}^{n} L_i + L_{n+1}$$

ce ci est égale à

$$L_{n+3} = L_{(n+1)+2} = 1 + \sum_{i=0}^{n+1} L_i$$

ce qui prouve le Lemme.

Théorème 3.5. *[5] Soit n un entier positif satisfaisant $0 \leq n \leq L_k$ pour $k \geq 1$, alors*

$$n = \sum_{0}^{k-1} \alpha_i L_i$$

où $\alpha_i \in \{0, 1\}$ tels que

$$\begin{cases} \alpha_i \alpha_{i+1} = 0 \text{ pour } i \geq 0 \\ \alpha_0 \alpha_2 = 0 \end{cases}$$

Cette représentation est unique.

Preuve. La preuve va ce décomposer en deux parties : existence et unicité de la représentation.

Existence : Supposons que n a aussi cette représentation $n = \sum_0^{k-1} \gamma_i L_i$ avec $\gamma_i \in \{0, 1\}$, $\gamma_i \gamma_{i+1} = 0$ et $\gamma_0 \gamma_2 = 0$.

Supposons pour une preuve par contradiction que les deux représentations ne sont pas identiques,

$$\sum_{0}^{\infty} |\gamma_i - \alpha_i| \neq 0$$

alors, soit k la plus grande valeur de i tel que $\gamma_i \neq \alpha_i$. Plus précisément $k \geq 2$, et comme $\gamma_k \neq \alpha_k$, on peut supposer, sans perte de généralité, que $\alpha_k = 1, \gamma_k = 0$.

Pour un $m \leq n$,

$$m = \sum_{0}^{k} \alpha_i L_i = \sum_{0}^{k-1} \gamma_i L_i, \text{ avec } \alpha_k = 1$$

Alors

$$\sum_{0}^{k} \alpha_i L_i \geq L_k$$

à partir des contraintes sur les coefficients (γ_i),

$$\sum_{0}^{k-1} \gamma_i L_i \leq L_{k-1} + L_{k-3} + \ldots + L_{1,2}(k) = L_k - 1$$

Ainsi $m \geq L_k$ tant que $m \leq L_k - 1$, contradiction.

Unicité : Supposons que n à deux représentations

$$n = \sum_{0}^{k} \beta_i L_i = \sum_{0}^{k-1} \gamma_i L_i \tag{3.2.1}$$

où $\beta_i, \gamma_i \in \{0, 1\}$ tel que $\beta_k = \gamma_m = 1, \beta_i + \beta_{i+1} \neq 0$. Pour $0 \leq i \leq k-2$:

$$\beta_0 + \beta_2 \neq 0, \gamma_i + \gamma_{i+1} \neq 0$$

et pour $0 \leq i \leq m-2$

$$\gamma_0 + \gamma_2 \neq 0$$

Sans perte de généralité, on prend $m \geq k \geq 2$. Si $m > k$ alors la représentation à droite en (3.2.1) avec les contraintes de constantes implique

$$n \geq \begin{cases} L_m + L_{m-2} + \ldots + L_2 + L_1 = L_{m+1} \geq L_{k+2} & \text{(m pair)} \\ L_m + L_{m-2} + \ldots + L_3 + L_1 + L_0 = L_{m+1} \geq L_{k+2} & \text{(m impair)} \end{cases}$$

mais

$$n = \sum_{0}^{k} \beta_i L_i \leq \sum_{0}^{k} L_i = L_{k+2} - 1$$

contradiction. Donc $m = k$ dans (3.2.1).

3.3 Méthode de la décomposition de Zeckendorf :

Pour décomposer un entier x de la forme de Zeckendorf $x = \sum_{r=1}^{\infty} e_r F_r$ il suffit de suivre les étapes suivantes :

1. Trouver le plus grand nombre de Fibonacci $F_r \leq x$;
2. Effectuer la soustraction $X = x - F_r$; affecter un '1' à e_r et conserver ce coefficient ;
3. Affecter X à x et répéter les étapes 1 et 2 jusqu'à avoir un X nul ;
4. Affecter des 0 aux e_i où $0 < i < r$ et $e_i \neq 1$.

Le résultat de cette décomposition est : un vecteur de r éléments qui contient les coefficients e_r de la décomposition.

3.4 Généralisation du Théorème de Zeckendorf

Pour généraliser le Théorème de Zeckendorf, ces deux Lemmes sont essentiels dans la preuve .

Lemme 3.6.

$$\sum_{r=0}^{k} 2^r = 2^{k+1} - 1$$

Lemme 3.7. *Soit (g_n) une suite de Fibonacci généralisé d'ordre k, alors pour tout entier $i, g_{i+1}^{(k)} < 2g_i^{(k)}$.*

Preuve. Par la récurrence de Fibonacci on a :

$$2g_i^{(k)} = g_i^{(k)} + g_{i-1}^{(k)} + g_{i-2}^{(k)} + \ldots + g_{i-(k-1)}^{(k)} + g_{i-k}^{(k)}$$

Ceci implique que

$$2g_i^{(k)} = g_{i+1}^{(k)} + g_{i-k}^{(k)}$$

aussi

$$g_{i+1}^{(k)} < 2g_{i+1}^{(k)}$$

Maintenant, On va prouver un théorème analogue au tour de la représentation de Zeckendorf pour les suites de Fibonacci d'ordre k.

Théorème 3.8. *[8] Considérons l'ensemble des suites récurrentes linéaires d'ordre k $\left(U_n^{(k)}\right)_n$ satisfaisant la récurrence de Fibonacci généralisée d'ordre k telles que $U_0^{(k)} < U_1^{(k)} < \ldots < U_{k-2}^{(k)} < U_{k-1}^{(k)}$. Dans cet ensemble il existe une unique suite récurrente linéaire (U_n) d'ordre k généralisé tel que pour tout entier positif n on a une unique représentation de Zeckendorf d'ordre k.*

Preuve. En premier temps, on va démontrer par récurrence l'existence d'une unique représentation de Zeckendorf d'ordre k pour tout entier positif x. Il est claire que $\forall m \in \mathbb{N}, 0 \leq m \leq 2^k - 1$.

La représentation des entiers m tel que $0 \leq m \leq 2^k - 2$ correspond à la représentation en binaire de ces nombres. Par le lemme1 $2^0 + 2^1 + \ldots + 2^{k-1} = 2^k - 1$, donc la représentation de $2^k - 1$ est $g_k^{(k)}$.

Supposons que pour tout j tel que $0 \leq j \leq x$ il existe une unique représentation de Zeckendorf d'ordre k de x. Soit i le plus grand entier tel que $g_i^{(k)} \leq x$. On sait que ce i existe car les termes de (g_n) sont croissantes. Soit $x - g_i^{(k)} = g_{a_1}^{(k)} + g_{a_2}^{(k)} + \ldots + g_{a_r}^{(k)}$ une unique représentation de $x - g_i^{(k)}$, alors il existe une représentation de $x = g_i^{(k)} + g_{a_1}^{(k)} + g_{a_2}^{(k)} + \ldots + g_{a_r}^{(k)}$. Cette représentation des de Zeckendorf d'ordre k si $r < k-1$. Maintenant, supposons que $r \geq k-1$. On peut poser $a_1 > a_2 > \ldots > a_r$. On va montrer que $i > a_1$. Si $i < a_1$ alors a_1 est le plus grand entier tel que $g_{a_1}^{(k)} \leq x$ contradiction car i est le plus grand entier tel que $g_{a_1}^{(k)} \leq x$. Supposons que $i = a_1$, alors $x = g_i^{(k)} + g_{a_1}^{(k)} + g_{a_2}^{(k)} + \ldots + g_{a_r}^{(k)}$, le lemme 2 nous dis que cette somme est plus grande que $g_i^{(k)}$ donc $g_{i+1}^{(k)} > x$. Alors $g_i^{(k)} + g_{a_1}^{(k)} + g_{a_2}^{(k)} + \ldots + g_{a_r}^{(k)}$ est une représentation de Zeckendorf d'ordre k de x. Supposons qu'il existe une autre représentation de Zeckendorf d'ordre k de x qui ne contient pas g_i, alors la valeur maximale de cette représentation est

$$F(i) = \sum_{\substack{1 \leq j \leq i \\ k \nmid j}} g_{i-j}^{(k)}$$

On utilise une récurrence sur i pour démontrer $F(i) = g_i - 1 < x$ pour tout i. Par le lemme 1 , on sait que c'est vrai pour le cas de la basse $i \leq k-1$. Supposons que $i \geq k$ et que $F(l) = g_l^{(k)} - 1$ pour tout $l < i$. En particulier, $F_{i-k} = g_{i-k} - 1$,

$$\begin{aligned} F_i &= g_i^{(k)} + g_{i-1}^{(k)} + g_{i-2}^{(k)} + \ldots + g_{i-(k-1)}^{(k)} + F(i-k) \\ &= g_i^{(k)} - g_{i-k}^{(k)} + \left(g_{i-k}^{(k)} - 1 \right) \\ &= g_i^{(k)} - 1 < x. \end{aligned}$$

donc la représentation de Zeckendorf d'ordre k est unique. On montre maintenant l'unicité de (g_n). Supposons pour un ordre k arbitraire, il existe une autre suite de Fibonacci généralisé (h_n) avec $h_0 < h_1 < \ldots < h_{k-1}$ tel que chaque entier positif x à une unique représentation de Zeckendorf d'ordre k qui respecte (h_n). Alors il reste au moins un entier q dans l'intervalle $[0, k-1]$ tel que $g_q \neq h_q$. Si $h_q < g_q$ alors h_q a plus qu'une représentation de Zeckendorf d'ordre k dans (h_n). Les deux représentations sont h_q et la représentation en binaire de h_p dans $h_0, h_1, \ldots, h_{q-1}$. Si $h_p > g_q$ alors 2^q n'a pas une représentation de Zeckendorf d'ordre k dans (h_n). Comme $h_q > g_q^{(k)} = 2^q; 2^q$ a aucune représentation. Donc

notre conjoncture que (h_n) existe est fausse et (g_n) est unique.

Chapitre 4

Éléments de l'arithmétique de Zeckendorf pour les nombres de Lucas

Ce chapitre a fait l'objet d'une publication dans le journal international "Palestine Journal of Mathematics" (voir [7]).
Dans ce chapitre, nous présentons de nouveaux algorithmes d'addition, de soustraction, de multiplication et de division de deux entiers positifs en utilisant la forme de Zeckendorf et nous calculerons leur complexité .

4.1 Introduction

Il existe peu de travaux antérieurs dans ce domaine. Graham, Knuth et Patashnik (voir [16]) ont discuté de l'addition dans la représentation de Zeckendorf, Fliponi (voir [12]) l'a fait pour l'addition et la multiplication et Freitag philips (voir [14]) pour la soustraction et la division (voir [15]). Ainsi, aucun ouvrage antérieur n'a abordé l'arithmétique comme un tout cohérent, couvrant toutes les principales opérations, y compris la soustraction et la division. Tous ces algorithmes ont été mis en œuvre et testés sur ordinateur. La plupart des algorithmes sont développés par analogie avec les méthodes arithmétiques conventionnelles. Par exemple, la multiplication est effectuée en ajoutant des multiples appropriés du multiplicande, en fonction du schéma binaire sélectionné du multiplicateur. La division utilise une séquence de tests de soustraction, comme dans la division longue normale. Nous utilisons ici des modèles de complexité arithmétique, où le coût est mesuré

par le nombre d'instructions machine exécutées sur un seul processeur.

4.2 Théorèmes de Zeckendorf pour les nombres de Lucas

Les nombres de Lucas sont définis par la formule de récurrence :

$$\begin{cases} L_n = L_{n-1} + L_{n-2}, n \geq 2 \\ L_0 = 2, L_1 = 1. \end{cases}$$

et pour tous $n \geq 0$, nous avons la formule bien connue

$L_n = F_{n+1} + F_{n-1}$ où F_n est le nième nombre de Fibonacci avec $F_{-1} = 1$

Théorème 4.1. *Soit n un entier positif satisfaisant $0 \leq n < L_k$ pour un certain $k \geq 1$, alors*

$$n = \sum_{i=0}^{k-1} \alpha_i L_i$$

où $\alpha_i \in \{0, 1\}$ tels que

$$\begin{cases} \alpha_i \alpha_{i+1} = 0 \text{ pour tout } i \geq 0 \\ \alpha_0 \alpha_2 = 0 \end{cases}$$

Cette représentation est unique.

Preuve : (voir [5])

4.3 Méthode de la décomposition de Zeckendorf :

Pour décomposer un entier x de la forme de Zeckendorf $x = \sum_{r=1}^{\infty} e_r L_r$ il suffit de suivre les étapes suivantes :

1. Trouver le plus grand nombre de Lucas $L_r \leq x$;
2. Effectuer la soustraction $X = x - L_r$; affecter un '1' à e_r et conserver ce coefficient ;
3. Affecter X à x et répéter les étapes 1 et 2 jusqu'à avoir un X nul ;
4. Affecter des 0 aux e_i où $0 < i < r$ et $e_i \neq 1$.

Le résultat de cette décomposition est : un vecteur de r éléments qui contient les coefficients e_r de la décomposition.

Exemple : Décomposition de l'entier 50, ce tableau explique la représentation

Suite de Lucas	2	1	3	4	7	11	18	29	47	76
Vecteur des coefficients e_r	0	0	1	0	0	0	0	0	1	0

$50_L = 001000001$

Proposition 4.2. *Soit $m > 1$, si $L_n \leq m$ alors $n \leq \frac{ln(m-1)}{ln(\varphi)}$, où φ est le nombre d'or.*

Preuve : Soit $L_n = \varphi^n + \frac{1}{\varphi^n} \leq \varphi^n + 1$.

tant que $L_n \leq m$ alors $\varphi^n + 1 \leq m \Rightarrow \varphi^n \leq m - 1$

$\Rightarrow ln(\varphi^n) \leq ln(m-1)$

$\Rightarrow n \leq \frac{ln(m-1)}{ln(\varphi)}$.

Nous concluons que le nombre de bits de Zeckendorf pour la représentation n est au plus égal à $\lfloor \frac{ln(m-1)}{ln(\varphi)} \rfloor$.

4.4 Addition

Étant donné deux entiers positifs a et b écrits sous la forme de Zeckendorf, on peut obtenir la forme de Zeckendorf de a + b en répétant l'addition, en même temps, les nombres de Lucas occurrent dans l'un des deux nombres, disant b, à l'autre nombre est a. Dans chaque étape, quand on ajoute un nombre de Lucas, on exprime la somme partielle dans la forme de Zeckendorf. On commence par l'addition de tous les chiffres $a_i + b_i = z_i$, où $z_i \in \{0, 1, 2\}$. Les valeurs 0 et 1 ne posent aucun problème, car ils sont valides dans représentations Zeckendorf. Comme le 2 ou plusieurs 1 consécutifs sont inadmissibles dans la forme de Zeckendorf, on procède comme suit : Pour $z_i = 2$ on à deux cas :

Si $n = 1$ on à $2L_1 = L_0$, nous remplaçons 020 par 001 .

Si $n \geq 2$ on à $2L_n = L_{n+1} + L_{n-2}$, nous remplaçons 00200 par 01001. De façon équivalente au modèle x 2 y z Les chiffres se transforment en $(1 + x)0y(1 + z)$.

Si la combinaison 011 existe dans le vecteur e_r, nous le remplacerons par 001. Cette étape doit être réalisée en parcourant le vecteur de composantes e_r de gauche à droite.

Voici un tableau qui résume tous les cas possibles de l'addition dans la représentation de Zeckendorf :

Addition	Lucas poids	L_{i+1}	L_i	L_{i-1}	L_{i-2}
consécutifs 1		x	y	1	1
	devient	x	$y+1$	0	0
Éliminer a 2	ici $x \geq 2$	w	x	y	z
	devient	$w+1$	$x-2$	y	$z+1$
ajouter, bits à droite		L_2	L_1	L_0	
$d_2 \geq 2$	ici $x \geq 2$		x	0	
	devient		0	1	
$d_2 \geq 2$			0	x	
	devient	1	1	0	

TABLE 4.1 – Ajustements et corrections dans l'addition

Théorème 4.3. *La complexité de l'algorithme d'addition est* $O(ln(a))$.

Preuve : Soit a et b deux entiers tels que $0 < b \leq a$. Nous prenons n et n' le numéro de bit Zeckendorf pour la représentation de a et b respectivement. L'addition $(a+b)$ coûte $O(Max(n, n^{'}))$. Le nombre total $T(a)$ de l'opération est donné par :

$$T(a) = O(\frac{ln(a-1)}{ln(\varphi)}) = O(ln(a)).$$

Ce tableau est un exemple d'une somme de 33+19 dans la représentation de Zeckendorf :

a		1	0	0	0	1	0	0	0	$= 33$
b			1	0	0	0	0	1	0	$= 19$
somme initiale		1	1	0	0	1	0	1	0	$= 52$
consécutifs 1	1	0	0	0	0	1	0	1	0	$= 52$
devient	1	0	0	0	0	1	0	1	0	$= 52$
vérifier $33 + 19 = 52$										

TABLE 4.2 – Exemple de l'addition $(33 + 19)$

Ce tableau est un exemple d'une somme de 12+19 dans la représentation de Zeckendorf :

a			1	0	0	0	1	0	= 12
b		1	0	0	0	0	1	0	= 19
somme initiale		1	1	0	0	0	2	0	= 31
		1	1	0	0	0	0	1	= 31
consécutifs 1		1	1	0	0	0	0	1	= 31
	1	0	0	0	0	0	0	1	= 31
devient	1	0	0	0	0	0	0	1	= 31
vérifier $12 + 19 = 31$									

TABLE 4.3 – Exemple de l'addition $(12 + 19)$

4.5 Soustraction

Pour la soustraction, $a - b = z$, où $b < a$ et Z la différence.On commence par soustraire tous les chiffres $a_i - b_i = z_i$, où $z_i \in \{0, 1. - 1\}$. Les valeurs 0 et 1 ne posent aucun problème, car ils sont valides dans représentations Zeckendorf.
Le cas $z_i = -1$ est le plus difficile. Si on est dans ce cas, on passe le 1 au bit suivant que l'on écrit avec la règle de lucas 100... $\rightarrow$ 011... et on répète l'écriture du bit 1, le plus à droite des paires de 1.

$$1000... \rightarrow 0110... \rightarrow 001011... \rightarrow 00101011...$$

jusqu'à ce que le bit 1 coïncide avec le bit -1 dans la même position et on l'élimine en mettant 0 dans la case correspondante puis on passe aux autres 1 consécutifs.

Soustraction Lucas poids	L_{i+2}	L_{i+1}	L_i	L_{i-1}	L_{i-2}
éliminer -1	1	0	0	0	-1
	0	1	1	0	-1
devient	0	1	0	1	0

TABLE 4.4 – Ajustements et corrections dans la soustraction

Théorème 4.4. *La complexité de l'algorithme de soustraction est* $O(ln(a))$.

Preuve : Soit a et b deux entiers tels que $0 < b \leq a$. on prend n et n' le nombre de bits de Zeckendorf pour la représentation de a et b respectivement .La soustraction $(a-b)$ coût $O(Max(n, n^{'}))$. Le nombre total $T(a)$ d'opérations est donné par :

$$T(a) = O(\frac{ln(a-1)}{ln(\varphi)}) = O(ln(a)).$$

Ce tableau montre l'exemple 42-32 dans la représentation de Zeckendorf.

a	1	0	1	0	0	0	0	1	= 42
b	1	0	0	0	0	1	0	0	= 32
soustraire bit par bit			1	0	0	-1	0	1	= 10
récrire 1000				1	1	-1	0	1	= 10
récrire 0110, annulation -1				1	0	0	1	1	= 10
consécutifs 1				1	0	1	0	0	= 10
devient				1	0	1	0	0	= 10

TABLE 4.5 – Exemple de soustraction $(42 - 32)$

4.6 Multiplication

En utilisant les propositions suivantes, on peut développer une méthode de multiplication d'entiers dans la représentation de Zackendorf.

Proposition 4.5. *Si $n \geq 3$, alors*

$$L_k L_{k+n} = \begin{cases} F_{n-1} + F_{n+1} + F_{2k+n\pm1} & (k \ \ pair), \\ F_{n-2} + F_{n+1} + F_{2k+n+1} + \sum_{j=1}^{k-2} F_{2j+n+2} & (k \geq 3, impair). \end{cases}$$

Proposition 4.6. *Si $n \geq 5$, alors*

$$2L_k L_{k+n} = \begin{cases} F_{n\pm3} + F_{2k+n\pm3} & (k \geq 4, pair), \\ F_{n-4} + F_{2k+n+3} + \sum_{j=1}^{3} F_{2j+n-3} + \sum_{j=1}^{k-4} F_{2j+n+4} & (k \geq 5, impair). \end{cases}$$

Proposition 4.7. *Si $n \geq 5$, alors*

$$3L_kL_{k+n} = \begin{cases} \sum_{j=1}^{4} (F_{2j+n-5} + F_{2j+2k+n-5}) & (k \geq 4, pair), \\ F_{n-4} + F_{n+3} + \sum_{j=1}^{3} F_{2j+2k+n-3} + \sum_{j=1}^{k-4} F_{2j+n+4} & (k \geq 5, impair). \end{cases}$$

Proposition 4.8. *Si $n \geq 6$, alors*

$$4L_kL_{k+n} = \begin{cases} \sum_{j=1}^{4} (F_{3j+n-8} + F_{3j+2k+n-8}) & (k \geq 6, pair) \\ F_{n-4} + F_{n-2} + F_{n+1} + \sum_{j=1}^{3} F_{3j+2k+n-5} + \sum_{j=1}^{k-5} F_{2j+n+4} & (k \geq 5, impair) \end{cases}$$

Preuve voir [13]

Théorème 4.9. *La complexité de l'algorithme de multiplication est $O(a\,ln(a))$.*

Preuve : On a $ab = \underbrace{a + a + ...a}_{b\ fois}$ l'addition $a + a$ coût $O(ln(a))$, alors les additions coûtent $O(b\,ln(a))$, mais $b \leq a$, ainsi :

$$T(a) = O(a\,ln(a)).$$

Cet exemple montre comment calculer 17×10 dans la représentation de Zeckendorf :

a					1	0	1	0	0	1	=17
b						1	0	1	0	0	=10
Multiple de Luca 17											
multiple L_1					1	0	1	0	0	1	=17
multiple L_2			1	0	0	0	0	1	0	0	0	=51
multiple L_3			1	0	1	0	0	0	1	0	0	=68
multiple L_4		1	0	1	0	1	0	0	1	0	0	=119
multiple L_5	1	0	1	0	0	1	0	1	0	0	1	=187
Accumuler approprié multiples											
Ajouter multiple de L_2			1	0	0	0	0	1	0	0	0	=51
Ajouter multiple de L_4		1	0	1	0	1	0	0	1	0	0	=119
$L_2 + L_4$=		1	1	1	0	1	0	1	1	0	0	=170
Éliminer 1 consécutifs	1	0	0	1	0	1	1	0	0	0	0	=170
Éliminer 1 consécutifs	1	0	0	1	1	0	0	0	0	0	0	=170
devient=	1	0	1	0	0	0	0	0	0	0	0	=170

TABLE 4.6 – Exemple de la multiplication (17×10) dans la représentation de Zeckendorf

4.7 Division

En utilisant la proposition suivante, on peut dériver une méthode de division d'entiers dans la représentation de Zackendorf.

Proposition 4.10. *Pour $k = 4m$ et n impair, on obtient*

$$\frac{F_{kn}}{F_n} = \sum_{r=1}^{m} \left(L_{(k-4r+3)n} + L_{(k-4r+1)n} \right),$$

et donc

$$\frac{F_{kn}}{F_n} = S_{k,n},$$

où

$S_{k,n} = \sum_{r=0}^{[k/4]-1} (F_{(k-4r-1)n+1} + (\sum_{s=1}^{n-2} F_{(k-4r-1)n-2s}) + F_{(k-4r-3)n+1} + F_{(k-4r-3)n-2})$.

Nous procédons de la même manière pour les autres cas, où n est impair et $k \equiv 1, 2$ et $3 \mod 4$. Dans chaque cas, la partie "la plus significative" de la forme de Zeckendorf est $S_{k,n}$. La forme précise de Zackendorf est

$$\frac{F_{kn}}{F_n} = S_{k,n} + e_{k,n},$$

où la partie la moins significative de la somme de Zeckendorf est

$$e_{k,n} = \begin{cases} 0, & k \equiv 0 \mod 4), \\ F_2, & k \equiv 1 \mod 4), \\ F_{n+1} + F_{n-1}, & k \equiv 2 \mod 4), \\ F_{2n+1} + \sum_{r=1}^{n-1} F_{2n-2r}, & k \equiv 3 \mod 4). \end{cases}$$

Preuve :(voir [14])

Théorème 4.11. *La complexité de l'algorithme de division est* $O(a\,ln(a))$.

Preuve : (comme dans le **Théorème 4**)

Cet exemple montre comment calculer $250 \div 17$ dans la représentation de Zeckendorf :

a		1	0	1	0	1	0	0	0	1	0	0	=250
b							1	0	1	0	0	1	=17
Faire Multiples lucas de diviseur													
multiple L_1							1	0	1	0	0	1	=17
multiple L_2				1	0	0	0	0	1	0	0	0	=51
multiple L_3				1	0	1	0	0	0	1	0	0	=68
multiple L_4			1	0	1	0	1	0	0	1	0	0	=119
multiple L_5		1	0	1	0	0	1	0	1	0	0	1	=187
multiple L_6	1	0	1	0	1	0	0	0	0	0	0	1	=306
Faire soustraction													
$L5$ résidu=				1	0	0	1	0	1	0	1	0	=63
$L2$ résidu=							1	0	0	0	1	0	=12
quotient=							1	0	1	0	0	1	=17
reste =							1	0	0	0	1	0	=12

TABLE 4.7 – Exemple de la division $(250 \div 17)$ dans la représentation de Zeckendorf

Chapitre 5

Suites Récurrentes Linéaires et Applications au Codage

Le code de Fibonacci produit est un code préfixe et universel. Dans ce code, la séquence 11 apparaît uniquement en fin de chaque nombre encodé, et sert ainsi de délimiteur. Le lecteur peut consulter [25] , [29] , [32] , pour de plus amples informations sur ce sujet.

5.0.1 Principe

5.0.2 Encodage d'un mot

L'étape du codage se base sur la décomposition de Zeckendorf en prenant le vecteur des coefficients de cette représentation comme code de l'entier et puis on rajoute un préfixe à la fin du mot du code.

Pour encoder un entier X :

1. écrire x de la forme de Zeckendorf $X = \sum_{r=1}^{\infty} e_r F_r$,
2. extraire le vecteur des e_r,
3. ajouter un préfixe (ajouter un bit ' 1 ' a la fin du vecteur).

Exemple 5.1. Décomposition de l'entier 50 :

$50 = 34 + 13 + 3$, ce tableau explique la représentation

Suite de Fibonacci	1	2	3	5	8	13	21	34
Vecteur des coefficients e_r	0	0	1	0	0	1	0	1
Ajouter préfixe	001001011							

donc le code de Fibonacci de l'entier 50 est 001001011.

5.0.3 Décodage d'un mot

Pour effectuer l'opération inverse, il suffit de supprimer le dernier ' 1 ' de mot du code, puis de reporter les ' 0 ' et les ' 1 ' au fur et à mesure qu'on les rencontre dans représentation de Zeckendorf. On multiplie chaque coefficient non nul e_r par son nombre fibonacci F_r correspondant puis on effectue la somme.
Donc pour décoder un mot du code X :

1. supprimer le préfixe (le '1' qui se trouve à la fin du mot du code),
2. multiplier les coefficients e_r par leurs nombres de Fibonacci F_r correspondants,
3. sommer les e_iF_i.

Exemple 5.2. Décoder le mot 10001010011.

On enlève le dernier '1' puis on reporte les '0' et les '1' restants dans le tableau suivant :

Suite de Fibonacci	1	2	3	5	8	13	21	34	55	89
vecteur des coefficients e_r	1	0	0	0	1	0	1	0	0	1
résultat	$1+8+21+89=119$									

donc le mot 10001010011 désigne l'entier 119 selon le codage de Fibonacci.

5.0.4 Code de Fibonacci d'ordre $k \geq 2$

Les étapes du codage et du décodage se font d'une façon similaire à celle décrite ci-dessous, seul l'ordre va changer. On se base sur la formule récurrente des nombres de Fibonacci généralisés.

$$\begin{cases} g_n^{(k)} = g_{n-1}^{(k)} + g_{n-2}^{(k)} + \ldots + g_{n-k}^{(k)} \text{ , } \forall n \geq 1 \\ g_0^{(k)} = 1; \\ g_{-n}^{(k)} = \ldots = g_{-2}^{(k)} = g_{-1}^{(k)} = 0 \end{cases}$$

La décomposition d'un entiers selon la représentation de Zeckendorf donne un vecteur des coefficients où il n'y a pas k ' 1 ' consécutifs. Ainsi pour obtenir le code souhaité on ajoute $k-1$ ' 1 ' à la fin voir [21] du mot du code comme préfixe. Ce tableau contient les codes de Fibonacci des entiers de 1 à 20 pour les ordres 2,3 et 4

index	fib2	fib3	fib4
1	11	111	1111
	011	0111	01111
	0011	00111	001111
4	1011	10111	101111
6	00011	000111	0001111
6	10011	100111	1001111
7	01011	010111	0101111
8	000011	110111	1101111
9	100011	0000111	00001111
10	010011	1000111	10001111
11	001011	0100111	01001111
12	101011	1100111	11001111
13	0000011	0010111	00101111
14	1000011	1010111	10101111
15	0100011	0110111	01101111
16	0010011	00000111	11101111
17	1010011	10000111	000001111
18	0001011	01000111	100001111
19	1001011	11000111	010001111
20	0101011	00100111	110001111

5.0.5 Une variante dans le code de Fibonacci

Un cas plus général de la suite de Fibonacci est la suite de Gopala-Hemachandra (GH) définit dans [19] où ses termes sont de la forme :

$$\{a, b, a+b, a+2b, 2a+b, 2a+3b, 3a+5b, \ldots\}$$

Pour pour tous entiers a et b . Pour le cas $a = 1$ et $b = 2$,on retrouve les nombres de Fibonacci :F_2 F_3 F_4 .....

On introduit une variante du code de Fibonacci en utilisant la suite GH pour obtenir un nouveau code similaire au code de Fibonacci. La variante du second ordre de la suite de Fibonacci $VF_a(n)$ est définie par la suite GH où $b = 1 - a$ donc $VF_a(1) = a, (a \in \mathbb{Z}); VF_a(2) = 1 - a$ et pour $n \geq 3$

$$VF_a(n) = VF_a(n-1) + VF_a(n-2)$$

Exemple 5.3. Si on pose $a = -2$, les termes de la suite GH_{-2} sont les suivants :

$$\{-2, 3, 1, 4, 9, 14, 23, \ldots\}$$

Donc pour des différentes valeurs de a, on trouve de différentes suites. Mais ces suites variantes de Fibonacci donnent des représentations multiples de Zeckendorf pour le même entier par exemple James Harlod Tomas a prouvé [8] que pour la suite $VF_{-5} = \{-5, 6, 1, 7, 8, 15, 23, 38, \ldots\}$ il n'y a pas une représentation de Zeckendorf pour les entiers $n = 5, 12$. Une étude à été faite par [13] pour trouver lesquels des entiers de 1 à 100 n'ont pas une représentation de Zeckendorf pour $a \leq -2$. Les résultats sont les suivants :

1. Pour $n = 1, 2, 3, 4$ le code GH existe pour $a = -2, -3, \ldots, -20$.
2. Pour $1 \leq n \leq 100$ il y a au plus m codes non disponibles consécutifs dans $GH_{-(4+m)}$ où $1 \leq n \leq 16$.
3. Pour $1 \leq n \leq 100$, quand m augmente, la disponibilité dans $GH_{-(4+m)}$ diminue où $1 \leq m \leq 16$.

Les codes GH sont plus longs que le code standard de Fibonacci, donc ils sont moins souhaitables. La famille des codes GH ($GH_a(n)$) qui satisfait la condition pour encoder tous les entiers souhaitables $1 \leq n \leq M$ (où M est le nombre de messages sources possibles) nous permet d'avoir plusieurs codes plus universelles à notre disposition lors de la transmission du message.

Et en fin on conclu que ce code est de nature cryptographique haute. Comme les codes

de Gopala-Hemachandra sont déterminées par leurs valeurs initiales a, le codebook peut être facilement modifié à plusieurs reprises lors de la transmission, rendant le décodage beaucoup plus difficile. La présence de la représentation multiple du même entier permet d'avoir un codebook plus grand qu'il ne l'est réellement. Les longueurs des mots de codes ne sont pas toujours croissantes donc sa donne aussi des avantages cryptographiques.

5.1 La Q-matrice de Fibonacci

Méthode du codage/décodage en utilisant la Q-matrice de Fibonacci

Pour coder une information à l'aide de la Q_p-matrice de Fibonacci on suit ces étapes :

5.1.1 Étape d'encodage :

On représente le message initial sous forme d'une matrice carrée M d'ordre $p+1$ où $p = 0, 1.2, 3., ..$

On prend la Q_p^n-matrice de Fibonacci d'ordre $p+1$ comme matrice de codage.

Encoder le message M revient à faire une multiplication $M \times Q_p^n = E$

5.1.2 Étape du décodage

L'étape du décodage se résume à faire l'opération inverse :

étant donné un message E, on effectue la multiplication $E \times Q_p^{-n}$ pour trouver le message initial M.

Exemple 5.4. représentons un message initial de la forme d'une matrice (2×2)

$$M = \begin{pmatrix} m_1 & m_2 \\ m_3 & m_4 \end{pmatrix}$$

Supposons que tous les éléments de la matrice sont des entiers strictement positifs, c'est à dire, $m_1, m_2, m_3, m_4 > 0$. On va maintenant sélectionner pour toute valeur de n une matrice de Fibonacci pour $p = 1$ (puisque $p+1 = 2$) comme matrice de codage. Nous allons simplement écrire pour $n = 2$

$$Q_1^2 = \begin{pmatrix} F_1(3) & F_1(2) \\ F_1(2) & F_1(1) \end{pmatrix} = \begin{pmatrix} 2 & 1 \\ 1 & 1 \end{pmatrix}$$

L'inverse de Q_1^2 est donnée par $Q_1^{-2} = \begin{pmatrix} F_1(1) & -F_1(2) \\ -F_1(2) & F_1(3) \end{pmatrix} = \begin{pmatrix} 1 & -1 \\ -1 & 2 \end{pmatrix}$. Ainsi le codage du message M consiste à faire la multiplication de suivante :

$$M \times Q_1^2 = \begin{pmatrix} m_1 & m_2 \\ m_3 & m_4 \end{pmatrix} \begin{pmatrix} 2 & 1 \\ 1 & 1 \end{pmatrix} = \begin{pmatrix} 2m_1 + m_2 & m_1 + m_2 \\ 2m_3 + m_4 & m_3 + m_4 \end{pmatrix} = \begin{pmatrix} e_1 & e_2 \\ e_3 & e_4 \end{pmatrix} = E$$

où $e_1 = 2m_1 + m_2, e_2 = m_1 + m_2, e_3 = 2m_3 + m_4, e_4 = m_3 + m_4$.

Ensuite, le message codé $E = e_1, e_2, e_3, e_4$ est envoyé par un canal. Le décodage du message E est réalisé de la manière suivante :

$$\begin{pmatrix} e_1 & e_2 \\ e_3 & e_4 \end{pmatrix} \begin{pmatrix} 1 & -1 \\ -1 & 2 \end{pmatrix} = \begin{pmatrix} e_1 - e_2 & -e_1 - 2e_2 \\ e_3 - e_4 & -e_3 + 2e_4 \end{pmatrix} = \begin{pmatrix} m_1 & m_2 \\ m_3 & m_4 \end{pmatrix} = M$$

5.1.3 Connections entre les éléments de la matrice

considérons le cas $p = 1$ le plus simple du codage/décodage . On a :

$$E = M \times Q^n = \begin{pmatrix} m_1 & m_2 \\ m_3 & m_4 \end{pmatrix} \times \begin{pmatrix} F_{n+1} & F_n \\ F_n & F_{n-1} \end{pmatrix} = \begin{pmatrix} e_1 & e_2 \\ e_3 & e_4 \end{pmatrix}.$$

$$M = E \times Q^{-n} = \begin{pmatrix} e_1 & e_2 \\ e_3 & e_4 \end{pmatrix} \times Q^{-n}.$$

Pour n=2k+1 : $M = \begin{pmatrix} e_1 & e_2 \\ e_3 & e_4 \end{pmatrix} \times \begin{pmatrix} -F_{n+1} & F_n \\ F_n & -F_{n+1} \end{pmatrix}$, alors on a les égalités suivantes :

$$\begin{cases} m1 = -F_{n+1}e_1 + F_n e_2, \\ m2 = F_n e_1 - F_{n+1}e_2, \\ m3 = -F_{n+1}e_3 + F_n e_4, \\ m4 = F_n e_3 - F_{n+1}e_4, \end{cases}$$

Puisque les éléments de M sont strictement positifs donc on peut écrire les inégalités suivantes :

$$\begin{cases} -F_{n+1}e_1 + F_n e_2 > 0 \\ F_n e_1 - F_{n+1}e_2 > 0 \\ -F_{n+1}e_3 + F_n e_4 > 0 \\ F_n e_3 - F_{n+1}e_4 > 0 \end{cases}$$

On obtient $\frac{Fn+1}{Fn} < \frac{e_1}{e_2} < \frac{Fn}{Fn-1}$ et $\frac{Fn+1}{Fn} < \frac{e3}{e_4} < \frac{Fn}{Fn-1}$, comme le rapport de deux nombres de Fibonacci adjacents se ramène au nombre d'or donc

$$\begin{cases} e_1 \approx \tau e_2 \\ e_3 \approx \tau e_4 \end{cases} \tag{5.1.1}$$

où $\tau = \frac{1+\sqrt{5}}{2}$ est le nombre d'or.

Par analogie, pour le cas $n = 2k$ on peut avoir les mêmes égalités qui relient e_1 à e_2 et e_3 à e_4, on peut aussi trouver des identités pareilles pour un p quelconque

5.1.4 Détection et correction des erreurs

Pour le cas le plus simple $p = 1$, il a été prouvé dans [29] que la capacité de correction de cette méthode est $93,33\%$ qui dépasse pratiquement tous les codes bien connus. La détection et la correction d'erreurs est basée sur la propriété du déterminant de la matrice E. Si le déterminant de la matrice transmit est le même que la matrice reçu alors E est transmit dans un canal sans erreurs, sinon le message contient des erreurs. On essaye de corriger les erreurs en utilisant (5.1.1).

Première hypothèse : on a une seule erreur dans le message reçu. Il est clair qu'il y a quatre possibilités (l'erreur peut être dans e_1, e_2, e_3 ou e_4), on note (x, y, z, t) respectivement les messages qui contiennent des erreurs. On a les équations algébriques suivantes :

$$\begin{cases} xe_4 - e_2e_3 = (-1)^n \operatorname{Det} M \\ e_1e_4 - ye_3 = (-1)^n \operatorname{Det} M \\ e_1e_4 - e_2z = (-1)^n \operatorname{Det} M \\ e_1t - e_2e_3 = (-1)^n \operatorname{Det} M \end{cases} \tag{5.1.2}$$

donc

$$\begin{cases} x = \frac{(-1)^n \operatorname{Det} M + e_2e_3}{e_4} \\ y = -\frac{(-1)^n \operatorname{Det} M + e_1e_4}{e_3} \\ z = -\frac{(-1)^n \operatorname{Det} M + e_1e_4}{e_2} \\ t = \frac{(-1)^n \operatorname{DetM} M + e_2e_3}{e_1} \end{cases} \tag{5.1.3}$$

Ces formules donnent quatres possibilités d'une erreur mais on doit choisir la variante correcte parmi ces cas de solutions entières x, y, z, t. En plus, on doit choisir une qui satisfait (5.1.1). Si le calcul des formules (5.1.3) ne donne pas une solution entière, on doit conclure que l'hypothèse d'une seule erreur est incorrecte ou bien on a une erreur dans Det M (pour ce dernier cas on peut utiliser (5.1.1) pour vérifier l'exactitude E). Par analogie on peut vérifier tout les hypothèses de double erreur dans la matrice .Par exemple considérons ce cas :

Exemple 5.5. $E = \begin{pmatrix} x & y \\ e_3 & e_4 \end{pmatrix}$, on utilisant (5.1.2) on peut écrire

$$xe_4 - ye_3 = (-1)^n \operatorname{Det} M \tag{5.1.4}$$

mais selon (5.1.1) il y a cette relation entre x et y :

$$x \approx \tau y \tag{5.1.5}$$

donc il faut choisir parmi les solutions de (5.1.4) celle qui satisfait la relation (5.1.5).

Par cette méthode, on peut corriger tous les doubles et triples erreurs possibles. Puisque il y a 15 cas d'erreurs possibles cette méthode peut corriger 14 cas (les cas d'une,deux ou trois erreurs) donc la capacité de correction est égale à :

$$S_{\text{cor}} = \frac{14}{15} = 0.9333 = 93.33\%.$$

5.1.5 Déterminant de la matrice du code

La matrice du code E est définie par la formule

$$E = M \times Q_p^n$$

et de la théorie des matrices on a

$$\operatorname{Det} E = \operatorname{Det}\left(M \times Q_p^n\right) = \operatorname{Det} M \times \operatorname{Det} Q_p^n = \operatorname{Det} M \times (-1)^{pn} \tag{5.1.6}$$

Considérons la détection et la correction pour le cas $p = 2$

$$M = \begin{pmatrix} m_1 & m_2 & m_3 \\ m_4 & m_5 & m_6 \\ m_7 & m_8 & m_9 \end{pmatrix}, \quad Q_2^{-n} = \begin{pmatrix} F_2(n+1) & F_2(n) & F_2(n-1) \\ F_2(n-1) & F_2(n-2) & F_2(n-3) \\ F_2(n) & F_2(n-1) & F_2(n-2) \end{pmatrix},$$

$$E = M \times Q_2^{-n} = \begin{pmatrix} m_1 & m_2 & m_3 \\ m_4 & m_5 & m_6 \\ m_7 & m_8 & m_9 \end{pmatrix} \times \begin{pmatrix} F_2(n+1) & F_2(n) & F_2(n-1) \\ F_2(n-1) & F_2(n-2) & F_2(n-3) \\ F_2(n) & F_2(n-1) & F_2(n-2) \end{pmatrix} = \begin{pmatrix} e_1 & e_2 & e_3 \\ e_4 & e_5 & e_6 \\ e_7 & e_8 & e_9 \end{pmatrix}$$

La matrice E peut contenir une erreur ou deux ou neuf donc le nombre d'erreurs est :

$$2^{(2+1)^2} - 1 = 511$$

dans la matrice E. On ne peut pas corriger le message qui contient neuf erreurs donc la capacité de corriger 8 erreurs est $510/511 = 0,9980 = 99,80\%$.

En général, pour $p = m$ et $n > p + 1 = m + 1$, la capacité de correction de la méthode est

$$\frac{2^{(m+1)^2}-2}{2^{(m+1)^2}-1}$$

Ainsi pour des grandes valeurs de m : $\frac{2^{(m+1)^2}-2}{2^{(m+1)^2}-1} \approx 1 = 100\%$.

5.1.6 Redondance de la méthode du code de Fibonacci

Il y a deux causes qui déterminent la redondance de la méthode du codage. La première est Det M et la seconde est la mesure n de la matrice code.

Comme il a été prouvé dans [30] la redondance relative est donnée par Det M elle est égale à :

$$R = 0.333 = 33.3\%$$

La mesure n de la matrice code inclut sur deux caractéristiques importants de la méthode du codage. Pour des petites valeurs de n, la redondance contribué par la matrice Q^n est négligeable et en général elle est relative à celle de Det M. Ainsi pour les petites valeurs de n l'égalité (5.1.1) devient très approximatives qui diminue les capacités de correction de la méthode. La croissance de la valeur n rend l'égalité (5.1.1) plus précises qui améliore la capacité de correction de la méthode, mais augmente aussi la redondance.

C'est pourquoi le problème du choix de la valeur de n est plus important dans la méthode du codage de Fibonacci.

5.1.7 Comparaison de la méthode du codage de Fibonacci avec la théorie classique du codage :

L'idée principale des codes correcteurs d'erreurs algébriques consiste à une combinaison de codes n-bits redondants où il y a deux types de bits, k bits d'information et m bits pour la vérification formés à partir des bits d'information par addition modulo 2 de certains types de bits d'information. La distance minimale (distance de Hamming) détermine la capacité de détection et de correction d'erreurs d'une multiplicité donnée, cette distance est un paramètre principal du code redondant.

Cependant, il y a deux coefficients plus importants qui déterminent la capacité de détection et correction ce sont :

Le coefficient potentiel de détection d'erreurs $\mathbf{S}_d$: qui est déterminé par le rapport de tous les passages détectables sur tous les passages possibles. Il est déterminé par le nombre m des bits de vérification comme suit :

$$S_d = 1 - \frac{1}{2^m} \tag{5.1.7}$$

Le coefficient potentiel de correction d'erreurs $\mathbf{S}_c$: qui est déterminé par le rapport de tous les passages corrigibles sur tous les passages détectables possibles. Il s'exprime à l'aide du k comme suit

$$S_c = \frac{1}{2^k} \tag{5.1.8}$$

Comme on vu en (5.1.8) le coefficient S_d donne 1 (100%) quand m augmente. Ce fait donne un grand optimisme à l"application pratique des codes algébriques pour détecter les erreurs. Cependant, cet optimisme disparaît quand on calcule la capacité potentielle de ces codes algébriques par la formule (5.1.8), qui dépend des k bits d'information, tend vers 0 rapidement quand k augmente.

Exemple 5.6. considérons le code de Hamming $(15, 11)$ qui garantie la correction d'une seul erreur. Ce code utilise $m = 15 - 11 = 4$ bits de vérification. La capacité de correction de ce code est $S_c = 0.04882\%$ qui est très petite en comparant avec le code de Fibonacci.

On conclut que cette méthode de codage est une application des Q_p-matrices de Fibonacci cette méthode diffère de la forme classique du codage algébrique par :

— La méthode du codage/décodage en utilisant la Q_p-matrice de Fibonacci est réduite à la multiplication des matrices, ie une méthode algébrique bien connue et qui est bien réalisée par les nouvelles machines.

— La méthode simple du codage de Fibonacci pour $p = 1$ peut garantir la correction d'une, double et triples erreurs et la capacité de correction et égale à $93, 33\%$ qui dépasse tous les codes correcteurs connus.

Conclusion et perspectives

Le code universel de Fibonacci permet une compression sana perte et limite la propagation d'erreurs dans un message. En utilisant la variation sur le code universel on a pu construire de nouveaux codes universels de nature cryptographique. La méthode du codage/décodage en utilisant la Qp matrice de Fibonacci est réduite à la multiplication de matrices, ie une méthode algébrique bien connue et qui est bien réalisée par les nouvelles machines. La méthode simple du codage de Fibonacci pour p = 1 peut garantir la correction d'une, double ou triples erreurs et la capacité de correction et égale à $93,33\%$.

Comme futurs pistes de recherche, nous citons la construction d'un cryptosystŁme de Mc Eliece basé sur l'utilisation des codes universels de lucas, on utilisera les différentes matrices nécessaires pour générer les clefs, qui est une autre méthode de codage/décodage utilisant la QL matrice de Lucas. Nous envisageons aussi de résoudre d'autres Équations Diophantiennes quadratiques ou leurs solutions s'expriment en fonction des nombres de Fibonacci et Lucas généralises

Bibliographie

[1] A. Apostolico and A. Fraenkel, Robust transmission of unbounded strings using Fibonacci representations. IEEE Trans. on Information Theory, Vol 33, (1987), pp 238-245.

[2] M. Basu and B. Prasad, Long range variations on the Fibonacci universal code, Journal of Number Theory, vol. 130, no. 9, (2010), pp 1925–1931.

[3] E. R. Berlekamp, R. J. McEliece, and H. C. van Tilborg, "On the inherent intractability of certain coding problems," IEEE Trans. Inform. Theory, vol. 24, no. 3, (May 1978), pp. 384–386.

[4] R.P. Brent and P. Zimmerann, Modern Computer Arithmetic, Cambridge University Press ; (2010), pp 207-221.

[5] J.L. Broun, *Unique Representation of integers as sums of Distinct Lucas Numbers*, The Fibonacci Quarterly, Vol 7, No 3, (1969), pp 243-252.

[6] J. L. Brown, J.r . , Note on Complete Sequences of Int egers, American Mathematical Monthly, Vol. 68, No. 6, (1961), pp 557-560.

[7] R. Chergui, Zeckendorf Arithmetic For Lucas Numbers, Palestine Journal of Mathematics, Vol 9(1), (2020) ,337–342.

[8] R. Chergui, Généralisation du Théorème de Zeckendorf . arXiv 2403.17292V1 .

[9] R. Chergui, Chiffrement avec identité, Éditions universitaires européennes,(2017).

[10] R.J. Mc. Eliece, A public-key cryptosystem based on algebraic coding theory, DSN Progress Report,42-44, (1978), pp. 114–116.

[11] M. Demazure, Cours D'Algèbre : Primalité, Divisibilité, Codes, Cassini, Paris, 1997.

[12] P. Filiponi, *The Representation of Certain Integers as a Sum of Distinct Fibonacci Numbers*, Tech Rep. 2B0985. Fondazione Ugo Bordoni, Rome (1985).

[13] P. Filiponi, E. L.Hart *The Zeckendorf decomposition of certain Fibonacci-Lucas products*,The Fibonacci Quarterly, Vol 36, No 3, (1998) pp 240-247.

[14] H.T. Freitag and G.M. Phillips, *Elements of Zeckendorf arithmetic*, Applications Vol 7, (1998), pp 129-132.

[15] H.T. Freitag and G.M. Phillips, *On the Zeckendorf form of F_{kn}/F_n"*, The Fibonacci Quarterly, Vol 34, No 5, (1996) pp 444-446.

[16] R.L. Graham„ D.E. Knuth„ and O, Patashnik, *Concrete Mathematics*, Addison-Wesley, Reading, MA, (1991), pp 295-297.

[17] P.James Jones, Representation of solutions of Pell equations using Lucas sequences, Acta. Acad. Paed. Agriensis, Sec Math 30 (2003), pp 75-86.

[18] J. Mc Laughlin, B. Sury, Powers of matrix and combinatorial identities, Electronic Journal of Combinatorial Number Theory, A13, Vol 5, (2005).

[19] S. Kak, The golden mean and the physics of aesthetics, Foarm Magazine, arxiv :physics/0411195, Vol 5, (2006),pp 73-81.

[20] P. Kiss, Pell egyenletek megoldása lineáris rekurzív sorozatok segítségével, Acta Acad. Paed Agriensis, Eger, Vol 17 (1984), pp 813-824.

[21] T.S. Klein, M.K Benissan, Using Fibonacci Compression Codes as Alternatives to Dense Codes. Data Compression Conference DCC-(2008), pp 472-481.

[22] S. T. Klein, M. K. BenNissan. "On the Usefulness of Fibonacci Compression Codes",The Computer Journal, Vol53, (2009) 701–716.

[23] K. Liptai and P. Kiss, Solution of Diophantine Equations by second order Annales Univ. Sci. Budapest, Sect. Comp. Vol 18, (1999), pp 109-114

[24] E. Lucas Theorie des fonctions numériques simplement périodiques, American Journal of Mathematics, Vol. 1, N (1878), pp 184-240, 289-321.

[25] B. Manjusri and P. Bandhu, The generalized relations among the code elements for Fibonacci coding theory, Chaos, Solitons and Fractals, Elsevier, vol. 41(5), (2009), pp 2517-2525.

[26] F. Matyás, Second order lineair recurrences and Pell's equations of higher degree, Acta Acad. Paed. Agriensis, Sectio Mathematicae, Vol 29, (2002), pp 47-53.

[27] B. Prasad, Coding theory on Lucas p-numbers, Discrete Mathematics, Algorithms and Applications, Vol 8(4), (2016), pp 74-90.

[28] B. Prasad, m EXTENSION OF LUCAS p-NUMBERS IN INFORMATION THEORY, Jordan Journal of Mathematics and Statistics (JJMS) 12(4), (2019), pp 541 - 556

[29] A. Stakhov, V. Massingue and A. Sluchenkova. Introduction into Fibonacci coding and cryptography, Kharkov, Osnova, (1999).

[30] A. Stakhov, Introduction into algorithmic measurement theory. Moscow, Soviet Radio, (1977).

[31] AP. Stakhov, A generalization of the Fibonacci Q-matrix. Rep Natl Acad Sci Ukraine, Vol 9, (1999),pp 46-49.

[32] A. Stakhov, "Fibonacci matrices, a generalization of the "Cassini formula", and a new coding theory", Chaos, Solitons and Fractals, (2006), pp 56-66

[33] A.P. Stakhov, B. Rozin, Theory of Binet formulas for Fibonacci and Lucas p-numbers, Chaos, Solitons and Fractals, Vol 27,(2006), 1162-1177.

[34] Verner E.HOGATT, JR. and MARJORIE BICKNELL-JOHNSON : A primer for the Fibonacci numbers XVII, Generalized Fibonacci Numbers satisfying $u_{n+1}u_{n-1} - u_n^2 = \pm 1$, The Fibonacci Quarterly, Vol. 13, No. 4 (Apr. 1978), pp 130-137.

[35] L. Wayne McDaniel, Diophantine Representation of Lucas Sequences, Fibonacci Quarterly, Vol 33, (1995), pp 58-63.

[36] E. Zeckendorf, Représentation des nombres naturels par une somme de nombres de Fibonacci ou de nombres de Lucas ,Bull. Soc. R. Sci. Liège Vol. 41, (1972), pp 179 -182.