الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et Populaire
وزارة التعليم العالي والبحث العلمي
Ministère de l'enseignement supérieur et de la recherche scientifique

N° Réf :……

Centre Universitaire Abdelhafid Boussouf – Mila

Institut de Mathématiques et Informatique

Département de Mathématiques et Informatique

Mémoire préparé en vue de l'obtention du diplôme de master en informatique

Spécialité : Intelligence Artificielle et ses Applications (I2A)

Cryptanalyse d'un algorithme de chiffrement par des techniques heuristiques

Préparer Par:

Rafik bougaada

Mohamed islam yakhlaf

Soutenue devant le jury :

Président	:Mr. Selmane Samir	Grade : MAA C.U.Abd Elhafid Boussouf
Examinateur :	Mme.Boufaghes Hamida	Grade : MAA C.U.Abd Elhafid Boussouf
Encadreur	:Mme. Deffas Zineb	Grade : MAA C.U.Abd Elhafid Boussouf

Année Universitaire : 2023/2024

Remerciement

Nous remercions tout d'abord Dieu, le miséricordieux de nous avoir aidé et donné la force et le courage.

Qu'il nous soit permis ici de dire nos gratitudes

Nous tenons à remercier notre encadreur Madame ***Z.deffase*** *qui s'est toujours montré à l'écoute et très disponible tout au long de la réalisation de ce mémoire, ainsi pour l'inspiration, l'aide et le temps qu'elle a bien voulu nous consacrer et sans qui ce mémoire n'aurait jamais vu le jour.*

Nous tenons aussi à exprimer nos remerciements aux membres du jury qui ont accepté d'évaluer notre travail.

Enfin, nous adressons nos plus sincères remerciements à tous ceux qui de prés ou de loin on apporté un effort pour l'élaboration et la mise en forme de ce modeste travail.

Dédicaces

JE DEDIE CE MEMOIRE A …

Mon père

Je te dédie ce travail en témoignage de mon profond amour. Puisse dieu, le tout puissant, te préserver et t'accorder sante, longue vie et bonheur.

Ma très chère mère

Aucune dédicace ne saurait exprimer l'amour, l'estime, le dévouement et le respect que j'ai toujours eu pour vous.
Rien au monde ne vaut les efforts fournis jour et nuit pour mon éducation et mon bien être.

mes sœurs

À vous, qui avez toujours été mon pilier de force et mon soutien indéfectible à travers les hauts et les bas de ce voyage. Merci..

mon frère

À mon petit frère adoré avec tout mon amour et ma fierté,
Je vous dédie ce travail avec tous mes voeux de bonheur, de sante et de réussite.

A tous les membres de ma famille, petits et grands et mes chers amies

Rafik Bougaada

Dédicaces

À tous ceux qui ont contribué au succès de notre opération.

Chers parents, vos plus fidèles supporters. Merci, nos chers frères, pour votre influence et votre soutien

Nous exprimons notre profonde gratitude à tous ceux qui ont conseillé ou fourni des commentaires constructifs sur notre parcours scientifique.

Isalme yekhlef

.

ملخص

تحليل الشفرات، الذي يتمثل في فك تشفير الرسائل المشفرة دون معرفة مفتاح التشفير، له أهمية بالغة في مجال أمن المعلومات. لقد تطورت تقنيات تحليل الشفرات بمرور الوقت، بدءًا من الأساليب التقليدية وصولاً إلى النهج الحديثة المعتمدة على الخوارزميات الاستدلالية. في هذا السياق، تبرز كل من التحليل التكراري والخوارزميات الجينية كأدوات قوية لحل أنواع معينة من التشفيرات، لا سيما التشفيرات بالاستبدال. تهدف هذه الدراسة إلى استكشاف الاستخدام المتكامل للتحليل التكراري والخوارزميات الجينية في فك تشفير التشفيرات بالاستبدال. تتيح هذه المقاربة الهجينة إمكانية كبيرة لحل مشاكل تحليل الشفرات بكفاءة وفعالية، من خلال استغلال الخصائص اللغوية للغات الطبيعية وقدرات التطور في الخوارزميات الجينية.

الكلمات المفتاحية : التحليل التكراري، التشفيرات بالاستبدال، الخوارزميات الاستدلالية

Résumé :

La cryptanalyse, qui consiste à déchiffrer des messages chiffrés sans connaître la clé de chiffrement, est d'une importance cruciale dans le domaine de la sécurité de l'information. Les techniques de cryptanalyse ont beaucoup évolué, passant des méthodes classiques à des approches modernes utilisant des algorithmes heuristiques. Dans ce contexte, l'analyse fréquentielle et les algorithmes génétiques émergent comme des outils puissants pour résoudre certains types de chiffrements, en particulier les chiffrements par substitution. L'objectif de cette recherche est d'explorer en profondeur l'application combinée de l'analyse fréquentielle et des algorithmes génétiques pour déchiffrer les chiffrements par substitution. Cette approche hybride montre un potentiel considérable pour résoudre efficacement et de manière efficiente des problèmes de cryptanalyse, en exploitant à la fois les caractéristiques linguistiques des langues naturelles et les capacités évolutives des algorithmes génétiques.

Les mots clé : Analyse Fréquentielle, Chiffrements par Substitution, algorithmes heuristiques

Abstract :

Cryptanalysis, the art of deciphering encrypted messages without knowing the encryption key, holds paramount importance in the field of information security. Over time, cryptanalysis techniques have evolved from classical methods to modern approaches based on heuristic algorithms. In this context, frequency analysis and genetic algorithms stand out as powerful tools for solving certain types of ciphers, especially substitution ciphers. The objective of this research is to delve deeply into the combined use of frequency analysis and genetic algorithms in decrypting substitution ciphers. This hybrid approach offers significant potential for solving cryptanalysis problems effectively and efficiently, leveraging both the linguistic characteristics of natural languages and the evolutionary capabilities of genetic algorithms.

Keywords: Frequency Analysis, Substitution Ciphers , heuristic approaches

Table des matières

Table des matières

Chapitre I I I : Implémentation et expérimentation

Table des matières

Table des figures

Liste des tables

Introduction générale

Introduction générale

L'échange sécurisé de données, en particulier de texte, a été une préoccupation centrale depuis des siècles. Depuis l'époque de Jules César jusqu'à l'ère numérique moderne, le chiffrement des messages est essentiel pour protéger leur contenu contre tout accès non autorisé. La cryptographie, qui englobe les techniques de chiffrement, est devenue une discipline cruciale. Parallèlement, la cryptanalyse s'est développée pour décrypter les messages chiffrés sans connaissance de la clé déchiffrement. Le chiffre de substitution, par exemple, est apparu comme l'une des premières méthodes de chiffrement, remplaçant chaque lettre du texte original par une autre selon un système préétabli. Malgré sa complexité apparente, ce système a été vulnérable aux techniques d'analyse exploitant les caractéristiques linguistiques, comme dans le cas du français. L'analyse fréquentielle, par exemple, est une méthode fondamentale en cryptanalyse qui exploite les fréquences d'apparition des caractères pour identifier des motifs récurrents et potentiellement découvrir la clé de chiffrement utilisée. Dans ce contexte, nous explorons comment l'analyse fréquentielle des langues naturelles, notamment à travers les fréquences de bigrammes et trigrammes, peut être appliquée pour décrypter efficacement des textes chiffrés par substitution. Nous examinerons également l'efficacité de ces méthodes à travers une implémentation pratique utilisant des techniques d'analyse fréquentielle et d'algorithme génétique pour la résolution de ce problème. Les algorithmes génétiques (AG), faisant partie des techniques évolutionnistes, seront intégrés dans ce processus. Ces méthodes reposent sur une population d'individus et ont été couronnées de succès dans divers contextes pour résoudre des problèmes complexes d'optimisation et de recherche. En combinant l'analyse fréquentielle avec les AG pour évaluer les solutions générées (fitness), nous renforçons l'efficacité du décryptage en exploitant les caractéristiques linguistiques et statistiques des langues naturelles. Pour réussir l'application des AG dans le décryptage, il est crucial de définir précisément le problème d'optimisation, de structurer une population d'individus, de représenter ces individus, d'évaluer leur performance, d'implémenter des opérations d'évolution comme le croisement et la mutation, de sélectionner les individus prometteurs, et d'établir des critères de terminaison basés sur le nombre d'itérations ou les performances atteintes..

Ce projet se compose de trois chapitres :

Le premier chapitre offre une présentation détaillée de la cryptologie, de la cryptographie et de la cryptanalyse.

Le deuxième chapitre explore les méthodes d'optimisation appliquées à la cryptanalyse.
Enfin, le troisième chapitre se concentre sur l'application de l'analyse fréquentielle des langues naturelles, en mettant particulièrement l'accent sur les fréquences de bigrammes et de trigrammes, pour décrypter efficacement des textes chiffrés par substitution cipher .
Cette approche nous permettra d'évaluer l'efficacité des techniques combinées d'analyse fréquentielle et d'algorithmes génétiques dans le décryptage de textes chiffrés. En fin de parcours, nous fournirons une conclusion générale qui résumera les principales découvertes, les implications pratiques de nos résultats, ainsi que des recommandations pour de futures recherches et améliorations dans le domaine de la cryptanalyse.

.

Chapitre 01 :

La cryptanalyse

Chapitre 01 : la cryptanalyse

1 Introduction :

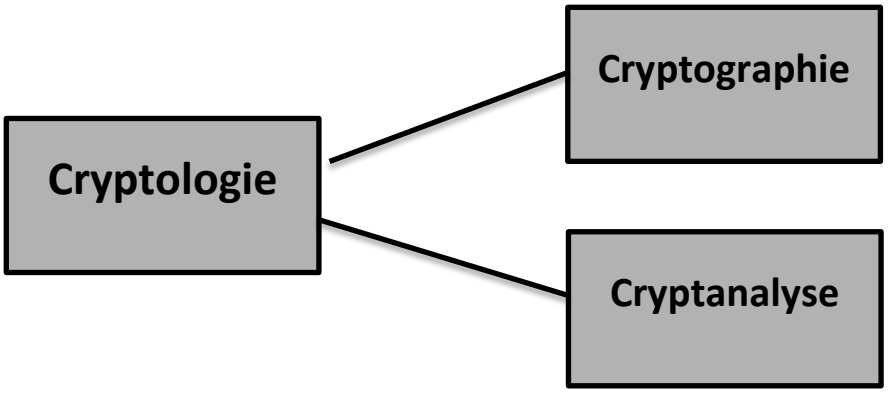
Dans un monde où les communications numériques sont omniprésentes et les données sensibles abondantes, la sécurité de l'information est devenue un enjeu majeur. La cryptologie, un domaine ancien mais en constante évolution, offre des outils essentiels pour protéger la confidentialité, l'intégrité et l'authenticité des données échangées. Au cœur de la cryptologie se trouvent la cryptographie, l'art du chiffrement des messages, et la cryptanalyse, l'art de les déchiffrer sans avoir accès à la clé appropriée.

2 Cryptologie :

La cryptologie, étymologiquement dérivée des mots grecs "kryptós" (caché) et "logos" (étude), est la science qui englobe à la fois la cryptographie (l'art de chiffrer les messages) et la cryptanalyse (l'art de les déchiffrer). Depuis des siècles, la cryptologie a joué un rôle central dans la préservation du secret des informations sensibles, qu'il s'agisse de communications diplomatiques, de secrets militaires ou de transactions commerciales confidentielles [1] [2]

2 .1 Généralités sur la Cryptologie :

La cryptologie est l'ensemble formé de la cryptographie et de la cryptanalyse [1]



La cryptologie fait partie d'un ensemble de théories et de techniques liées `a la transmission de l'information (théorie des ondes électromagnétiques, théorie du signal, théorie des codes correcteur d'erreurs, théorie de l'information, théorie de la complexité,...).

3 La Cryptographie

La cryptographie est l'art de cacher l'information, de la rendre accessible uniquement à un nombre restreint de personnes, Pour se faire on transforme le message pour le rendre illisible mais de manière à pouvoir réobtenir le message d'origine [3] [2]

3.1 Principes de la Cryptographie:

La cryptographie repose sur plusieurs principes fondamentaux : [1]

- **Confidentialité :** Assurer que seules les personnes autorisées peuvent accéder aux informations.
- **Intégrité :** Garantir que les données n'ont pas été altérées ou modifiées de manière non autorisée.
- **Authenticité :** Vérifier l'identité des parties impliquées dans une communication et s'assurer de l'origine des données.
- **Non-répudiation :** Empêcher qu'une partie puisse nier son implication dans une transaction ou une communication.

3.2 Techniques de Chiffrement :

La cryptographie utilise différentes techniques de chiffrement :

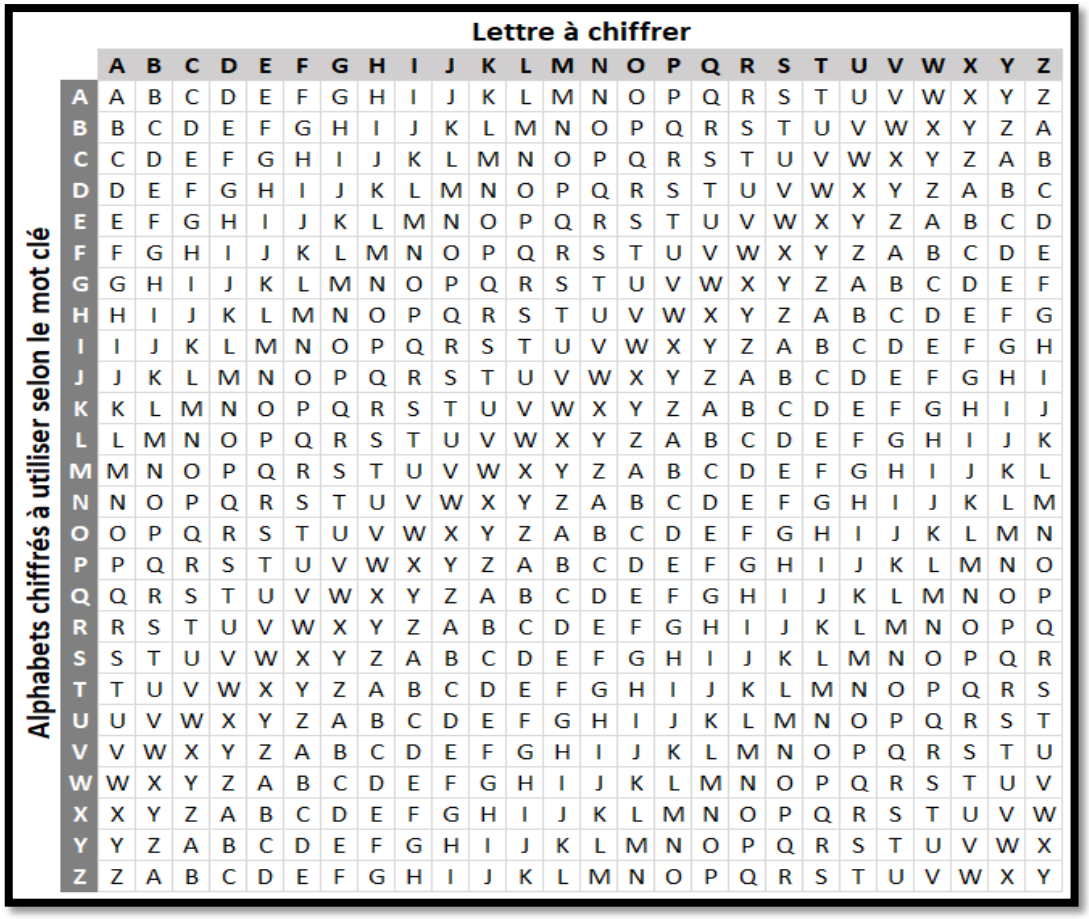
3.2.1 Cryptographie classique :

La cryptographie classique fait référence aux méthodes de chiffrement et de déchiffrement utilisées avant le développement des techniques modernes de cryptographie, telles que la cryptographie symétrique et asymétrique basées sur des algorithmes complexes [4] [5]

3.2.2 méthodes de chiffrement et de déchiffrement

A. Le code de Vigenère :

Le chiffre de vigenère est un système de chiffrement par substitution polyalphabétique utilisant une clé et un tableau à double entrée. Son principe est d'utiliser un chiffre de César, mais où le décalage utilisé change de lettre en lettre . Le chiffre de vigenère est une amélioration décisive du chiffre de César**.** Sa force réside dans l'utilisation non pas d'un, mais de 26 alphabets décalés pour chiffrer un message. On peut résumer ces décalages avec un carré de vigenère comme Figure1.1 **.** Ce chiffre utilise une clef qui définit le décalage pour chaque lettre du message [6]



	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
A	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
B	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	A
C	C	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
D	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
E	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	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	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

Figure1.1 : Carré de vigenère

La lettre de la clef est dans la colonne la plus à gauche, la lettre du message clair est dans la ligne tout en haut. La lettre chiffrée est à l'intersection de la ligne de la lettre clef et de la colonne de la lettre claire .

Exemple

Chiffrons le texte "CHIFFRE DE VIGENERE" avec la clef "BACHELIER" (cette clef est éventuellement répétée plusieurs fois pour être aussi longue que le texte clair). Le tableau ci-dessous illustre le processus de chiffrement, en incluant le texte clair, la clé répétée, les décalages correspondants et le texte chiffré résultant :

Clair	**C**	**H**	**I**	**F**	**F**	**R**	**E**	**D**	**E**	**V**	**I**	**G**	**E**	**N**	**E**	**R**	**E**
Clef	B	A	C	H	E	L	I	E	R	B	A	C	H	E	L	I	E
Décalage	1	0	2	7	4	11	8	4	17	1	0	2	7	4	11	8	4
Chiffré	D	H	K	M	J	C	M	H	V	W	I	I	L	R	P	Z	I

Table 1.1 : Chiffrement vigenère avec clé et décalages

B. Substitution polyalphabétique :

La substitution polyalphabétique est une technique de chiffrement par substitution où chaque caractère du texte en clair n'est pas toujours remplacé par le même caractère du texte chiffré, contrairement à la substitution monoalphabétique [7]

Exemple

Mot-clé : CIPHER (correspondant numériquement à (2, 8, 15, 7, 4, 17))

Texte en clair : cryptosystem (correspondant numériquement à (2, 17, 24……))

comme indiqué dans le tableau 1.2 ci-dessous :

Texte en clair	**c**	**r**	**y**	**p**	**t**	**o**	**s**	**y**	**s**	**t**	**e**	**m**
Valeur (clair)	2	17	24	15	19	14	18	24	18	19	4	12
Mot-clé (val.)	2	8	15	7	4	17	2	8	15	7	4	17
Ajout (mod 26)	4	25	13	22	23	5	20	6	7	0	8	3

Table 1.2 : Chiffrement Polyalphabétique avec Exemple de Chiffrement

Dans la méthode de substitution polyalphabétique, chaque caractère du texte clair est remplacé par un caractère du texte chiffré en fonction d'une clé de chiffrement, souvent appelée mot-clé. Chaque caractère du mot-clé correspond à un décalage dans l'alphabet pour le chiffrement. Par exemple :

> **4 -> E :** Si le caractère du texte clair correspondant à la quatrième position est 'E' dans le texte chiffré, cela signifie qu'il y a un décalage spécifique appliqué par la clé à ce caractère.

Alors : Le texte chiffré est donc : **EZNWXFUGHAID**

Étapes de Déchiffrement :

Texte chiffré	***E***	***Z***	***N***	***W***	***X***	***F***	***U***	***G***	***H***	***A***	***I***	***D***
Valeur (chiff.)	4	25	13	22	23	5	20	6	7	0	8	3
Mot-clé (val.)	2	8	15	7	4	17	2	8	15	7	4	17
Soustraction	2	17	24	15	19	14	18	24	18	19	4	12

Table 1.3 : Chiffrement Polyalphabétique avec Exemple de déchiffrement

Dans la méthode de substitution polyalphabétique, chaque caractère du texte chiffré est déchiffré en utilisant une clé de chiffrement spécifique, souvent appelée mot-clé. Ce mot-clé détermine

une série de décalages dans l'alphabet pour chaque caractère du texte clair. Voici comment cela fonctionne pour le texte chiffré donné :

Texte chiffré et valeurs numériques :

- Le texte chiffré est représenté par "EZNYXFUGHAID", avec des valeurs numériques associées à chaque caractère : (4, 25, 13, 22, 23, 5, 20, 6, 7, 0, 8, 3).

Mot-clé :

- Le mot-clé utilisé pour le chiffrement est "CIPHER", correspondant numériquement à (2, 8, 15, 7, 4, 17).

Déchiffrement :

- Chaque caractère du texte chiffré est déchiffré en soustrayant le décalage correspondant déterminé par le mot-clé.
- Par exemple, pour le premier caractère "E" avec la valeur numérique 4 :
  - Le décalage est déterminé par la première lettre du mot-clé "C", qui a une valeur de 2.
  - Ainsi, 4 - 2 = 2. La deuxième lettre de l'alphabet, correspondant à la valeur 2, est "c".
- Ce processus est répété pour chaque caractère du texte chiffré en utilisant les décalages successifs définis par le mot-clé.

Résultat du déchiffrement :

- En appliquant ce processus à tous les caractères du texte chiffré "EZNYXFUGHAID", on obtient le texte clair déchiffré "cryptosystem".

3.2.3 Chiffrement Symétrique :

Le chiffrement symétrique est une technique de cryptographie qui utilise la même clé pour chiffrer et déchiffrer les données

Comme illustré à la figure 1.2, en cryptage symétrique, les clés de cryptage et de décryptage sont identiques.

Exemples d'Algorithmes de Chiffrement Symétrique : [4] [8]

- **AES (Advanced Encryption Standard) :** Un algorithme de chiffrement symétrique largement utilisé pour sécuriser les données sensibles dans de nombreux protocoles et applications.
- **DES (Data Encryption Standard) :** Un ancien algorithme de chiffrement symétrique qui utilise une clé de 56 bits.
- **3DES (Triple Data Encryption Standard) :** Une version améliorée de DES qui applique le chiffrement DES trois fois avec différentes clés pour renforcer la sécurité.

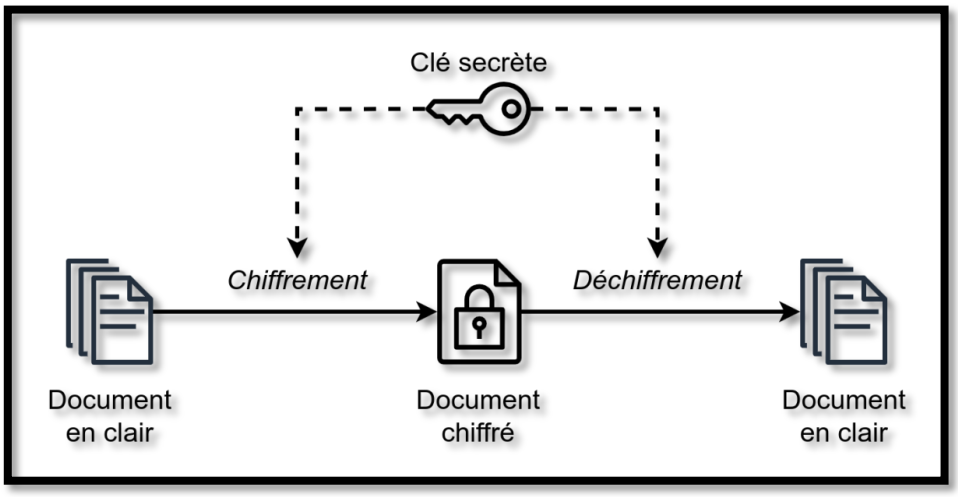


Figure1.2 : Schéma de cryptage symétrique

3.2.4 Chiffrement Asymétrique

Utilisation de paires de clés publiques pour chiffrer et privées pour déchiffrer les données , comme illustré à la figure 1.3, les clés de cryptage et de décryptage sont différentes : une clé publique pour le cryptage et une clé privée pour le décryptage.

Exemples d'Algorithmes de Chiffrement Symétrique : [4] [8]

- **RSA (Rivest-Shamir-Adleman) :** L'algorithme RSA est largement utilisé pour le chiffrement asymétrique et la création de signatures numériques.
- **DSA (Digital Signature Algorithm) :** Le DSA est spécifiquement conçu pour la création de signatures numériques basées sur des clés asymétriques.
- **ElGamal :** L'algorithme ElGamal est utilisé pour le chiffrement et les échanges de clés asymétriques

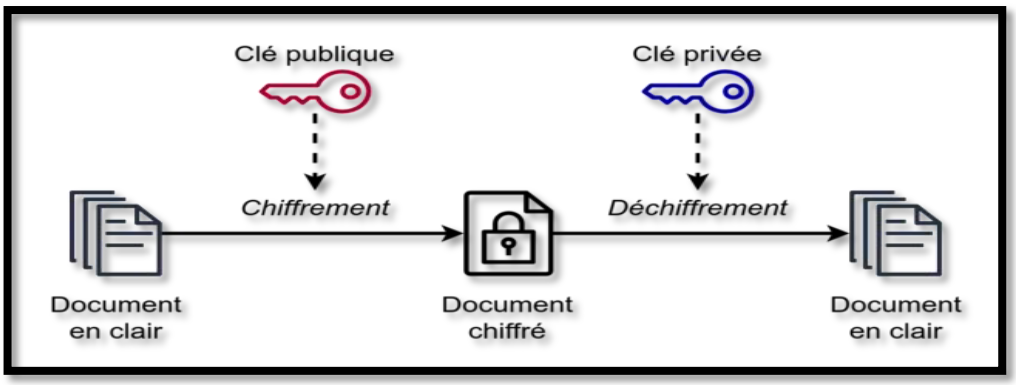


Figure 1.3 : Schéma de cryptage asymétrique

4 La Cryptanalyse :

cryptanalyse est l'art et la science de décrypter des messages chiffrés sans avoir accès à la clé de chiffrement. C'est une discipline essentielle dans le domaine de la sécurité de l'information, car elle permet de tester et de renforcer la sécurité des systèmes de chiffrement [5] [4]

4.1 Objectifs de la Cryptanalyse : [5] [9]

- **Décryptage des Messages Chiffrés :**

La cryptanalyse vise à retrouver le texte clair original à partir d'un texte chiffré sans connaissance de la clé de chiffrement utilisée.

- **Identification de Faiblesses :**

En analysant les méthodes de chiffrement et les textes chiffrés, la cryptanalyse cherche à identifier les faiblesses et les vulnérabilités des systèmes cryptographiques.

- **Optimisation des Techniques :**

En développant des techniques de cryptanalyse avancées, l'objectif est d'améliorer la capacité à casser les chiffrements complexes de manière efficace et efficiente.

4.2 Techniques de cryptanalyse de chiffrements

4.2.1 Analyse Fréquentielle :

L'analyse fréquentielle en cryptanalyse est une méthode qui repose sur les fréquences d'apparition des lettres dans un texte chiffré pour identifier des correspondances avec les lettres les plus fréquentes dans la langue cible comme illustré à la figure 1.5 et la figure1.6, l'analyse des fréquences en français et english permet d'identifier les lettres les plus couramment utilisées. Voici comment elle fonctionne [9] [10] [3] [11] :

- **Collecte de données :**

L'attaquant collecte un échantillon de texte chiffré à analyser, de préférence assez grand pour des résultats précis.

- **Calcul des fréquences :**

Les fréquences d'apparition de chaque lettre dans le texte chiffré sont calculées, représentées sous forme de pourcentage ou de nombre d'occurrences.

- **Comparaison avec les fréquences attendues :**

En comparant ces fréquences avec celles attendues pour la langue utilisée (par exemple, anglais), des correspondances probables sont identifiées.

➢ Identification des substitutions :

Les lettres les plus fréquentes dans le texte chiffré sont associées aux lettres les plus fréquentes dans le texte clair, aidant à deviner des substitutions et à déduire des parties de la clé de chiffrement.

➢ Limitations :

L'analyse fréquentielle est moins efficace contre les chiffrements qui modifient radicalement les fréquences, utilisent des méthodes de confusion complexes ou des clés de chiffrement aléatoires.

➢ Attaque par Force Brute :

Cette approche consiste à essayer toutes les clés possibles jusqu'à trouver la bonne clé qui déchiffre le texte. C'est une méthode exhaustive qui nécessite beaucoup de puissance de calcul mais qui peut être efficace pour certains types de chiffrements.

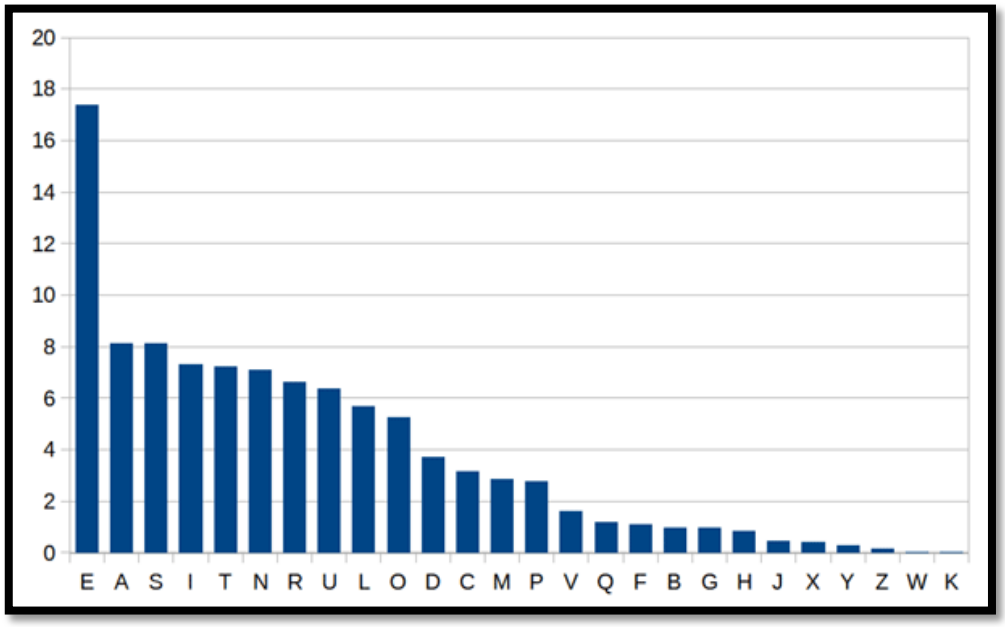


Figure 1.4: Analyse des fréquences en français

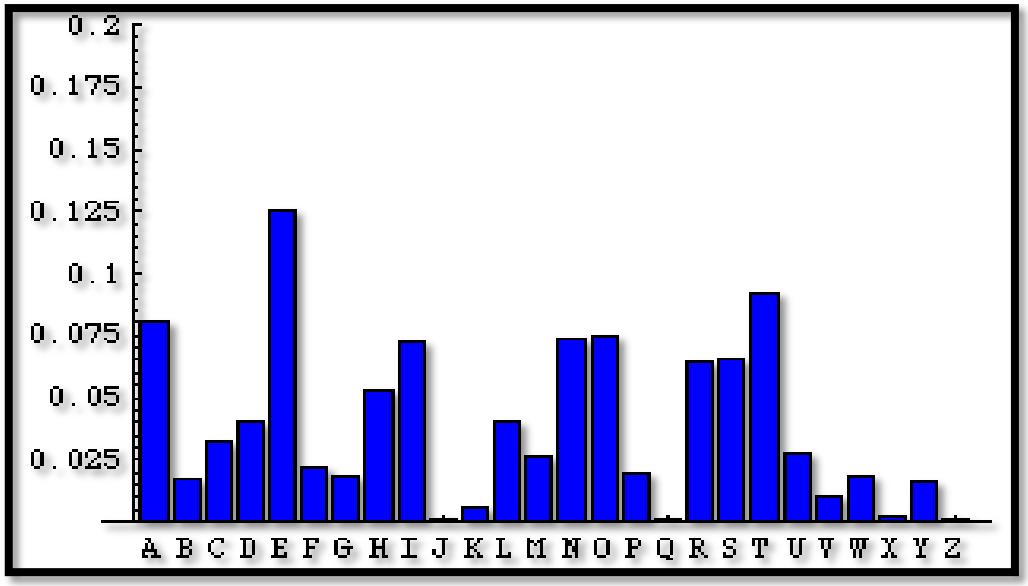


Figure 1.5 : Analyse des fréquences en anglish

4.2.1.1 Analyse Fréquentielle : Utilisation des Bigrammes

L'analyse fréquentielle en cryptanalyse repose principalement sur les fréquences d'apparition des lettres dans un texte chiffré pour identifier des correspondances avec les lettres les plus fréquentes dans la langue cible, par exemple l'anglais. En plus des fréquences simples, l'analyse peut être affinée en utilisant des bigrammes, qui sont des paires de lettres consécutives consécutives comme figure 1.7 . [12] [13]

Fonctionnement :

- **Collecte de données :**

L'attaquant collecte un échantillon de texte chiffré à analyser, de préférence assez grand pour des résultats précis.

- **Calcul des fréquences :**

Les fréquences d'apparition de chaque lettre et de chaque paire de lettres (bigrammes) dans le texte chiffré sont calculées, souvent représentées sous forme de pourcentage ou de nombre d'occurrences.

- **Comparaison avec les fréquences attendues :**

En comparant ces fréquences avec celles attendues pour la langue utilisée, des correspondances probables sont identifiées, non seulement pour les lettres individuelles mais aussi pour les bigrammes.

➢ Identification des substitutions :

Les bigrammes les plus fréquents dans le texte chiffré sont associés aux bigrammes les plus fréquents dans le texte clair, ce qui aide à deviner des substitutions et à déduire des parties de la clé de chiffrement.

➢ Limitations :

Bien que puissante, l'analyse fréquentielle peut être contournée par des chiffrements qui altèrent significativement les fréquences attendues, utilisent des méthodes de confusion complexes ou des clés de chiffrement aléatoires.

➢ Approche par Force Brute :

En parallèle de l'analyse fréquentielle, l'attaque par force brute consiste à essayer toutes les clés possibles jusqu'à trouver celle qui déchiffre correctement le texte. Cette méthode exhaustive demande beaucoup de puissance de calcul mais peut s'avérer efficace pour certains types de chiffrements. [12] [13]

th 3.56%	of 1.17%	io 0.83%
he 3.07%	ed 1.17%	le 0.83%
in 2.43%	is 1.13%	ve 0.83%
er 2.05%	it 1.12%	co 0.79%
an 1.99%	al 1.09%	me 0.79%
re 1.85%	ar 1.07%	de 0.76%
on 1.76%	st 1.05%	hi 0.76%
at 1.49%	to 1.05%	ri 0.73%
en 1.45%	nt 1.04%	ro 0.73%
nd 1.35%	ng 0.95%	ic 0.70%
ti 1.34%	se 0.93%	ne 0.69%
es 1.34%	ha 0.93%	ea 0.69%
or 1.28%	as 0.87%	ra 0.69%
te 1.20%	ou 0.87%	ce 0.65%

Figure 1.6 : La fréquence des bigrammes de lettres les plus courants

4.2.2 Attaque par Dictionnaire :

L'attaque par dictionnaire est une méthode de cryptanalyse qui consiste à essayer une liste prédéfinie de mots (le "dictionnaire") pour casser des mots de passe ou des clés de chiffrement. Voici les étapes clés de cette attaque : [4] [11]

- **Préparation du dictionnaire :**

  L'attaquant crée ou utilise une liste de mots courants, de mots de passe faibles ou de clés potentielles pour tester la sécurité d'un système.

- **Essais avec le dictionnaire :**

  Chaque mot du dictionnaire est utilisé comme clé potentielle pour déchiffrer un texte chiffré ou pour accéder à un système protégé.

- **Comparaison des résultats :**

  L'attaquant vérifie si un mot du dictionnaire permet de réussir l'attaque, c'est-à-dire de déchiffrer avec succès le texte ou de déverrouiller le système.

- **Efficacité et limites :**

  L'attaque par dictionnaire est efficace contre les mots de passe ou les clés faibles, mais elle est moins efficace contre des mots de passe longs, aléatoires et complexes, ou contre des mesures de sécurité.

- **Utilisation pratique :**

  Cette attaque est largement utilisée pour évaluer la robustesse des mots de passe et des clés de chiffrement. Pour se protéger, il est recommandé d'utiliser des mots de passe forts qui ne sont pas présents dans les dictionnaires courants.

4.2.3 Attaque par Texte Clair Chiffré (Known-plaintext Attack) :

L'attaque par texte clair chiffré (connue également sous le nom d'attaque par texte clair choisi) est une technique de cryptanalyse qui exploite la connaissance d'une partie du texte clair associé à son texte chiffré correspondant pour récupérer la clé de chiffrement. Voici les étapes clés de cette attaque: [14] [4] [15]

- **Collecte de paires texte clair - texte chiffré :**

L'attaquant collecte des paires de données où il connaît à la fois le texte clair original et le texte chiffré correspondant. Ces paires sont souvent obtenues en envoyant des requêtes au système de chiffrement avec des textes clairs spécifiques et en enregistrant les résultats chiffrés.

- **Analyse des correspondances :**

L'attaquant analyse les correspondances entre les parties du texte clair et les parties correspondantes du texte chiffré. Cette analyse peut révéler des modèles, des répétitions ou des relations entre les données chiffrées et déchiffrées.

- **Déduction de la clé de chiffrement :**

En utilisant les informations obtenues à partir des paires texte clair - texte chiffré, l'attaquant tente de déduire des parties de la clé de chiffrement. Par exemple, il peut identifier des morceaux de clé qui sont utilisés pour chiffrer des parties spécifiques des données.

- **Utilisation des déductions pour attaquer d'autres données :**

Une fois que l'attaquant a obtenu des informations sur la clé de chiffrement, il peut l'utiliser pour attaquer d'autres données chiffrées par le même système de chiffrement, même s'il n'a pas accès au texte clair correspondant.

- **Efficacité et prévention :**

Cette attaque est très efficace lorsque des paires texte clair - texte chiffré sont disponibles et que le système de chiffrement n'est pas conçu pour résister à cette attaque. Les mesures de sécurité telles que le chiffrement avec des clés uniques pour chaque message (chiffrement par blocs avec modes d'opération sécurisés) peuvent limiter l'impact de cette attaque.

4.2.4 Attaque par Analyse des Temps d'Exécution (Timing Attack) :

L'attaque par analyse des temps d'exécution est une méthode de cryptanalyse qui exploite les variations de temps de traitement d'un système cryptographique pour obtenir des informations sur la clé de chiffrement. Voici les étapes clés de cette attaque [14] [15]:

- **Principe de base :**

L'attaque par analyse des temps d'exécution repose sur le fait que le temps de traitement d'un algorithme de chiffrement peut varier en fonction des données en entrée et de la clé utilisée.

- **Collecte de données :**

L'attaquant enregistre le temps nécessaire pour chiffrer ou déchiffrer différentes données à l'aide du système ciblé. Il effectue ces mesures à plusieurs reprises pour chaque donnée et pour différentes clés potentielles.

- **Analyse des variations de temps :**

En analysant les variations de temps de traitement pour différentes clés et données, l'attaquant cherche des corrélations qui pourraient indiquer la bonne clé de chiffrement.

- **Déduction de la clé :**

L'attaquant utilise les informations obtenues pour déduire des parties de la clé de chiffrement. Par exemple, il peut identifier des patterns ou des schémas dans les temps d'exécution qui correspondent à des caractéristiques de la clé.

- **Utilisation des déductions pour attaquer le système :**

Une fois qu'une partie de la clé de chiffrement est déduite avec succès, l'attaquant peut l'utiliser pour attaquer d'autres données chiffrées par le même système.

- **Efficacité et prévention :**

L'attaque par analyse des temps d'exécution est particulièrement efficace contre les systèmes dont le temps de traitement dépend directement de la clé de chiffrement. Pour se protéger, les concepteurs de systèmes cryptographiques peuvent utiliser des techniques pour uniformiser le temps de traitement indépendamment de la clé utilisée.

5 Conclusion :

Dans ce chapitre, nous avons examiné en détail le domaine de la cryptologie, une discipline cruciale pour garantir la sécurité des informations dans un monde dominé par les communications numériques et les données sensibles

Chapiter 02 :
Méthodes d'optimisation en cryptanalyse

Chapitre 02 : Méthodes d'Optimisation en Cryptanalyse

1 Introduction :

Dans le domaine de la cryptanalyse, où l'objectif est de casser les systèmes de chiffrement pour accéder aux données protégées, les méthodes d'optimisation jouent un rôle crucial. Ces techniques avancées permettent d'explorer efficacement l'espace des solutions, de trouver des configurations de paramètres optimales et de résoudre des problèmes complexes de manière plus rapide et plus efficace.

2 Généralités :

Les méthodes d'optimisation en cryptanalyse sont des techniques informatiques utilisées pour résoudre des problèmes de déchiffrement en trouvant les meilleures configurations de clés ou de paramètres. Elles s'inspirent souvent des processus naturels ou des comportements collectifs pour explorer efficacement l'espace des solutions et trouver des résultats optimaux. [18]

2.1 Algorithmes Génétiques (GA) :

2.1.1 Définition :

Les algorithmes génétiques (GA), faisant partie des algorithmes évolutionnaires, sont des méthodes d'optimisation stochastiques qui simulent l'évolution naturelle à travers la compétition, où seuls les individus les plus aptes survivent et transmettent leur patrimoine génétique aux générations suivantes. Ils ont été formellement introduits en littérature par Holland en 1975, suivi de travaux de vulgarisation par Goldberg (1989), Jean-Marc & Thomas (1993), et tlo@mit.edu (2003). De nos jours, les GA sont largement utilisés dans divers domaines tels que l'industrie, l'économie, la planification et les outils de prise de décision pour fournir des solutions proches de l'optimum dans un temps raisonnable

2.1.2 Description et algorithme :

Les algorithmes génétiques (GA) sont des méthodes d'optimisation inspirées par le processus d'évolution naturelle et la sélection naturelle des espèces. Voici une description et un aperçu de l'algorithme génétique typique : [7] [19]

- Les GA traitent les solutions possibles comme des individus dans une population.
- Chaque individu est représenté par un ensemble de caractéristiques (gènes) qui constituent une solution potentielle au problème.
- Les individus sont évalués en fonction de leur "fitness" ou aptitude par rapport à une fonction objectif donnée.

- Les individus les mieux adaptés ont plus de chances de survivre et de se reproduire, tandis que les moins adaptés sont éliminés ou ont moins de chances de se reproduire.

2.1.3 Vocabulaire :

Le vocabulaire des Algorithmes Génétiques (AG) est étroitement lié aux principes de l'évolution et de la génétique, et comprend des termes spécifiques qui décrivent les éléments et les processus clés de ces algorithmes. Voici les principaux termes utilisés dans le contexte des Algorithmes Génétiques : [20] [19]

- **Individu :** Un élément de l'espace de recherche représentant une solution potentielle au problème.
- **Population :** Un ensemble fini d'individus qui constitue la génération actuelle dans l'algorithme.
- **Génération :** Une itération dans laquelle une nouvelle population est créée à partir de la population actuelle en utilisant les opérateurs génétiques.
- **Évolution :** Le processus itératif de recherche d'un ou plusieurs individus optimaux à travers les générations successives.
- **Performance :** La mesure de la qualité d'un individu par rapport à l'objectif d'optimisation, utilisée pour comparer les individus et déterminer les plus aptes.
- **Évaluation d'un individu :** Le calcul de sa performance en fonction de la fonction objectif.
- **Croisement (Crossover) :** L'opérateur de reproduction qui combine les gènes de deux individus pour créer de nouveaux descendants.
- **Mutation :** L'opérateur de modification aléatoire d'un ou plusieurs gènes d'un individu pour introduire de la diversité dans la population.
- **Sélection :** Le processus de choix des individus les mieux adaptés pour la reproduction, basé sur leur performance.
- **Remplacement :** Le processus de formation d'une nouvelle génération en remplaçant certains individus de la génération précédente par des descendants

Ce procédé peut être résumé ou schématisé comme suit :

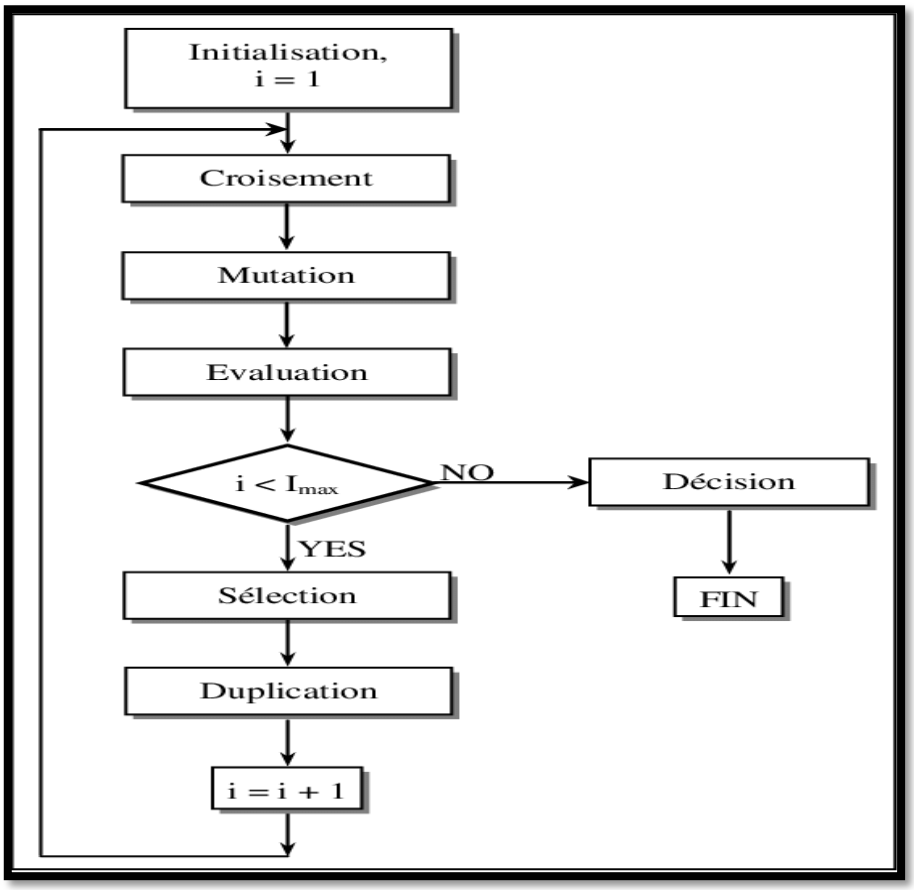


Figure 2.1: Organigramme de l'algorithme génétique

2.1.4 Utilisation des Algorithmes Génétiques :

2.1.4.1 Optimisation des clés:

L'application des Algorithmes Génétiques (AG) à la recherche de clés en cryptanalyse implique l'utilisation de principes évolutifs pour explorer efficacement un grand espace de solutions potentielles et trouver la ou les clés qui déchiffrent correctement les textes chiffrés. Voici comment les AG sont appliqués à la recherche de clés : [21] [22]

- **Représentation des Clés :**

  Dans une recherche de clés basée sur les AG, les clés cryptographiques sont représentées comme des individus ou des chromosomes dans une population. Chaque chromosome représente une solution potentielle de clé, généralement codée sous forme de chaîne de bits.

- **Initialisation de la Population :**

  Une population de clés potentielles est initialisée de manière aléatoire ou en utilisant des heuristiques spécifiques. Cette population initiale sert de point de départ pour l'algorithme AG.

- **Évaluation de la Fitness :**

  La fitness de chaque candidat clé est évaluée en fonction d'une fonction de fitness ou objectif. Cette fonction mesure à quel point chaque clé déchiffre correctement les textes chiffrés ou répond à certains critères (par exemple, similitude avec des textes en clair connus).

- **Sélection :**

  Les individus ayant des scores de fitness plus élevés, c'est-à-dire les clés qui déchiffrent mieux les textes chiffrés, sont sélectionnés avec une probabilité plus élevée pour la reproduction. Cela imite le processus de sélection naturelle, où les individus les plus adaptés ont plus de chances de transmettre leurs informations génétiques.

- **Reproduction (Croisement et Mutation) :**

  Les individus sélectionnés subissent une reproduction à travers des opérations de croisement et de mutation. Le croisement implique de combiner des informations génétiques de deux clés parentes pour produire des clés descendantes. La mutation introduit des changements aléatoires dans les clés descendantes pour explorer de nouvelles zones de l'espace de solution.

- **Création d'une Nouvelle Génération :**

  Les clés descendantes, ainsi que certaines clés parentes, forment la nouvelle génération de clés potentielles. Ce processus se poursuit de manière itérative sur plusieurs générations.

- **Convergence et Identification des Clés :**

  L'algorithme AG se poursuit jusqu'à ce qu'un critère d'arrêt soit atteint, comme un nombre maximum de générations ou l'atteinte d'un niveau de fitness satisfaisant. À la convergence, la ou les clés les mieux adaptées sont identifiées comme des solutions potentielles au problème de recherche de clés.

- **Validation et Test :**

  Les clés identifiées sont validées en déchiffrant des textes chiffrés et en vérifiant la précision du déchiffrement. Les tests peuvent impliquer l'utilisation des clés pour déchiffrer d'autres textes chiffrés pour garantir leur efficacité.

- **Optimisation et Raffinement :**

  Le processus de recherche de clés basé sur les AG peut être optimisé et affiné en ajustant des paramètres tels que la taille de la population, les taux de croisement et de mutation, et les mécanismes de sélection pour améliorer l'efficacité de la recherche et la qualité des solutions.

2.1.5 Chiffrement affine:

Les algorithmes génétiques peuvent être utilisés pour briser le chiffrement affine, une méthode de cryptographie classique. Le chiffrement affine est défini par la formule :

$$E(x) = (a \cdot x + b) \bmod m$$

où x est la lettre à chiffrer, a et b sont les clés du chiffrement, et m est la taille de l'alphabet (généralement 26 pour l'alphabet anglais). Pour que le chiffrement soit réversible, a doit être copremier avec m (c'est-à-dire que le PGCD de a et m doit être 1).

Le déchiffrement est donné par :

$$D(y) = a^{-1} \cdot (y - b) \bmod m$$

où a^{-1} est l'inverse multiplicatif de a modulo m.

Utilisation des Algorithmes Génétiques pour Déchiffrer un Texte Affine [23]

- **Représentation des Individus :**

  Chaque individu dans la population représente une paire de clés (a,b). Par exemple, un individu peut être représenté sous la forme d'un vecteur [a,b].

- **Population Initiale :** Une population initiale d'individus est générée aléatoirement. La taille de la population est un paramètre clé de l'algorithme.

- **Fonction de Fitness :**

  La fonction de fitness évalue la qualité des clés de déchiffrement proposées. Une approche courante est de mesurer la similarité du texte déchiffré avec une langue cible en utilisant des fréquences de lettres ou des digrammes/trigrammes.

- **Sélection :**

  Les individus les plus aptes (ceux avec la meilleure valeur de fitness) sont sélectionnés pour la reproduction. La sélection peut se faire par roulette, tournoi, ou d'autres méthodes de sélection.

- **Croisement (Crossover) :**

  Les individus sélectionnés sont croisés pour produire de nouveaux individus. Par exemple, avec un croisement en un point, une partie des clés du parent 1 est combinée avec une partie des clés du parent 2.

- **Mutation :**

  Pour introduire de la diversité génétique, certaines clés peuvent subir des mutations aléatoires. Par exemple, la clé a peut être remplacée par une autre valeur coprime avec m.

- **Nouvelle Génération :**

  La nouvelle génération est constituée des individus résultant du croisement et de la mutation. Ce processus est répété sur plusieurs générations jusqu'à ce qu'un critère d'arrêt soit atteint (nombre de générations, seuil de fitness, etc.).

2.1.6 chiffrement par substitution :

Le chiffrement par substitution est une technique très simple qui a été utilisée pendant des siècles. L'idée de base est de remplacer chaque lettre ou symbole par une autre lettre ou symbole d'une manière que seul l'expéditeur et le destinataire connaissent. Le destinataire peut décrypter le message et le lire, tandis que toute personne intermédiaire verra un ensemble de lettres apparemment aléatoire. Dans ce cas, il y a trois étapes principales dans ce processus.

- Supprimer tous les caractères non alphabétiques du message (espaces, ponctuation, etc.).
- Créer un mapping de caractères pour chaque lettre.
- Utiliser le mapping de caractères pour chiffrer le message.

La première étape rend plus difficile le décryptage du message. Cela rend très difficile la détection de mots et de phrases dans le message. La deuxième étape est la clé du chiffrement par substitution. C'est cette partie qui doit être le "secret" que seuls l'expéditeur et le destinataire connaissent. La clé nous indique quelles lettres dans le message chiffré correspondent à quelles lettres dans le message original. Par exemple, "a" dans le message original peut être un "q" dans le message chiffré. La troisième étape consiste simplement à utiliser cette clé pour chiffrer le message original en un message chiffré.L'utilisation des algorithmes génétiques pour attaquer un chiffrement par substitution simple implique de trouver la correspondance entre les caractères du texte en clair et les caractères du texte chiffré. Voici comment vous pourriez aborder cela en utilisant des algorithmes génétiques : [23]

- **Initialisation :**

  Commencez par créer une population initiale de correspondances potentielles entre les caractères du texte en clair et les caractères du texte chiffré. Cette population peut être une série de permutations aléatoires de l'alphabet ou de l'ensemble de caractères utilisé dans le chiffrement.

- **Fonction de Fitness :** Définissez une fonction de fitness qui évalue la qualité de chaque correspondance en fonction de la cohérence avec la langue, des analyses fréquentielles, ou

d'autres mesures pertinentes. Les correspondances qui produisent des résultats plus "plausibles" sont mieux notées.

- **Sélection :**

  Utilisez des mécanismes de sélection comme la sélection par tournoi ou la sélection par roulette pour choisir les correspondances les plus prometteuses pour la reproduction.

- **Croisement :**

  Appliquez des opérations de croisement aux paires de correspondances sélectionnées pour créer des correspondances "enfants". Par exemple, vous pouvez échanger des parties des correspondances entre les parents pour créer de nouvelles correspondances.

- **Mutation :**

  Introduisez des changements aléatoires dans les correspondances "enfants" pour explorer de nouvelles possibilités. Cela peut inclure le remplacement de caractères, le déplacement de correspondances, ou d'autres transformations similaires.

- **Évaluation :**

  Évaluez la qualité des correspondances "enfants" en utilisant la fonction de fitness.

- **Remplacement :**

  Remplacez les correspondances les moins performantes de la population par les nouvelles correspondances "enfants", tout en maintenant une diversité suffisante.

- **Terminaison :**

  Répétez le processus d'évolution et d'évaluation sur plusieurs générations jusqu'à ce qu'une correspondance satisfaisante soit trouvée ou qu'un critère de terminaison soit atteint (par exemple, nombre maximal d'itérations).

Au fil des générations, l'algorithme génétique évolue une population de correspondances entre les caractères du texte en clair et du texte chiffré, en favorisant celles qui produisent des résultats cohérents et "plausibles" lors du déchiffrement. En itérant à travers les étapes de sélection, de croisement, de mutation et d'évaluation, l'algorithme converge vers une solution efficace pour attaquer le chiffrement par substitution simple

3 Recuit simulé :

3.1 définition :

Le recuit simulé en cryptographie est une technique d'optimisation stochastique utilisée pour résoudre des problèmes cryptographiques, notamment dans le domaine de la cryptanalyse et de l'optimisation des systèmes cryptographiques. Il s'agit d'une approche heuristique qui simule le

processus de recuit dans les métaux pour explorer efficacement des espaces de recherche complexes et trouver des solutions optimales ou proches de l'optimalité pour des problèmes cryptographiques spécifiques

3.2 Description :

Le recuit simulé est une méthode d'optimisation stochastique utilisée en cryptanalyse pour attaquer des systèmes de chiffrement, notamment les chiffrements par transposition. L'objectif est de trouver la clé de chiffrement ou la permutation des caractères qui déchiffre correctement le texte chiffré [18]

3.3 Vocabulaire :

Voici le vocabulaire couramment utilisé dans le contexte du recuit simulé en cryptographie : [24]

- **Recuit Simulé (SA) :**

  Technique d'optimisation stochastique inspirée du processus de recuit dans les métaux, utilisée pour résoudre des problèmes d'optimisation complexes.

- **Température :** Paramètre contrôlant la probabilité d'accepter des solutions de moins bonne qualité lors de la recherche.
- **Taux de Refroidissement :** Taux auquel la température diminue au cours du processus.
- **Solution Initiale :**

  Point de départ de l'algorithme, généralement obtenu de manière aléatoire ou par des heuristiques.

- **Voisinage :**

  Ensemble des solutions proches de la solution actuelle, générées en effectuant des perturbations.

- **Fonction de Fitness :**

  Fonction évaluant la qualité d'une solution, utilisée pour comparer différentes solutions.

- **Acceptation Probabiliste :**

  Mécanisme permettant d'accepter une nouvelle solution même si elle est de moins bonne qualité, basé sur des probabilités.

- **Refroidissement :**

  Processus de diminution de la température au fil du temps, permettant de converger vers une solution optimale.

- **Terminaison :** Condition d'arrêt de l'algorithme, généralement basée sur la température atteignant un seuil prédéfini ou un nombre d'itérations maximum.

- **Optimalité :**

  Caractéristique d'une solution satisfaisant les critères d'optimisation définis pour le problème.

3.4 Utilisation du recuit simulé :

3.4.1 le chiffrement par transposition :

En 1983, Kirkpatrick a proposé un algorithme basé sur l'analogie entre le recuit des solides et la résolution de problèmes d'optimisation combinatoire.

Une attaque sur le chiffrement par transposition utilisant le recuit simulé [25] [27] [26]

Voici les étapes :

1. Définir la température initiale, T(0).
2. Générer une solution initiale - arbitrairement définie comme la transformation identitaire (peut être générée de manière aléatoire ou autrement).
3. Évaluer la fonction de coût pour la solution initiale. Appelons cela C(0).
4. Pour la température T, répéter plusieurs fois (par exemple, 100 × M fois) :
5. Générer une nouvelle solution en modifiant la solution actuelle de quelque manière que ce soit.
6. Évaluer la fonction de coût pour la nouvelle solution proposée.
7. Consulter la fonction Metropolis pour décider si la nouvelle solution proposée sera acceptée ou non.
8. Si elle est acceptée, mettre à jour la solution actuelle et son coût associé.

Si le nombre de transitions acceptées pour la température T dépasse une limite donnée (par exemple, 10 × M), passer à l'étape 5.

9. Si le nombre de transitions acceptées pour la température T était nul, arrêter (retourner la solution actuelle comme la meilleure), sinon réduire la température (par exemple, 0.95 × T(i+1) = T(i)) et revenir à l'étape 4.

3.4.2 Chiffrement de Vigenère:

Le chiffrement de Vigenère utilise une clé composée de lettres. Chaque lettre de la clé correspond à un décalage à appliquer aux lettres du texte clair. La fonction de chiffrement est donnée par :

$$E(x) = (x + k_i) \bmod m$$

où :

- x est la position de la lettre du texte clair dans l'alphabet (0 pour A, 1 pour B, etc.),

- k_i est la position de la lettre ***i-ème*** de la clé dans l'alphabet,
- ***m*** est la taille de l'alphabet (26 pour l'alphabet anglais) [27]

Utilisation des recuit simulé déchiffrer un texte chiffré par Vigenère. [27]

1. **Initialisation :**

- **Solution Initiale** : Commencer avec une clé de chiffrement aléatoire d'une longueur déterminée. Par exemple, si la longueur de la clé est supposée être 3, choisir une clé comme "ABC".
- **Température Initiale** *:* Définir une température initiale élevée, par exemple 1000.
- **Fonction de *Fitness* *:*** Définir une fonction pour évaluer la qualité du texte déchiffré. Cette fonction peut être basée sur la fréquence des lettres ou des digrammes dans la langue de destination.

2. **Perturbation :**

Générer une nouvelle solution (nouvelle clé) en modifiant légèrement la solution actuelle. Par exemple, changer une lettre de la clé actuelle "ABC" en "ABD".

3. **Décryptage :**

Utiliser la clé pour déchiffrer le texte chiffré et obtenir le texte clair.

4. **Évaluation de la Fitness :**

Évaluer la qualité du texte clair obtenu en utilisant la fonction de fitness. Par exemple, comparer les fréquences des lettres du texte déchiffré avec les fréquences attendues dans la langue cible (comme l'anglais).

5. **Acceptation de la Nouvelle Solution :**

- Si la nouvelle solution a une meilleure fitness (le texte déchiffré ressemble plus à la langue cible), l'accepter comme la solution actuelle.
- Si la nouvelle solution est moins bonne, l'accepter avec une probabilité PPP donnée par la fonction de Boltzmann : $\mathbf{P = e^{-\Delta E/T}}$, où $\boldsymbol{\Delta E}$ est la différence de fitness et $\boldsymbol{T}$ est la température actuelle. Cette étape permet d'éviter les minima locaux en acceptant occasionnellement des solutions moins optimales.

6. **Refroidissement :**

Réduire progressivement la température selon un schéma de refroidissement, par exemple **T=T×0.99**

7. **Itération :**

Répéter les étapes de perturbation, décryptage, évaluation de la fitness, acceptation de la nouvelle solution, et refroidissement jusqu'à ce que la température soit très basse ou qu'un nombre maximum d'itérations soit atteint.

3.4.3 chiffrements par substitution homophonique :

Les chiffrements par substitution homophonique sont des techniques de cryptographie où chaque lettre ou symbole du texte clair est remplacé par plusieurs autres lettres ou symboles de manière à introduire davantage de confusion et de complexité dans le chiffrement. L'utilisation du recuit simulé dans ce contexte vise à casser ces chiffrements en trouvant la clé de déchiffrement qui permet de retrouver le texte clair à partir du texte chiffré. [28] [4]

Dans le cas de la cryptanalyse des chiffrements par substitution homophonique, le recuit simulé est utilisé pour rechercher la clé de déchiffrement qui minimise une fonction de coût définie en fonction de la similarité entre le texte déchiffré et une langue naturelle (comme l'anglais, le français, etc.).

Le processus d'utilisation du recuit simulé pour attaquer les chiffrements par substitution homophonique implique généralement les étapes suivantes :

1. **Initialisation :**

Une clé de déchiffrement initiale est générée, parfois de manière aléatoire.

2. **Évaluation :**

Le texte déchiffré est obtenu en appliquant la clé de déchiffrement sur le texte chiffré. Une fonction de coût est alors utilisée pour évaluer la qualité de la solution, généralement basée sur la fréquence des lettres ou des mots dans le texte déchiffré par rapport à une langue naturelle.

3. **Recherche Locale :**

Le recuit simulé explore l'espace des clés de déchiffrement en effectuant des modifications locales pour améliorer la solution.

4. **Acceptation Probabiliste :**

Le recuit simulé utilise une fonction de probabilité pour décider d'accepter ou de rejeter les modifications apportées à la clé de déchiffrement. Cela permet d'éviter les minimums locaux et de rechercher des solutions plus optimales.

5. **Refroidissement :**

Le processus se poursuit avec des températures décroissantes, simulant le refroidissement du système. Cela permet d'explorer progressivement des solutions de meilleure qualité.

6. **Convergence :**

Le processus de recuit simulé converge vers une solution qui minimise la fonction de coût, représentant ainsi la clé de déchiffrement optimale ou une approximation proche de celle-ci.

4 Algorithme de Recherche Tabou :

4.1 Définition :

L'algorithme de Recherche Tabou (TS) est une technique d'optimisation utilisée en cryptanalyse pour résoudre des problèmes d'optimisation combinatoire, notamment dans le domaine de la cryptographie. Son objectif est de trouver une solution optimale ou proche de l'optimalité en explorant efficacement l'espace des solutions tout en évitant de rester bloqué dans des minimums locaux.

4.2 Description et algorithme :

L'algorithme de Recherche Tabou (TS) est une méthode d'optimisation qui peut être appliquée à la cryptanalyse pour résoudre des problèmes de recherche de clés de chiffrement ou d'autres problèmes d'optimisation combinatoire. Voici une description et un schéma algorithmique simplifié de l'algorithme de Recherche Tabou [18] [24]

1. **Initialisation :**

- Définir une solution initiale (clé de chiffrement initiale).
- Initialiser une liste tabou vide.

2. **Évaluation :**

Évaluer la qualité de la solution initiale en fonction d'une fonction de coût (ex: qualité du déchiffrement du texte chiffré avec la clé).

3. **Recherche Tabou :**

- Générer des solutions voisines en modifiant la solution actuelle.
- Éviter les solutions taboues (interdites) en consultant la liste tabou.

4. **Choix de la Meilleure Solution :**

Parmi les solutions voisines, choisir la meilleure solution selon la fonction de coût, en évitant les solutions taboues.

5. **Mise à jour de la Liste Tabou :**

Mettre à jour la liste tabou en ajoutant la solution précédente et en retirant les solutions les plus anciennes si nécessaire.

6. **Critère d'Arrêt :**

Répéter les étapes 3 à 5 jusqu'à atteindre un critère d'arrêt (nombre d'itérations, convergence vers une solution optimale, etc.).

4.3 Utilisation du Recherche Tabou :

4.3.1 Attaquer et analyser des algorithmes de chiffrement :

La Recherche Tabou, une technique d'optimisation, peut être utilisée pour attaquer et analyser des algorithmes de chiffrement tels que VMPC (Variably Modified Permutation Composition). Voici comment elle peut être appliquée : . [29] [24]

- **Identification des Objectifs :**

Avant de commencer l'attaque, les objectifs de la cryptanalyse doivent être clairement définis. Cela peut inclure la récupération de la clé secrète utilisée pour le chiffrement VMPC, l'analyse de la résistance aux attaques par faute, l'évaluation de la complexité de l'algorithme, etc

- **Modélisation du Problème :**

La cryptanalyse Tabou nécessite la modélisation du problème sous forme d'un espace de recherche avec des solutions potentielles. Dans le cas du chiffrement VMPC, cela impliquerait de représenter l'algorithme, ses paramètres, et la clé secrète comme des éléments de cet espace de recherche.

- **Initialisation de la Solution Initiale :**

Un point de départ initial est défini dans l'espace de recherche, souvent de manière aléatoire ou par des techniques heuristiques. Cette solution initiale peut représenter une supposition sur la clé secrète ou d'autres paramètres de l'algorithme.

- **Exploration de l'Espace de Recherche :**

L'algorithme de cryptanalyse Tabou explore l'espace de recherche en générant et évaluant des solutions potentielles. Il utilise des mécanismes de mouvement pour se déplacer à travers l'espace de recherche, en évitant les pièges locaux grâce à des listes Tabou qui mémorisent les mouvements interdits.

- **Évaluation des Solutions :**

Chaque solution générée est évaluée en fonction de critères prédéfinis, tels que la corrélation avec des modèles de texte clair, la qualité de la séquence de chiffrement générée, la capacité à retrouver des structures dans le chiffrement, etc.

- **Stratégies de Mouvement :**

L'algorithme de cryptanalyse Tabou utilise des stratégies de mouvement pour explorer efficacement l'espace de recherche. Cela peut inclure des opérations de perturbation, de croisement avec d'autres solutions, de mutation de paramètres, etc.

- **Critères d'Arrêt :**

L'attaque Tabou se poursuit jusqu'à ce qu'un critère d'arrêt soit atteint, tel qu'une solution satisfaisante trouvée, un nombre maximal d'itérations atteint, ou une stagnation de l'exploration de l'espace de recherche .

4.3.2 Chiffrement par Substitution Monoalphabétique :

le chiffrement par substitution monoalphabétique. Ce type de chiffrement remplace chaque lettre du texte clair par une lettre différente de l'alphabet selon une clé de substitution fixe.

Par exemple, une clé de substitution pourrait être une permutation de l'alphabet comme suit :

$$\mathbf{Cle' = \{A \rightarrow Q, B \rightarrow W, C \rightarrow E, \ldots, Z \rightarrow P\}}$$

Utilisation la Recherche Tabou pour Déchiffrement par Substitution Monoalphabétique [30] [31]

- **Initialisation** :

- Choisir une clé de substitution initiale aléatoire.
- Définir une fonction de fitness pour évaluer la qualité du texte déchiffré.
- Initialiser la liste tabou pour mémoriser les solutions récemment visitées.

- **Perturbation :**

- Générer une nouvelle clé voisine en échangeant deux lettres de la clé actuelle.

- **Évaluation** :

- Utiliser la nouvelle clé pour déchiffrer le texte chiffré.
- Calculer la fitness du texte déchiffré.

- **Mise à jour de la solution** :

- Si la nouvelle solution est meilleure que la solution actuelle, l'accepter.
- Si elle est moins bonne, l'accepter seulement si elle n'est pas dans la liste tabou.
- Mettre à jour la liste tabou avec la solution actuelle.

- **Itération** :

- Répéter les étapes de perturbation, évaluation et mise à jour de la solution jusqu'à atteindre un critère d'arrêt (nombre maximum d'itérations ou convergence de la solution).

5 Conclusion :

Ce chapitre explore là les méthodes d'optimisation dans le domaine de la cryptanalyse en permettant de résoudre des problèmes complexes de déchiffrement et de recherche de clés.

À travers l'étude des algorithmes génétiques, de la recherche tabou, , et du recuit simulé, nous avons pu observer leur efficacité dans la cassure de systèmes de chiffrement et dans la recherche de solutions optimales

Chapiter 03 :

Implémentation et expérimentation

Chapitre 03 : Implémentation et expérimentation

1 Introduction :

La cryptanalyse des textes chiffrés par substitution constitue une branche essentielle de la sécurité informatique, visant à retrouver le contenu original d'un message sans accès direct à la clé de chiffrement. Cette discipline repose sur l'exploitation intelligente des structures linguistiques propres à chaque langue naturelle, où l'analyse des fréquences de caractères et de séquences de caractères joue un rôle crucial. L'utilisation des fréquences de bigrammes et de trigrammes, qui capturent les relations prédictibles entre paires et triplets de caractères, offre une méthode robuste pour décrypter efficacement les messages chiffrés.

2 Contexte de travail :

La cryptanalyse de textes chiffrés est une discipline fondamentale en sécurité informatique, visant à retrouver le texte original à partir de sa version chiffrée. Cette tâche est particulièrement complexe sans la clé de chiffrement correspondante. L'analyse fréquentielle des langues naturelles offre une méthode puissante pour décrypter les messages chiffrés en exploitant les modèles statistiques des caractères et des séquences de caractères qui se produisent avec une fréquence prévisible dans la langue cible.

Les fréquences de bigrammes et trigrammes sont des outils essentiels dans cette approche, car ils capturent les relations entre paires et triplets de caractères dans un texte. Par exemple, en anglish, certains bigrammes comme "he" et "th" ainsi que des trigrammes comme "the" sont beaucoup plus fréquents que d'autres combinaisons de lettres. Cette connaissance statistique peut être utilisée pour estimer la probabilité qu'une certaine clé de déchiffrement soit correcte.

3 Analyse Fréquentielle : Bigrammes et Trigrammes :

Les bigrammes et trigrammes se réfèrent respectivement à des séquences de deux et trois caractères consécutifs dans un texte. En cryptanalyse, leur utilisation repose sur l'hypothèse que les fréquences d'apparition de ces séquences sont relativement constantes dans une langue donnée. Voici comment nous les avons intégrés dans notre approche :

3.1 Chargement des Fréquences de N-grammes :

Nous avons extrait les fréquences de bigrammes et trigrammes à partir de fichiers CSV (bi-ngramFrequency.csv et tri-ngramFrequency.csv). Ces fichiers contiennent des données précalculées où chaque ligne représente un n-gramme et son nombre d'occurrences dans un corpus de texte.

4 Algorithme Génétique :

L'algorithme génétique est une méthode heuristique inspirée par la théorie de l'évolution naturelle. Il est utilisé pour explorer efficacement de grands espaces de solutions potentielles en cryptanalyse. Voici comment nous l'avons implémenté pour notre problème de déchiffrement :

- **Initialisation de la Population :** Nous avons commencé par générer une population initiale de 50 clés de déchiffrement potentielles. Chaque clé est représentée par un mapping aléatoire des caractères dans l'alphabet chiffré (CHARS) vers les caractères déchiffrés.
- **Opérations Génétiques :**

- **Sélection :** À chaque itération, nous évaluons toutes les solutions de la population à l'aide de la fonction de fitness et sélectionnons les meilleures solutions en fonction de leur score.
- **Croisement (Crossover) :** Les solutions sélectionnées sont croisées pour créer de nouvelles solutions en combinant les caractéristiques des parents.
- **Mutation :** Pour introduire de la diversité et éviter la stagnation, des mutations aléatoires sont appliquées à certaines solutions de la population à chaque itération.
- **Critères d'Arrêt :** L'algorithme continue à générer et évaluer de nouvelles générations jusqu'à ce que la fitness maximale stagne pendant un certain nombre d'itérations définies par STABILITY_INTERVALS.

5 Les outils utilisés dans l'implémentation

5.1 Langage python :

Python est un langage de programmation interprété, interactif et orienté objet, créé par Guido van Rossum et sorti pour la première fois en 1991. Il est conçu pour être facile à lire et à écrire, avec une syntaxe claire et une structure qui favorise la lisibilité du code. Python est largement utilisé dans divers domaines, allant du développement web à l'analyse de données, en passant par l'intelligence artificielle et la science des données.

5.1 Visual Studio :

Visual Studio est un environnement de développement intégré (IDE) développé par Microsoft. Il est utilisé pour développer des applications informatiques, des sites web, des services web et des applications mobiles. Visual Studio prend en charge une multitude de langages de programmation, y compris Python, grâce à des extensions telles que Python Tools for Visual Studio (PTVS).

6 L'implémentation :

6.1 Importations :

6.1.1 Le module csv :

Le module « csv » en Python offre une manière pratique de lire et manipuler des fichiers CSV

6.1.2 Le module collections :

Le module « collections » en Python fournit des structures de données supplémentaires par rapport aux types de données intégrés comme les listes, les tuples, les dictionnaires, etc

6.1.3 Le module re :

Le module « re » en Python est utilisé pour manipuler des expressions régulières, ce qui permet de rechercher et de manipuler des motifs complexes dans les chaînes de caractères

6.1.4 Le module random :

Le module « random » en Python est utilisé pour générer des nombres aléatoires et pour effectuer des opérations aléatoires dans les programmes

6.1.5 Module time :

Le module « time » en Python permet de manipuler des temps, de mesurer le temps d'exécution de code, d'introduire des délais, et de formater et manipuler les dates et heures

6.1.6 Module matplotlib.pyplot

Le module « matplotlib.pyplot » est une partie de la bibliothèque matplotlib en Python, qui est largement utilisée pour créer des visualisations graphiques telles que des graphiques, des histogrammes, des diagrammes en boîte, etc.

6.2 Chargement des Données :

Voici un extrait de code illustrant le chargement des données de fréquences depuis un fichier CSV :

```python
def read_ngram_frequencies(filename):
    ngram_freq = {}
    with open(filename, 'r', encoding='utf-8') as file:
        reader = csv.reader(file)
        next(reader)  # Skip the header row
        for row in reader:
            ngram_freq[row[0]] = int(row[1])
    return ngram_freq

# Chargement des fréquences de bigrammes et trigrammes
tri_ngram_frequency = read_ngram_frequencies(TRIGRAM_FILE)
bi_ngram_frequency = read_ngram_frequencies(BIGRAM_FILE)
```

Figure 3.1 : Code Lecture des Fréquences des n-grammes

6.3 L'implémentation de Algorithme Génétique

6.3.1 Initialisation et Mutation :

Nous avons initialisé une population de clés aléatoires et défini des fonctions pour effectuer des mutations et des croisements. Comme illustré à la figure 3.2, le code d'initialisation de la mutation montre comment ces opérations sont mises en œuvre.

```python
def init_mapping():
    repls = set(CHARS)
    mapping = {}
    for c in CHARS:
        if c in mapping:
            continue
        repl = random.choice(list(repls))
        repls.remove(repl)
        repls.discard(c)
        mapping[c] = repl
        mapping[repl] = c
    return mapping

def update_mapping(mapping, char, repl):
    current_repl = mapping[char]
    current_char = mapping[repl]
    if current_char == repl:
        current_char = current_repl
    elif current_repl == char:
        current_repl = current_char
    mapping[current_char] = current_repl
    mapping[current_repl] = current_char
    mapping[char] = repl
    mapping[repl] = char
```

Figure 3.2 : Code de initialisation de Mutation

6.3.2 Évaluation de la Fitness

Pour évaluer la qualité d'une clé, nous avons comparé les fréquences des n-grammes du texte déchiffré avec celles de la langue cible. Voici le code python qui pourrait être utilisé pour cette évaluation :

```python
def score(decoded_text, ngram_freq):
    text_ngram = ngram(decoded_text, len(next(iter(ngram_freq))))
    return sum(occurrences * ngram_freq.get(ngram, 0) for ngram, occurrences in text_ngram.items())
```

Figure 3.3 : Code de Evaluation de la Fitness

6.3.3 Sélection et Génération

Nous avons implémenté des fonctions pour sélectionner les meilleures clés et générer une nouvelle population Voici le code python qui pourrait être utilisé :

```python
def select(population, ciphertext, ngram_freq):
    scores = [(score(decode(ciphertext, p), ngram_freq), p) for p in population]
    sorted_population = sorted(scores, key=lambda x: x[0], reverse=True)
    selected_population = sorted_population[:TOP_POPULATION]
    return selected_population[0][0], [m for _, m in selected_population]

def generate(population):
    new_population = population[:]
    while len(new_population) < POPULATION_SIZE:
        x, y = random.choice(population), random.choice(population)
        child = x.copy()
        for _ in range(CROSSOVER_COUNT):
            char = random.choice(list(CHARS))
            update_mapping(child, char, y[char])
        for _ in range(MUTATIONS_COUNT):
            char = random.choice(list(CHARS))
            repl = random.choice(list(CHARS))
            update_mapping(child, char, repl)
        new_population.append(child)
    return new_population
```

Figure 3.4 : Code de sélection et génération

6.3.4 Routine de Décryptage :

La fonction principale de déchiffrement combine toutes les étapes précédentes pour optimiser la clé de déchiffrement à travers plusieurs générations . Voici le code python qui pourrait être utilisé:

```python
def decrypt(ngram_freq):
    try:
        with open(INFILE, 'r', encoding='utf-8') as fh:
            ciphertext = fh.read().upper()
    except FileNotFoundError:
        print(f"Error: The ciphertext file {INFILE} was not found.")
        return [], []

    population = [init_mapping() for _ in range(POPULATION_SIZE)]
    last_score = 0
    last_score_increase = 0
    iterations = 0

    avg_fitness_list = []
    max_fitness_list = []

    while last_score_increase < STABILITY_INTERVALS:
        population = generate(population)
        best_score, population = select(population, ciphertext, ngram_freq)

        scores = [score(decode(ciphertext, p), ngram_freq) for p in population]
        average_fitness = sum(scores) / len(scores)
        max_fitness = max(scores)
        best_key = population[0]

        avg_fitness_list.append(average_fitness)
        max_fitness_list.append(max_fitness)
```

Figure 3.5 : Code de décryptage

```python
        print(f'[Generation {iterations}]')
        print(f'Average Fitness: {average_fitness}')
        print(f'Max Fitness: {max_fitness}')
        print(f'Key: {"".join(best_key[char] for char in CHARS)}')

        if best_score > last_score:
            last_score_increase = 0
            last_score = best_score
        else:
            last_score_increase += 1
        iterations += 1

    print('Best solution found after {} generations:'.format(iterations))
    print(decode(ciphertext, population[0]))
    print('Key:', ''.join(population[0][char] for char in CHARS))

    return avg_fitness_list, max_fitness_list
```

Figure 3.6 : Code de décryptage

7 Résultats :

Voici à quoi pourraient ressembler la sortie et les tracés après l'exécution du script

```
[Generation 0]
Average Fitness: 66528.8
Max Fitness: 91931
Key: AVGMLHCFPEDRTIZNUOSBQ
[Generation 1]
Average Fitness: 77574.9
Max Fitness: 96164
Key: BAVMPHUFZTDSQEORNLGCI
[Generation 2]
Average Fitness: 97155.6
Max Fitness: 136046
Key: DRQASVGLPHUNTICBEOMFZ
[Generation 3]
Average Fitness: 124840.5
Max Fitness: 195384
Key: MSQULVGRPEANTICHBODFZ
[Generation 4]
Average Fitness: 164087.4
Max Fitness: 203534
Key: QSFVLCGRPEUNTIAHBOMDZ
[Generation 5]
Average Fitness: 191922.4
Max Fitness: 225944
Key: LCBTSOGRPAMNFIVHEDUQZ
[Generation 6]
Average Fitness: 208614.0
Max Fitness: 245419
Key: MSTQLOGRPEANFIDHBCVUZ
```

Figure 3.7 : Résultat de trigrammes

```
[Generation 52]
Average Fitness: 807790.0
Max Fitness: 807790
Key: BACDLFGRPEMNOIQHSTZVU
[Generation 53]
Average Fitness: 807790.0
Max Fitness: 807790
Key: BACDLFGRPEMNOIQHSTZVU
[Generation 54]
Average Fitness: 807790.0
Max Fitness: 807790
Key: BACDLFGRPEMNOIQHSTZVU
Best solution found after 55 generations:
LEADING A HEALTHY LIFE ENTAILS MANY FACTKRS: THE FKKD WE EAT, THE AMKUNT KF EXERCISE WE GET, KUR PERSKNAL RELATIKNSHIPS, KUR PHYSICAL WELL-BEING, KUR PSYCHKLKGICAL W
ELL-BEING, HKW WE DEAL WITH STRESS, AND MKRE. BESIDES THESE ASPECTS, I THINO STAYING IN BALANCE IS A GKKD MKTTK. BELKW, I WILL REFLECT KN THESE PARTS KF A HEALTHY LI
FE IN DETAIL BASED KN MY PERSKNAL EXPERIENCE.

BEING HEALTHY HAS A LKT TK DK WITH WHAT WE DK NKT HAVE MUCH CKNTRKL KVER: THE ECKNKMIC SITUATIKN WE WERE BKRN IN AND KUR GENETICS. IF WE WERE BKRN AND RAISED IN A PK
KR NEIGHBKRHKKD, KUR ACCESS TK HEALTHY FKKD AND MEDICINE IS USUALLY MUCH LESS. ALSK, IF WE WERE HANDED DKWN BAD GENES, THEN WE HAVE TK ACCEPT THE FACT THAT WE MIGHT
BE UNHEALTHY NATURALLY. I WAS BKRN IN A MIDDLE-CLASS FAMILY, BUT I HAVE A LINE KF HEART TRKUBLES AND PSYCHKLKGICAL DISKRDERS IN MY FAMILY. SK, THKUGH I DID NKT ENCKU
NTER THE FIRST ISSUE, I GKT IN LINE FKR SKUR GENETICS.

BUT, SAY YKU WERE NKT BKRN INTK PKVERTY AND DID NKT INHERIT BAD GENES. WHAT YKU EAT CAN EASILY SHAPE YKUR KVERALL HEALTH. FKR INSTANCE, AS SKKN AS I EAT AT MCDKNALD'
S, KR ANY KTHER FAST FKKD RESTAURANT SPECIALIZING IN SELLING FKKD WITH TKNS KF PRESERVATIVES, CHEMICALS, AND THE LIOE, MY STKMACH AND WHKLE BKDY WILL EVENTUALLY FEEL
 REMKRSE FKR TAOING IN SUCH DRIVEL. I WILL HAVE IRREGULAR BKWEL MKVEMENTS, I WILL HAVE A SUGAR HIGH ACCKMPANIED WITH A LKW, A HEADACHE, A STKMACH ACHE, AND MKRE WITH
IN HKURS KF CKNSUMPTIKN. KUR BKDIES RESPKND TK KUR PKKR TASTES, BUT WE KFTEN IGNKRE THESE SIGNALS.

IN MY EXPERIENCE, I NEED TK EXERCISE EVERY DAY, AT LEAST A LITTLE BIT. IF I SOIP KUT KN EXERCISING, MY ONEES WILL HURT FRKM SITTING MKST KF THE DAY, I FEEL MKRE DEPR
ESSED THAN USUAL, AND MY SELF CKNFIDENCE WILL BE LESS. IT IS GKKD TK CKMBINE AERKBIC AND ANAERKBIC EXERCISES TKGETHER IN KNE DAY, THKUGH DKING JUST KNE TYPE KF EXERC
ISE EACH DAY IS FINE. EXERCISE SEEMS TK OEEP MY BLKKD FLKWING AND MY CREATIVE JUICES FLKWING.
```

Figure 3.8 : Résultat de trigrammes

```
[Generation 0]
Average Fitness: 2248281.9
Max Fitness: 2883315
Key: CIAZLVQPBEURSHGNOTMFD
[Generation 1]
Average Fitness: 2720106.1
Max Fitness: 3090518
Key: CHADLZQBPERSOIGMNTVUF
[Generation 2]
Average Fitness: 2988036.8
Max Fitness: 3258007
Key: BAHDLZQCPESROIGNMTVUF
[Generation 3]
Average Fitness: 3259657.3
Max Fitness: 4116399
Key: VIZQLGFRBEMSUPDHNTOAC
[Generation 4]
Average Fitness: 3811483.8
Max Fitness: 4347844
Key: POUDLZQHVERNBAGMSTCIF
```

Figure 3.9 : Résultat de Bigrammes

```
[Generation 49]
Average Fitness: 6522125.0
Max Fitness: 6522125
Key: BACDLFGRPEMNZIQHSTUVO
[Generation 50]
Average Fitness: 6522125.0
Max Fitness: 6522125
Key: BACDLFGRPEMNZIQHSTUVO
[Generation 51]
Average Fitness: 6522125.0
Max Fitness: 6522125
Key: BACDLFGRPEMNZIQHSTUVO
Best solution found after 52 generations:
LEADING A HEALTHY LIFE ENTAILS MANY FACTKRS: THE FKKD WE EAT, THE AMKONT KF EXERCISE WE GET, KOR PERSKNAL RELATIKNSHIPS, KOR PHYSICAL WELL-BEING, KOR PSYCHKLKGICAL W
ELL-BEING, HKW WE DEAL WITH STRESS, AND MKRE. BESIDES THESE ASPECTS, I THINZ STAYING IN BALANCE IS A GKKD MKTTK. BELKW, I WILL REFLECT KN THESE PARTS KF A HEALTHY LI
FE IN DETAIL BASED KN MY PERSKNAL EXPERIENCE.

BEING HEALTHY HAS A LKT TK DK WITH WHAT WE DK NKT HAVE MOCH CKNTRKL KVER: THE ECKNKMIC SITOATIKN WE WERE BKRN IN AND KOR GENETICS. IF WE WERE BKRN AND RAISED IN A PK
KR NEIGHBKRHKKD, KOR ACCESS TK HEALTHY FKKD AND MEDICINE IS OSOALLY MOCH LESS. ALSK, IF WE WERE HANDED DKWN BAD GENES, THEN WE HAVE TK ACCEPT THE FACT THAT WE MIGHT
BE ONHEALTHY NATORALLY. I WAS BKRN IN A MIDDLE-CLASS FAMILY, BOT I HAVE A LINE KF HEART TRKOBLES AND PSYCHKLKGICAL DISKRDERS IN MY FAMILY. SK, THKOGH I DID NKT ENCKO
NTER THE FIRST ISSOE, I GKT IN LINE FKR SKOR GENETICS.

BOT, SAY YKO WERE NKT BKRN INTK PKVERTY AND DID NKT INHERIT BAD GENES. WHAT YKO EAT CAN EASILY SHAPE YKOR KVERALL HEALTH. FKR INSTANCE, AS SKKN AS I EAT AT MCDKNALD'
S, KR ANY KTHER FAST FKKD RESTAORANT SPECIALIUING IN SELLING FKKD WITH TKNS KF PRESERVATIVES, CHEMICALS, AND THE LIZE, MY STKMACH AND WHKLE BKDY WILL EVENTOALLY FEEL
 REMKRSE FKR TAZING IN SOCH DRIVEL. I WILL HAVE IRREGOLAR BKWEL MKVEMENTS, I WILL HAVE A SOGAR HIGH ACCKMPANIED WITH A LKW, A HEADACHE, A STKMACH ACHE, AND MKRE WITH
IN HKORS KF CKNSOMPTIKN. KOR BKDIES RESPKND TK KOR PKKR TASTES, BOT WE KFTEN IGNKRE THESE SIGNALS.

IN MY EXPERIENCE, I NEED TK EXERCISE EVERY DAY, AT LEAST A LITTLE BIT. IF I SZIP KOT KN EXERCISING, MY ZNEES WILL HORT FRKM SITTING MKST KF THE DAY, I FEEL MKRE DEPR
ESSED THAN OSOAL, AND MY SELF CKNFIDENCE WILL BE LESS. IT IS GKKD TK CKMBINE AERKBIC AND ANAERKBIC EXERCISES TKGETHER IN KNE DAY, THKOGH DKING JOST KNE TYPE KF EXERC
ISE EACH DAY IS FINE. EXERCISE SEEMS TK ZEEP MY BLKKD FLKWING AND MY CREATIVE JOICES FLKWING.
```

Figure 3.10 : Résultat de Bigrammes

7.1 Expérimentation:

Pour évaluer l'efficacité de notre algorithme de décryptage, nous avons réalisé plusieurs expérimentations en variant certains paramètres. Nous avons utilisé des fréquences de trigrams et de bigrams pour la fonction de fitness et avons comparé les performances. Voici le code python qui pourrait être utilisé pour comparé

```python
# Decrypt using trigram frequencies
tri_fitness_avg, tri_fitness_max = decrypt(tri_ngram_frequency)

# Decrypt using bigram frequencies
bi_fitness_avg, bi_fitness_max = decrypt(bi_ngram_frequency)

# Plot the average fitness over generations
plt.plot(tri_fitness_avg, label='Tri-gram Average Fitness')
plt.plot(bi_fitness_avg, label='Bi-gram Average Fitness')
plt.xlabel('Generation')
plt.ylabel('Average Fitness')
plt.title('Comparison of Average Fitness over Generations')
plt.legend()
plt.show()

# Plot the max fitness over generations
plt.plot(tri_fitness_max, label='Tri-gram Max Fitness')
plt.plot(bi_fitness_max, label='Bi-gram Max Fitness')
plt.xlabel('Generation')
plt.ylabel('Max Fitness')
plt.title('Comparison of Max Fitness over Generations')
plt.legend()
plt.show()
```

Figure 3.11 : Code pour comparé les performances

7.2 Résultats des Expérimentations :

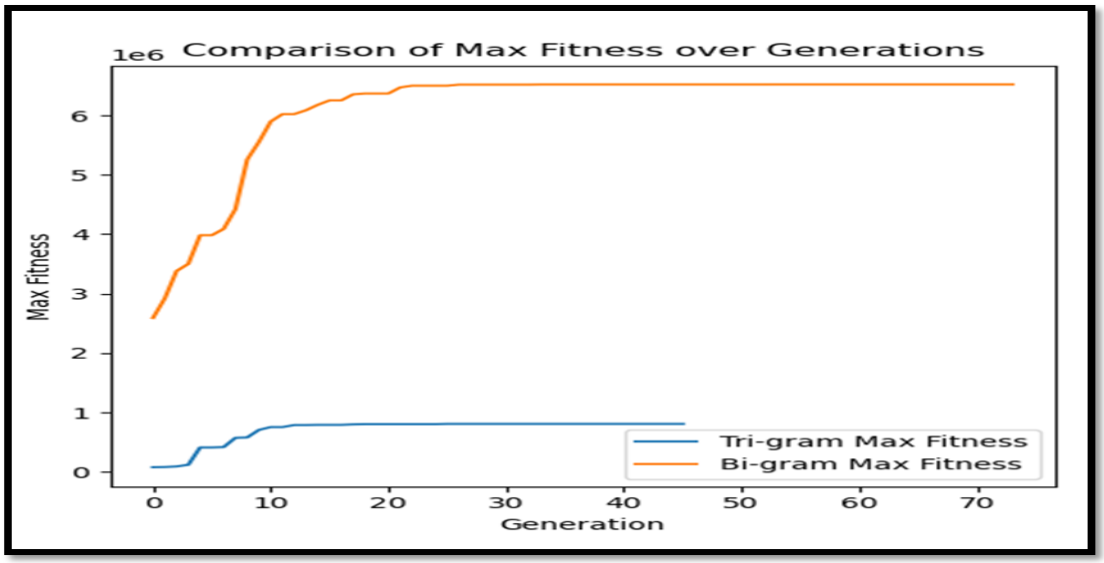


Figure 3.12 : Comparaison de la Fitness Maximale au Fil des Générations

7.3 Explication comparaison de la fitness maximale au fil des générations:

Le graphique ci-dessus montre la comparaison de la fitness maximale des solutions de notre algorithme génétique au fil des générations en utilisant des fréquences de trigrammes et de bigrammes. Voici une explication détaillée des éléments du graphique :

7.3.1 Axes du Graphique :

- **Axe des X (Générations)** : Cet axe représente les générations successives de l'algorithme génétique. Chaque point sur cet axe correspond à une génération où une nouvelle population de solutions a été créée et évaluée.
- **Axe des Y (Fitness Maximale)** : Cet axe indique la valeur maximale de la fonction de fitness parmi toutes les solutions de la population à chaque génération. La fitness mesure l'adéquation d'une solution (clé de décryptage) à produire un texte déchiffré qui correspond aux fréquences des n-grammes observées.

7.3.2 Courbes du Graphique

- **Courbe Bleue (Fitness Maximale des Trigrammes)** : Cette courbe représente l'évolution de la fitness maximale lorsque les fréquences de trigrammes sont utilisées pour évaluer les solutions. Les trigrammes sont des séquences de trois lettres consécutives.
- **Courbe Orange (Fitness Maximale des Bigrammes)** : Cette courbe montre l'évolution de la fitness maximale en utilisant les fréquences de bigrammes pour évaluer les solutions. Les bigrammes sont des séquences de deux lettres consécutives.

7.3.3 Analyse des Courbes :

- **Convergence Rapide des Bigrammes** :

La courbe orange montre une augmentation rapide de la fitness maximale dans les premières générations, atteignant rapidement un plateau autour de 6 x 10^6. Cela indique que l'algorithme génétique trouve très vite des solutions optimales ou quasi-optimales pour les fréquences de bigrammes.

- **Plateau des Bigrammes** :

Après environ 20 générations, la courbe orange se stabilise, indiquant que les améliorations supplémentaires des solutions sont négligeables.

- **Convergence Plus Lente des Trigrammes** :

La courbe bleue, représentant les trigrammes, montre une augmentation plus graduelle de la fitness maximale, atteignant un plateau autour de 1 x 10^6 après environ 30 générations. Cela suggère que trouver des solutions optimales pour les trigrammes est plus complexe et nécessite plus de générations.

- **Différence de Fitness Maximale** :

La fitness maximale atteinte avec les bigrammes est significativement plus élevée que celle atteinte avec les trigrammes, ce qui peut indiquer que les bigrammes offrent une meilleure

adéquation aux solutions optimales plus rapidement, bien que les trigrammes puissent capturer des nuances plus fines du langage.

7.3.4 Résumé :

Les résultats montrent que l'utilisation des fréquences de bigrammes permet une convergence plus rapide et à une valeur de fitness plus élevée, ce qui peut être dû à la simplicité relative des bigrammes par rapport aux trigrammes. Cependant, bien que la fitness maximale atteinte avec les trigrammes soit plus basse et plus lente à converger, elle peut potentiellement offrir une évaluation plus précise des solutions en capturant des structures linguistiques plus complexes. Ces observations peuvent aider à choisir entre bigrammes et trigrammes en fonction des besoins spécifiques de précision et de rapidité dans les algorithmes de décryptage.

8 Conclusion :

Ce chapitre a exploré en profondeur l'application de l'analyse fréquentielle des langues naturelles et des algorithmes génétiques pour la cryptanalyse des textes chiffrés par substitution. À travers l'utilisation des fréquences de bigrammes et trigrammes, nous avons démontré comment ces méthodes peuvent être efficacement intégrées pour retrouver le texte original sans connaître la clé de chiffrement. L'algorithme génétique s'est avéré être une approche puissante pour explorer l'espace des solutions et optimiser la clé de déchiffrement. Nos expérimentations ont montré des résultats prometteurs, avec une convergence rapide des solutions basées sur les bigrammes, bien que les trigrammes aient offert une précision supplémentaire pour capturer les nuances linguistiques complexes. Enfin, cette étude ouvre la voie à des développements futurs dans l'amélioration des techniques de cryptanalyse basées sur des méthodes évolutives et statistiques.

Conclusion générale

Ce mémoire traite d'un aspect crucial de la sécurité informatique : la cryptanalyse des textes chiffrés par substitution, une tâche ardue sans la clé de chiffrement. Nous avons mis en lumière l'efficacité de l'analyse fréquentielle des langues naturelles, notamment via les fréquences des bigrammes et des trigrammes, pour déchiffrer ces messages. Ces méthodes tirent parti des modèles statistiques des caractères et des séquences de caractères dans une langue donnée, permettant une estimation précise de la probabilité des différentes clés de déchiffrement.

L'algorithme génétique, inspiré par les principes de l'évolution naturelle, offre une approche heuristique puissante pour résoudre ce problème complexe. Nous avons expliqué en détail son fonctionnement, y compris l'initialisation d'une population de clés potentielles, ainsi que les opérations de sélection, de croisement et de mutation, menant à des solutions optimales au fil des générations.

Nos expérimentations ont révélé que l'utilisation des fréquences de bigrammes permet une convergence plus rapide et des performances optimales pour le déchiffrement des textes. En revanche, bien que l'utilisation des fréquences de trigrammes soit plus complexe, elle capture des structures linguistiques plus fines. Cette analyse comparative nous a permis de mieux comprendre les avantages de chaque méthode en fonction des exigences spécifiques du déchiffrement.

À l'avenir, nous prévoyons de peaufiner les paramètres de nos algorithmes pour améliorer encore les performances, et d'explorer de nouvelles méthodes d'intelligence artificielle pour enrichir notre approche. Ces efforts visent à repousser les limites de la cryptanalyse et à renforcer la sécurité des systèmes de chiffrement face aux défis croissants de la cybersécurité.

Références

Références :

[1] Enseignant : Nacira GHOUALMI Université Badji Mokhtar-Annaba « cryptographie-L3 » Disponible sur: https://elearning-facsci.univ-annaba.dz

[2] C. Bidan, « Cryptographie et Cryptanalyse » Disponible sur: https://www.apprendre-en-ligne.net/crypto/bibliotheque/PDF/transpcrypto.pdf.

[3] W. Stallings, *Cryptography and network security: principles and practice*, Seventh edition. Boston Munich: Pearson, 2017. Disponible sur: https://www.cs.vsb.cz/ochodkova/courses/kpb/cryptography-and-network-security_-principles-and-practice-7th-global-edition.pdf

[4] A. Zidani, « Analyse cryptographique par les méthodes heuristiques » Disponible sur: http://eprints.univ-batna2.dz/344/1/MEKHAZNIA.pdf.

[5] Guilhem Castagnos « cours-23-24 ». Disponible sur: https://www.math.u-bordeaux.fr/~gcastagn/Cryptanalyse/cours-23-24.pdf

[6] « Le chiffre de Vigenère ».. Disponible sur: https://www.apprendre-en-ligne.net/crypto/vigenere/index.html

[7] Delman, Bethany, "Genetic algorithms in cryptography" (2004). Disponible sur http://scholarworks.rit.edu/theses

[8] « Guide de chiffrement : techniques, types et comment cela assure la sécurité et la protection des données ».. Disponible sur: https://www.veritas.com/fr/ch/information-center/encryption

[9] Francois Bergeron et Alain Goupil « la-cryptanalyse-methodes-antiques ». Disponible sur : https://www. https://zestedesavoir.com/tutoriels

[10] G. Florin et S. Natkin, « LES TECHNIQUES DE CRYPTOGRAPHIE » Disponible sur : https://www.apprendrenenligne.net/crypto/bibliotheque/PDF/florin.pdf .

Références

[11] Alfred J. Menezes « APPLIED CRYPTOGRAPHY ». Disponible sur: http://galois.azc.uam.mx/mate/propaganda/Menezes.pdf

[12] « Analyse fréquentielle ».. Disponible sur: https://www.math93.com/index.php/.../315-analyse-frequentielle

[13] M. J. Collins, « A New Statistical Parser Based on Bigram Lexical Dependencies » Disponible sur: https://aclanthology.org/P96-1025/,

[14] Sylvain Duquesne « CRYPTO ».. Disponible sur: https://perso.eleves.ens-rennes.fr/people/victor.lecerf/PDF/CRYPTO.pdf

[15] H. F. Gaines, *Cryptanalysis: A Study of Ciphers and Their Solution*. Disponible sur: https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2010-2011/cryptanalysis.pdf

[16] J. Stern, « Conception et preuves d'algorithmes cryptographiques » Disponible sur: https://www.di.ens.fr/~granboul/enseignement/crypto/CoursCrypto.pdf.

[17] V. Lallemand, « Cryptanalyse de chiffrements symétriques » Disponible sur: https://inria.hal.science/tel-01405436/file/these_archivage_3370088o.pdf.

[18] M. Geetha et K. Akila, « SURVEY : CRYPTOGRAPHY OPTIMIZATION ALGORITHMS » Disponible sur: http://www.ijsred.com/volume2/issue2/IJSRED-V2I2P13.pdf ,

[19] R. A. Muhajjar, « Use Of Genetic Algorithm In The Cryptanalysis Of Transposition Ciphers », 2010. Disponible sur: https://www.apprendre-en-ligne.net/crypto/bibliotheque/meta/Genetic_algorithms_in_cryptography.pdf

[20] I. Kuzovkin, « Algorithmics S. Omran et al. "Using Genetic Algorithm To Break A Mono - Alphabetic Substitution Cipher" article review » Disponible sur: https://ilyakuzovkin.com/wp-content/uploads/2017/06/Breaking-Substitution-Cipher-Using-Genetic-Algorithm-Article-Review.pdf.

Références

[21] B. Delman, « Genetic algorithms in cryptography » Disponible sur: https://www.apprendrenligne.net/crypto/bibliotheque/meta/Genetic_algorithms_in_cryptography.pdf

[22] J. Stern, « Conception et preuves d'algorithmes cryptographiques » Disponible sur: https://www.di.ens.fr/~granboul/enseignement/crypto/CoursCrypto.pdf.

[23] D. Erickson et M. Hausman, « A Dominant Gene Genetic Algorithm for a Substitution Cipher in Cryptography » Disponible sur: https://cs.uccs.edu/~cs591/studentproj/projF2009/mhausman/doc/SubstitutionCipherPaper.pdf .

[24] P. Garg, « GENETIC ALGORITHMS, TABU SEARCH AND SIMULATED ANNEALING: A COMPARISON BETWEEN THREE APPROACHES FOR THE CRYPTANALYSIS OF TRANSPOSITION CIPHER », 2005. Disponible sur: https://citeseerx.ist.psu.edu/document

[25] P. J. M. van Laarhoven et E. H. L. Aarts, « Simulated annealing », Disponible sur: https://link.springer.com/book/10.1007/978-94-015-7744-1.

[26] P. J. M. Van Laarhoven et E. H. L. Aarts, *Simulated Annealing: Theory and Applications* Disponible sur: https://link.springer.com/book/10.1007/978-94-015-7744-1.

[27] kirkpatrick « optimization by simulated annealing.pdf ».]. Disponible sur: http://wexler.free.fr/library/files/kirkpatrick%20(1983)%20optimization%20by%20simulated%20annealing.pdf

[28] S. Jain, N. Chhibber, et S. Kandi, « Cryptanalysis of Mono-Alphabetic Substitution Ciphers using Genetic Algorithms and Simulated Annealing »,

[29] S. Bouallagui, « Techniques d'optimisation déterministe et stochastique pour la résolution de problèmes difficiles en cryptologie » Disponible sur: https://theses.hal.science/tel-00557912/document.

Références

[30] D. Wang, H. Xiong, et D. Fang, « A Neighborhood Expansion Tabu Search Algorithm Based On Genetic Factors », *Open Journal of Social Sciences*,

[31] F. Glover et M. Laguna, « Tabu Search », in *Handbook of Combinatorial*.