# Extension et parallèlisation d'un algorithme de chiffrement évolutionnaire basé occurrences

Mohamed lamine Semassel[1], Ismahane Souici[2], Hamid Seridi[3, 4]

[1] Université 20 août 55 Skikda, Algérie
[2] Université de Guelma, Algérie
[3] LAIG, Université de Guelma, Algérie
[4] Université de Reims, France

{semassel_medlamine, souici.ismahane, seridi}@yahoo.fr

**Résumé.** Ce papier présente une tentative d'extension de l'Algorithme de Chiffrement Evolutionnaire basé Occurrences (ACEO) permettant le chiffrement de messages regroupant la majorité des caractères du code ASCII. Pour pouvoir traiter des messages écrits suivant d'autres langues que celles adoptées par l'ACEO, telle que la langue chinoise, l'hébreu, …, nous avons pensé à augmenter l'espace de recherche par les caractères de ces langues. Cette augmentation va entrainer un allongement du temps de calcul. Pour éviter cela, la solution la plus prometteuse sera de faire une parallélisation de cet algorithme en utilisant le modèle en îlots. Ce modèle repose sur la division de la population en petites sous populations évoluant chacune sur un processeur suivant un schéma fonctionnel et en envoyant ses meilleurs individus soit vers une population voisine, soit dans un pool commun. Une étape dite de migration est ensuite appliquée sur les sous populations, où chacune d'entre elles reçoit des individus soit envoyés par ses populations voisines soit péchées dans le pool central. L'intérêt de cette méthode de parallélisation, est que chaque sous population évolue selon des paramètres différents, c'est pour cela qu'il doit y avoir des bon paramètres pour produire des meilleures solutions à chaque moment de l'évaluation.

**Mot clés:** chiffrement évolutionniste, ACEO, parallélisation, optimisation, extension, modèle en îlots, sous population.

## 1. Introduction

Pendant longtemps la science des secrets était utilisée seulement dans certains domaines plus au moins très sensibles, mais depuis l'avancement considérable des technologies actuelles en matière de traitement de données, de puissance de calcul et de réseaux de télécommunication, elle est devenu, une tâche critique dans un nombre

important d'applications telles que l'historique médical, la sécurité sur internet, la sécurité des réseaux…etc.

Par conséquence, la conception d'algorithmes de chiffrement puissants est devenue plus compliquée, néanmoins il existe des algorithmes de chiffrement qui ont résisté à plusieurs attaques comme : le RSA qui fait parti du groupe asymétrique, l'AES et l'IDEA qui font partie du group symétrique et le PGP [1] qui représente l'hybridation des deux systèmes précédents.

Une nouvelle approche est apparue dans la conception des algorithmes de chiffrement, c'est l'utilisation des algorithmes évolutionnaires inspiré de la théorie de l'évolution naturelle [1]. Ces algorithmes évolutionnaires, par leurs aspect aléatoire et leurs bonnes performances ont prouvé leur efficacité dans la résolution de problèmes réels tel que le SEC (ou encore appelé OTL), SEC-EX et l'ACEO. Dans cet article nous présentons une extension de ce dernier car il assure le maximum de brouillage du texte à chiffrer. Son extension se résume dans : l'augmentation de l'espace de recherche et l'utilisation d'un modèle de parallélisation .

L'article est organisé comme suit : une description de quelques algorithmes de chiffrements évolutionnaire, une description de l'extension de l'ACEO nommée ACEO2, une comparaison entre les différents algorithmes évolutionnaire de chiffrement présentés et une discussion.

## 2. Algorithmes de chiffrement évolutionnaire

### 2.1 Description du SEC

Le SEC est un algorithme de chiffrement évolutionnaire. Dans l'opération de chiffrement il encode un texte $M_0$ constitué de 256 caractères de code ASCII, par l'application d'un ensemble de transformations (substitution, permutation et chiffrement affiné) pour obtenir un texte M. Son principe consiste à construire des listes $L_i$ contenant les différentes positions des caractères $C_i$, puis il joue sur l'ordre de ces listes pour obtenir un désordre maximal à condition de ne pas modifier le contenu des listes. Dans le SEC un individu est un vecteur de taille m et ses gènes sont représentés par les listes $L_i$ ; ensuite, le SEC enchaine le processus évolutionnaire (évaluation, sélection, opérateur de croisement et de mutation). Enfin, une clé dite « clé génétique » est générée [3].

Pour l'opération de déchiffrement, le SEC et grâce à la clé génétique il trouvera les listes des positions des différents caractères puis le texte M sera récupéré. La dernière étape dans l'opération de déchiffrement est l'application des fonctions inverses à celles utilisées pour l'encodage. On obtiendra ainsi le message initial $M_0$ [3].

### 2.2 Description du SEC-EX

La méthode de chiffrement utilisée par l'algorithme SEC peut être appliquée sur n'importe quel ensemble d'entités (caractères, bloc de caractère, bloc de bits etc...).

Pour assurer une protection contre les attaques basées sur l'étude des fréquences des caractères du texte à chiffrer, on coupe ce dernier en blocs de bits après l'application d'une opération de codage binaire et puis on applique l'algorithme évolutionnaire.

Le codage adopté par SEC-EX consiste à : conversion du texte T en binaire, choix aléatoire d'un entier k, coupage du texte T en bloc $B_1$, $B_2$,..., $B_m$ de taille k (si le dernier bloc contient moins de k bits, on le complète avec des bits 0). Nommons $L_i$ la liste des différentes positions du bloc $B_i$ dans T et par Ch-intial le vecteur des gènes $L_i$ ($1 \leq i \leq m$).

L'application de l'algorithme SEC sur les listes des blocs $L_i$ permet le changement de la distribution des listes des différents blocs $B_i$ de T. considérons Ch-final la solution finale obtenue par l'application de cet algorithme. Le texte chiffré est obtenu par l'association de chaque bloc $B_i$ à une liste $L_i$' de Ch-final. Le couple k et les permutations qui transforme Ch-initial vers Ch-final constitue la clé secrète [1].

Concernant l'opération de déchiffrement, et en supposant que T' est le texte chiffré codé en binaire, et que k est la première composante de la clé secrète, il faudra couper T' en blocs $B_1$,$B_2$,.., $B_m$ de même taille k. Grâce à la seconde composante de la clé secrète, les blocs sont affectés vers leurs listes de positions correspondantes dans le texte en clair. Par conséquent, les caractères du texte chiffré sont complètement différents des caractères du texte en clair, de plus leur nombre est augmenté ce qui entraine un changement des fréquences d'apparition après le chiffrement.

### 2.3 Description de l'ACEO

Dans l'ACEO une transformation du message en clair $M_0$ vers un schéma particulier est faite en calculant le nombre d'occurrence, $O_i$, dans $M_0$ des 149 caractères admissibles. Ces 149 nombres d'occurrence représentent les différents gènes et leur ensemble constitue un individu ACEO. Il est à noter que les caractères qui ne figurent pas dans $M_0$ auront des occurrences nulles. Ainsi, la somme des 149 valeurs de gènes égale la longueur du message $M_0$.

Pour augmenter la complexité de la tâche des cryptanalyses, ACEO applique m perturbations [3] sur le chromosome $I_0$, dit initial et représentant le codage du message en clair, pour obtenir un brouillage $I'_0$. Ainsi, même si une cryptanalyse réussit à dégager la façon de construire le message chiffré et à trouver la bonne combinaison des nombres d'occurrence qui forme son codage, ces informations ne seront plus utiles car l'ACEO change la répartition des 149 caractères dans le schéma de codage à chaque instanciation du problème [4].

ACEO utilise OX « Order Cross-over » comme opérateur de croisement. Il consiste à générer des descendants en trois phases avec un taux de 60% à 100% [7]. Pour ce qui est de la mutation, il utilise une permutation aléatoire de deux gènes d'un chromosome et cela avec un taux de 0.1% à 5% [7]. Les individus résultants de ces deux opérateurs seront rajoutés à la population des parents pour les acheminer vers l'étape d'évaluation, où l'ACEO associe des valeurs d'adaptation à chaque d'individu $I_i$ de la population.

Le processus de chiffrement permet de générer une clé dite clé de session et qui varie d'un message à un autre [4].

## 3 Description de l'algorithme proposé ACEO2

Dans l'ACEO2 nous présentons une proposition d'extension de l'algorithme de chiffrement évolutionnaire ACEO pour pouvoir d'un coté traiter des messages qui utilisent d'autres langues. D'un autre coté, pour augmenter la confusion, en compliquant ainsi la tâche des cryptanalystes, l'augmentation de l'espace de recherche semble être une solution adéquate. Mais dans ce cas on risque d'avoir une augmentation considérable du temps de calcul. Donc, on propose l'utilisation du modèle en Îlots comme solution pour pouvoir paralléliser le processus de chiffrement évolutionnaire.

### 3.1 Codage

Dans l'opération de codage, on calcule le nombre d'occurrence $O_i$ pour chaque caractère $C_i$ du texte à chiffrer, puis on met ces derniers dans une table (TCAR) [4] qui regroupe les 149 caractères de l'ACEO, 26 caractères chinois et 32 caractères hébreux. La codification des individus est schématisée dans la figure (**Fig. 1**).

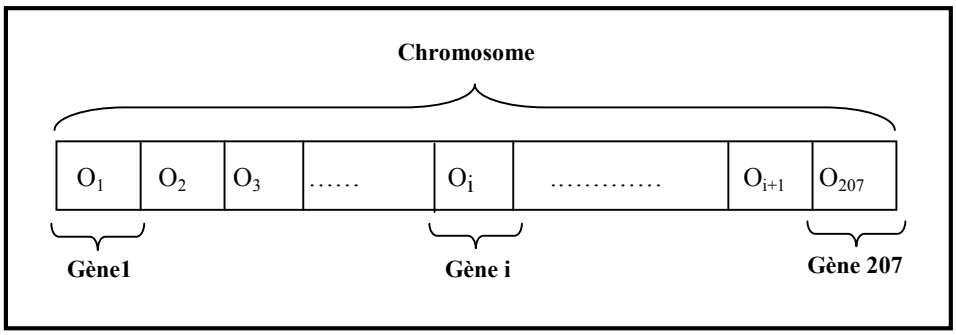


**Fig. 1.** Codage d'un individu

### 3.2 Processus évolutionnaire

Après l'opération du codage, une population initiale est crée par l'application de plusieurs perturbations sur le résultat du codage. Ce dernier change d'une instanciation du problème à une autre [4]. Avec ce nouveau schéma de codage la fonction d'évaluation pour un individu $I_i$ devient :

$$\mathbf{F(I_i)}=\sum_{j=1}^{207} \left| O_{ji} - O_{j0} \right|$$

Avec $I_i = [O_1, O_2, \ldots, O_{207}]$ et $O_{ji}$ est le $J^{ème}$ gène d $i^{ème}$ individu,

Et pour éviter le problème de convergence prématurée on a adopté une sélection aléatoire.

### 3.3 L'ACEO2 parallèle

Pour trouver plus de solutions optimisées dans une courte durée de calcul, la parallèlisation semble une solution indispensable du fait qu'elle réduit le temps de traitement nécessaire pour trouver des solutions acceptables. Ceci est assuré par l'implémentation des AEs sur des architectures parallèles.

Dans la théorie d'équilibre d'évolution proposée par Sewall Wright [12], la vitesse d'évolution des sous-populations est plus rapide que celle des grandes populations. En plus, dans certains cas les sous-populations donnent des solutions bien meilleures grâce à la migration d'individus entre les différentes sous-populations. Cette migration donne aussi des possibilités de recherche globale.

Il exist plusieurs modèles pour implémenter une parallèlisation d'un AE. L'ACEO2 utilise le modèle en îlots présentant des caractéristiques importante du fait que : la population est divisée en petites sous populations. Les figures (**2 et 3)** montrent les deux scénarios choisis par l'ACEO2, et l'introduction d'un opérateur de migration [13].

Dans le modèle de parallélisation adopté (modèle en ilôts), chaque sous population évolue indépendamment des autres. Une étape dite de migration est rattachée au processus. Elle consiste à envoyer les meilleurs individus vers une sous population voisine ou bien vers un pool commun ce qui donne lieu à des coûts relatifs de communications pendant la migration.

Cette conception peut être implémentée même si des machines parallèles ne sont pas disponibles. On peut faire une simulation avec un réseau de station de travail ou avec une seule machine.

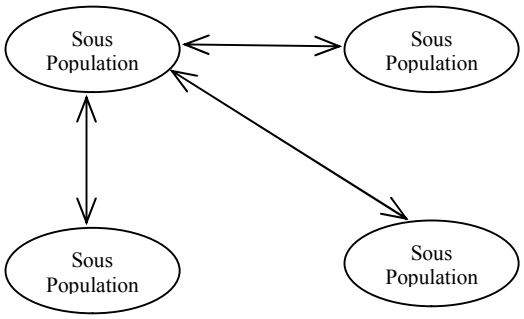


**Fig. 2.** Structure de connexions pour 4 sous-populations

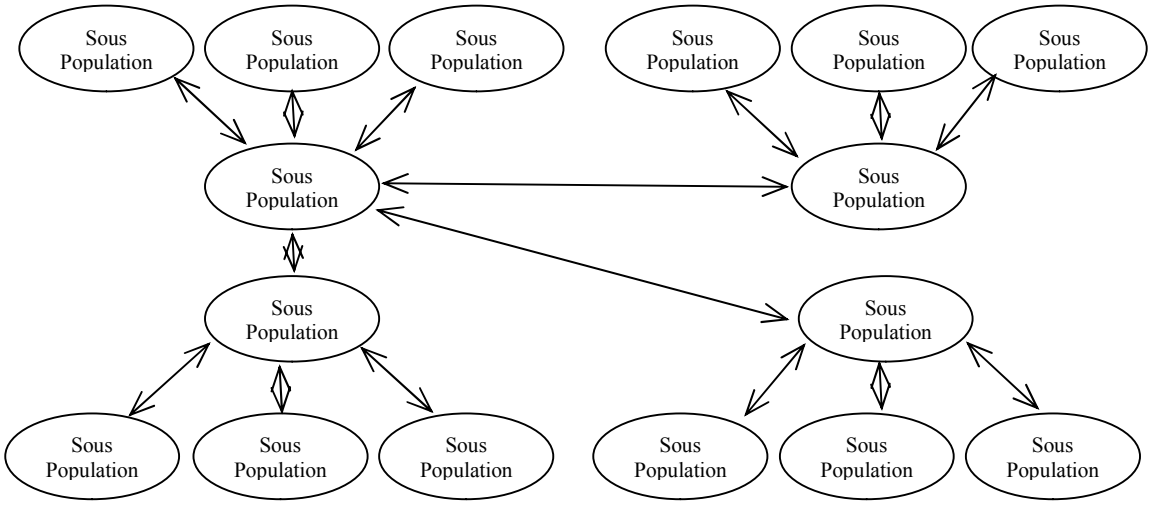


**Fig. 3.** Structure de connexions pour 16 sous-populations

## 4 Discussions

Les différents algorithmes de chiffrement présentés dans cet article font partie de la classe symétrique. Leurs caractères aléatoires et le codage n'utilisent que des informations peu utiles , en plus du fait que le texte chiffré peut contenir des caractères qui ne figurent pas dans le texte en clair ; toutes ces caractéristiques rendent cette nouvelle classe, dite évolutionniste, très puissante contre plusieurs types d'attaques. Ces algorithmes utilisent, aussi, une clé de session générée pendant le processus de chiffrement, cette clé est fortement dépendante du message à chiffrer. Dans le SEC par exemple, le message à chiffrer contient au moins 30 caractères différents ce qui nous ramène à une clé de 240 bits, le même principe est suivi par le SEC-EX. Dans l'ACEO, une clé est de taille égale à 1192 bits vu que le message à chiffrer peut contenir 149 caractères différents. Ceci le rend plus dur contre les attaques exhaustives, mais cela rend le processus de chiffrement plus coûteux en temps de calcul. Pour pallier à ce problème, l'ACEO2 applique le modèle de parallèlisation en îlots (voir 3.3) qui a prouvé son efficacité avec plusieurs problèmes [13]

## 5 Conclusion

Ce travail présente une conception d'une extension d'un algorithme de chiffrement évolutionnaire qui a prouvé sa réussite par rapport à d'autres algorithmes appartenant à la même classe. Cette conception est basée sur l'enrichissement de l'espace de

recherche et l'application d'un modèle efficace pour paralléliser des processus évolutionnaires.

## Références


1. Omary, F., Tragha, A., Bellaachia, A. Mouloudi, A.: Design and valuation of Two Symmetrical Evolutonist-Based Ciphering Algorithm. In: IJCSNS International Journaln of Computer Science and Network Security VOL.7 No.2, February 2007.
2. Omary, F., Tragha, A., Bellaachia, A. Mouloudi, A.: A new Ciphering Method Associated With Evolutionnary Algorithm. In: ICCSA 2006, LNCS 3984, PP. 346-354, 2006. Springer6Verlag Berlin Heidelberg 2006.
3. Omary, F., Tragha, A., Bellaachia, A. Mouloudi, A.: Cryptographie évolutionniste , application des algorithms évolutionniste en cryptographie, Journal Electronique d'Intelligence Artificielle.
4. Souici, I., Seridi, H., Aissaoui, Z. : Nouvel Algorithme de chiffrement évolutionniste ACEO (Algorithme de chiffrement evolutionnaire basé Occurrences), JIG'2007 – 3ème Journées Internationales sur l'informatique Graphique.
5. Caux, C., Pierreval, H., Portmann, M.C.: Les algorithmes génétiques et leurs application aux problèmes d'ordonnancement In: APII, VOL 29-N° 4--5, (1995) 409-443.
6. Goldberg, D.E.: Genetic algorithms insearch optimisation and Machine Learning. Addison-Wesley, Publishing Company, Inc(1989).
7. Grenfenslette, J.J.: Optimization of control parameters for genetic algorithms In: IEEE translation on systems Man, and cybernitics, Vol 16 N°1 (1986) 122-128.
8. KhamPhang Bounsaythip C. : Algorithmes heuristiques et évolutionnistes, Thèse de doctorat université de Lille, Octobre (1988).
9. Labouret G. : Introduction à la cryptographie (2001)., http://www.hsc.fr/ressources/cours/crypto/index.html.fr
10. Meier, W.: On the Security of the IDEA Block Cipher In: Advances in Cryptology EUROCRYPT '93 Proceedings, Springer-Verlag, 1994, PP. 371-385
11. Menzes, A.J,. Paul, C., Dorschot, V., Vanstone, S.A.: Handbook of Applied Cryptography. CRC Press fifth Printing (2001).
12. Belding, T.C,.: The distributed genetic algorithm revisited, In Proceeding of the Sixth International Conference on Genetic Algorithms, PP, 144-121, Morgan Kaufmann, 1995.
13. Paz, E.C,.: Designing scalable multi-population parallel genetic algorithm, In IllGAL Report No. 98009, Illinois Genetic Algorithms Laboratory, 1998.