République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Ferhat ABBAS - Sétif 1 -


*THESE*

Présentée à la Faculté des Sciences

Département d'Informatique
Pour l'Obtention du Diplôme de

*DOCTORAT EN SCIENCES*

Spécialité : Informatique

# Thème

# Contributions à la Résolution de l'Emergence Inversée en Utilisant les Métaheuristiques Quantiques

Présentée par

SEFIA DJEMAME (ép. ZAZOUA)

Soutenue le : 08/12/2018

Devant le jury composé de :

| | | |
|---|---|---|
| Pr. MOUSSAOUI Abdelouahab | Univ. Ferhat Abbas - Sétif 1 | Président |
| Pr. BATOUCHE Mohamed | Univ. Constantine2 | Rapporteur |
| Pr. BENHOCINE Abdelhamid | Univ. Ferhat Abbas - Sétif 1 | Examinateur |
| Pr. CHIKHI Salim | Univ. Constantine2 | Examinateur |
| Pr. SIARRY Patrick | Univ. Paris-Est Créteil- France | Invité |

# Résumé

Les systèmes complexes présentent une propriété intéressante : l'émergence. Cette propriété est présente dans beaucoup de modèles inspirés de la nature. Parmi ces modèles : les automates cellulaires, les colonies de fourmis, les systèmes multi-agents, les essaims de particules, les réseaux de neurones artificiels. La résolution de l'émergence inversée dans ces systèmes consiste à déterminer les règles de base qui permettent à un collectif d'individus simples de coopérer et de produire à un niveau global, une fonction émergente. Dans la littérature, ce problème est qualifié de "difficile" et induit d'intenses recherches. Dans cette optique, nous proposons dans cette thèse trois méthodes de résolution. La première méthode s'appuie sur la métaheuristique PSO, qui guide un processus évolutionnaire d'automate cellulaire. La validation a été faite à travers l'extraction de contours sur images. La deuxième méthode utilise les principes de l'informatique quantique hybridée avec le PSO, à travers la métaheuristique Quantum PSO, pour tirer profit de la diversification de la population, le parallélisme, et la richesse des opérateurs quantiques. Ce modèle a été utilisé pour résoudre deux problèmes : la détection de contours et le filtrage d'images. La troisième méthode est une hybridation entre l'algorithme génétique et l'informatique quantique. L'algorithme obtenu a montré une bonne capacité de recherche globale. Un nombre réduit de chromosomes quantiques a suffi pour étudier le problème.
Les résultats expérimentaux obtenus par les trois méthodes ont démontré la grande capacité des métaheuristiques utilisées pour apporter une solution satisfaisante au problème de l'émergence inversée et assurer une excellente convergence du système et un bon équilibre entre exploitation et exploration .

**Mots clés :** Système complexe, Automate cellulaire, Emergence inversée, Métaheuristiques, Informatique quantique, Particle swarm optimization, Genetic algorithm, Quantum PSO, Quantum GA, Traitement d'images.

# Abstract

Complex systems present an interesting property : emergence. This property exists in several models inspired from nature. Within these models : cellular automata, ant colonies, multi-agent systems, particle swarms, artificial neural networks. The resolution of reverse emergence in these systems consists in extracting the basic rules which allowed the collectif of simple individuals to cooperate and produce at a global level an emergent function. In the litterature, this problem is qualified as "difficult" and induces intense research. In this context, we propose in this thesis three resolution methods. The first method is based on PSO metaheuristic, which guides an evolutionary cellular automata. The validation is done through the extraction of edges on images. The second method uses quantum computing principles, hybridized with PSO, producing Quantum PSO metaheuristic. The purpose is to take advantage from diversification of population, parallelism, and richness of quantum operators. This model is used to resolve two problems : edge detection and filtering of images. The third method is an hybridization between genetic algorithm and quantum computing. The obtained algorithm shows a good global search ability. A reduced number of quantum chromosomes is sufficient to cover the problem.
The experimental results obtained by the three methods demonstrate the good capacity of the metaheuristics used, to provide a satisfactory solution to the problem of reverse emergence, and insure excellent convergence of the system and a good balance between exploitation and exploration.

**Keywords :** Complex systems, Cellular automata, Reverse emergence, Metaheuristics, Quantum computing, Particle swarm optimization, Genetic algorithm, Quantum PSO, Quantum GA, Image processing.

**Dédicaces**

*A la mémoire de mon père, mon premier instituteur,*

*A ma mère, ma première institutrice.*

*A mon mari et mes enfants pour leur soutien, leur patience et leurs encouragements, ça n'a pas été toujours facile !*

*A ma grande famille, mes frères, ma soeur, mon appui indéfectible dans cette vie !*

*Sofia*

# Remerciements

Louange à ALLAH le tout Puissant Qui m'A donné la force et la volonté pour réaliser ce modeste travail, et en venir à bout.

J'adresse mes plus vifs remerciements à mon directeur de thèse : le Professeur Batouche Mohamed Chawki, qui a dirigé cette thèse et en a supervisé le déroulement. Ses orientations, ses conseils judicieux, et ses encouragements ont été un support infaillible pour la finalisation de cette thèse.

Je tiens à exprimer ma gratitude à Pr Moussaoui Abdelouaheb pour avoir accepté de présider mon jury. J'adresse un grand merci à tous les membres de mon jury, pour avoir ainsi marqué leur intérêt pour mon travail : Pr Benhocine Abdelhamid, Pr Chikhi Salim et Pr Patrick Siarry, veuillez trouver ici l'expression de ma profonde gratitude.

Je tiens à rendre hommage à Pr Boukerram Abdellah, ALLAH yrahmou, qui faisait initialement partie de ce jury. Nous nous rappelons de ses qualités humaines et professionnelles, et nous prions ALLAH tout puissant de l'accueillir dans son vaste paradis.

Je tiens à exprimer ma reconnaissance au Professeur Patrick Siarry, et à Dr Hamouche Oulhadj, du laboratoire LISSI, Université Paris 12, qui m'ont accueillie avec bienveillance au sein de leur laboratoire, pour le temps qu'ils m'ont accordé, les conseils prodigués et les corrections apportées à ce travail, et qui ont contribué à son enrichissement.

Je tiens à remercier le staff administratif du département Informatique : Dr Khentout Chabane, Dr Lekhfif Abdelaziz, Dr Toumi Lyazid, Dr Alti Adel, ainsi que Mr le Doyen de la faculté des sciences : Pr Louail Layachi, et Mr le vice-doyen chargé de la post-graduation : Dr Mosbah Ammar. J'apprécie toute l'aide et les facilités qu'ils m'ont accordées avec bienveillance.

Je remercie beaucoup celle qui a toujours été à mes côtés, dans les moments de doute, de découragement, de fatigue, de stress...MERCI AHLEM !

Je remercie mes amies et collègues qui ont toujours cru en moi, qui n'ont jamais cessé de m'encourager, et de me pousser à continuer et finaliser ce travail, malgré toutes les difficultés rencontrées : Mme Sabrina Benmahmoud, Melle Nassima Adjissi, Mme Samira Boukerram, Mme Khadidja Harbouche, Mme Aliouet Zibouda, Melle Mechta Djamila...trouvez ici l'expression de ma profonde gratitude.

Enfin je remercie tous ceux qui, de près ou de loin, ont contribué à l'aboutissement de ce travail.

# Table des matières

# Liste des figures

# Liste des tableaux

# Introduction Générale

DANS la nature, beaucoup de systèmes présentent un comportement émergent. Ils produisent des structures complexes, résultant des interactions locales de composants individuels simples. On peut citer comme exemples de systèmes complexes naturels, dans lesquels apparaissent des phénomènes émergents : Le fourragement et la construction de nids par les insectes sociaux (abeilles, fourmis, termites), les motifs en spirale apparaissant dans les cultures d'amibes, les oscillations synchronisées du cerveau, et le déplacement collectif d'animaux (poissons, oiseaux).
La propriété d'émergence est la résultante directe de la complexité des interactions à l'intérieur du système. Les éléments du système interagissent de façon locale. Les interactions en elles même sont simples, mais le nombre important des éléments et le phénomène de rétroaction (feed-back) qu'a le système avec son environnement, produisent un comportement global complexe et intéressant.
De ce fait, le calcul émergent est devenu un domaine de recherche en pleine expansion. Il est basé sur trois principes : la simplicité, le large parallélisme et l'aspect local. Le principe fondamental d'un système émergent est qu'un grand ensemble d'individus basiques, évoluant selon des règles simples, peuvent construire ensemble une structure complexe, sans avoir recours à un contrôle global, ou à une chaine de commandes centralisée (fig. 1).

Si l'on veut exploiter toute la puissance des Systèmes Complexes (SCs) pour la résolution de problèmes, on devra comprendre le phénomène de l'émergence. La question cruciale qui se pose est : quelles sont les règles locales qui permettent d'obtenir un certain comportement global du système ? Cette question est plus connue sous le nom de "problème inverse". [GSD+03] ont qualifié ce problème d'"*extêmement difficile*". Sa résolution reste un grand défi pour les systèmes émergents. Son importance s'est accrue durant les dernières années, au point de devenir un domaine de recherche à part entière.

Afin d'utiliser toutes les potentialités et la puissance offertes par les SCs, et afin de résoudre des problème du monde réel, on doit d'abord les modéliser et les implémenter sur ordinateur. Les Automates Cellulaires (ACs) sont un modèle mathématique puissant, qui permet de simu-

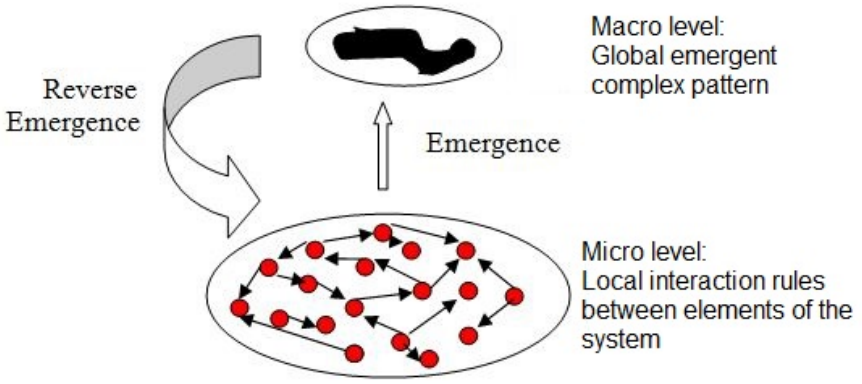


FIGURE 1 – Système complexe et concept d'émergence inversée

ler efficacement le comportement d'un système complexe. Ils ont l'avantage d'être relativement faciles à comprendre et à implémenter. Leurs utilisations pratiques sont très nombreuses, et ont prouvé leur efficacité notamment dans le domaine de traitement d'images. Un AC est une grille de cellules qui changent d'état à chaque instant t. Ce changement d'état est régi par une règle de transition simple, mettant en jeu une cellule et son voisinage immédiat (interactions locales), et produit alors l'état futur de la cellule en question. L'ensemble de toutes ces interactions locales aboutit à la configuration finale du système. Un AC peut avoir un comportement très élaboré et aboutir, au niveau global, à des structures complexes.

## PROBLEMATIQUE ET MOTIVATIONS

Il existe plusieurs manières d'appréhender le problème de l'émergence inversée. Les règles peuvent être générées laborieusement et minutieusement de manière manuelle, ceci est un procédé lent et difficile, et ne convient pas à de larges espaces de problèmes.
Traditionnellement, les approches de résolution rigoureuses, appelées aussi déterministes, sont basées sur les hypothèses, les caractérisations, les déductions et les expérimentations. Elles sont utilisées pour résoudre divers problèmes d'optimisation et aboutissent à des solutions optimales. Mais, souvent elles sont gourmandes en temps d'exécution, surtout lorsqu'elles traitent des problèmes en temps réel. Outre les algorithmes classiques, les méthodes stochastiques, comme les métaheuristiques, sont largement utilisées comme méthodes de résolution pour un large éventail de problèmes d'optimisation. Elles se caractérisent par leur grande puissance et flexibilité et leur capacité à apporter des solutions efficaces à des problèmes difficiles. Les métaheuristiques permettent d'avoir des solutions satisfaisantes et de bonne qualité après un temps d'exécution raisonnable.
Récemment, et afin d'introduire l'automatisation à la résolution du problème d'émergence inversée, il y'a un regain d'intérêt avec l'introduction de méthodes de résolution nouvelles telles

que les algorithmes évolutionnaires, les réseaux de neurones et les métaheuristiques.
Dans cette thèse, nous nous intéressons aux métaheuristiques. Ce sont des techniques générales de résolution, sans être spécifiques à un problème particulier. Elles ont prouvé leur efficacité pour trouver des solutions de bonne qualité pour un large spectre de problèmes d'optimisation "NP-difficiles".
Dans la littérature on a recensé peu de travaux traitant des techniques de recherche pour la résolution de l'émergence inversée, et l'automatisation de la génération de règles, particulièrement en traitement d'images. [MCD+96] a utilisé un algorithme génétique pour résoudre le problème de densité de classification. [Sip97] a utilisé des règles évolutionnaires pour réaliser amincissement et remplissage de creux dans des rectangles. Malgré que ces tâches étaient simples, et les résultats obtenus médiocres, ce travail a démontré la faisabilité de la chose. [ABC98] a généré des AC pour classification de motifs. [Ros06] a réalisé des tâches simples de traitement d'images, en faisant évoluer un AC avec la méthode SFFS. [BMA06] ont appliqué un algorithme génétique avec un AC pour extraire des contours sur des images simples. [CD06] les auteurs ont présenté un algorithme génétique pour générer des formes binaires simples 2D et 3D à partir d'AC.

La majorité des travaux existants ont utilisé l'algorithme génétique pour faire évoluer l'automate cellulaire, rechercher la fonction de transition, et la résolution de l'émergence inversée. Cependant, l'algorithme génétique présente quelques inconvénients :
• Il est coûteux en temps de calcul : certains problèmes nécessitent des jours pour aboutir à la solution.
• Le processus d'apprentissage est lent.
• La recherche de solutions est aveugle, non dirigée, il est difficile d'orienter un algorithme génétique vers une solution optimale, si elle est connue à priori.
• L'algorithme génétique est très sensible aux valeurs initiales des paramètres. La mutation et le croisement peuvent influencer significativement le processus de recherche.
• C'est un algorithme stochastique, il n y'a aucune garantie d'obtention de la solution optimale, mais plutôt une forte probabilité d'y arriver.

Afin de palier à ces inconvénients, notre intérêt s'est porté sur une autre métaheuristique basée population, qui partage des points communs avec l'algorithme génétique, mais le surpasse en d'autres : c'est la métaheuristique Particle Swarm Optimization (PSO). Ce sera l'objet de notre première contribution dans cette thèse, et sera détaillée dans le chapitre 5.

D'autre part, et malgré le succès des différentes métaheuristiques (AG, PSO, ACO,...etc), il est devenu évident que focaliser sur ces algorithmes seuls est plutôt restrictif, surtout quand il

s'agit de traiter certains problèmes d'optimisation, tels que les problèmes d'optimisation combinatoire, et les problèmes d'optimisation à large échelle. Une combinaison judicieuse d'une métaheuristique avec une autre métaheuristique, ou avec d'autres algorithmes d'optimisation (recherche opérationnelle), intelligence artificielle (analyse de contraintes), ou avec d'autres domaines de recherche récents (informatique quantique) peuvent aboutir à de meilleures solutions pour ces problèmes d'optimisation.
La motivation essentielle de ces combinaisons est d'obtenir un système plus performant, qui inclut et exploite les avantages respectifs et les points forts de ces techniques, tout en minimisant leurs inconvénients.

Parmi ces nouveaux domaines qui ont connu un intérêt grandissant, il y'a l'informatique quantique qui est le résultat d'un rapprochement multidisciplinaire incluant la physique, la chimie, l'informatique et les mathématiques. Cette révolution qui s'inspire des principes de la physique quantique permettrait de diminuer remarquablement la complexité algorithmique grâce notamment au calcul parallèle. Une telle possibilité de parallélisme peut être exploitée pour les problèmes d'optimisation qui manipulent une grande quantité d'informations.
L'informatique quantique permet donc de concevoir des algorithmes parallèles plus puissants. Cependant, ces algorithmes ne sont réalisables que sur des machines quantiques dont la construction est toujours en état de recherche. En attendant la construction de telles machines, l'idée de simuler les algorithmes quantiques sur des ordinateurs classiques ou de les combiner à des techniques conventionnelles est apparue.

Les recherches dans le domaine de l'hybridation des modèles bio-inspirés et de l'informatique quantique ont commencé à la fin des années 1990. Le but principal de cette hybridation est l'amélioration du rendement de chacun de ces deux paradigmes en s'inspirant mutuellement l'un de l'autre. Pour atteindre ce but, deux axes de recherche ont vu le jour : Le premier vise la génération de nouveaux algorithmes quantiques en utilisant les techniques de programmation automatique (programmation génétique par exemple), ceci est motivé par la difficulté de concevoir de nouveaux algorithmes quantiques. L'autre se base sur le calcul classique inspiré des fondements quantiques et destiné à être exécuté sur des ordinateurs classiques. C'est une branche d'étude dans le calcul conventionnel où on a introduit certains principes de la mécanique quantique, tels que : l'interférence et la superposition d'états.

Dans ce cadre, on s'est intéressé à l'hybridation des métaheuristiques PSO et AG avec l'informatique quantique. Ceci s'est soldé par deux autres contributions, détaillées en chapitre 5 et chapitre 6.

# CONTRIBUTIONS

1)Dans une première contribution, nous avons utilisé la métaheuristique PSO au lieu de l'algorithme génétique (GA). En effet, PSO présente plusieurs avantages par rapport à l'algorithme génétique :
• PSO est très facile à implémenter, et présente peu de paramètres à ajuster.
• PSO est relativement plus rapide que GA, et présente un bon compromis entre vitesse et efficacité.
• PSO ne nécessite pas des opérateurs pour évoluer, comme la mutation et le croisement, car les solutions potentielles émergent simplement de l'espace de recherche du problème, se mouvant itérativement dans la direction de la solution optimale courante.
• PSO converge plus rapidement, et fournit de meilleures solutions que GA.

Dans ce contexte, nous avons exploré les capacités de PSO pour faire évoluer un automate cellulaire pour la détection de contours sur divers types d'images. L'automate cellulaire est le système complexe qui modélise une image. Il évolue par des règles de transition simples, partant d'une configuration initiale qui est l'image à traiter, et aboutissant à la configuration finale qui est l'image segmentée en contours. Le processus évolutionnaire guidé par PSO a pour but d'extraire le sous-ensemble de règles capables d'aboutir à l'image finale désirée. L'algorithme développé s'est avéré efficace, et a abouti à de très bons résultats. Les expérimentations sont encourageantes et démontrent la faisabilité, la convergence et la robustesse de l'algorithme PSO pour la résolution de l'émergence inversée.
Ce travail a fait l'objet des travaux scientifiques suivants :
• **Article** : "Combining Cellular Automata and Particle Swarm Optimization for Edge Detection", Safia Djemame, Mohamed Batouche, International Journal of Computer Applications (ISSN 0975-8887), volume 57-N° 14, pp 16-22, novembre 2012.
•**Colloque** : "Une Approche de Détection de Contours Basée sur la Métaheuristique OEP" , Djemame Safia, Mohamed Batouche, $11^{eme}$ Colloque Africain sur la recherche en Informatique CARI 2012, 13-16 octobre 2012, Alger, Algérie.

2) Dans une deuxième contribution, nous avons exploré la combinaison de la métaheuristique GA avec l'informatique quantique, dans le but de développer un algorithme puissant, ayant la capacité de résoudre le problème de l'émergence inversée. On a utilisé l'algorithme génétique quantique pour faire évoluer un automate cellulaire qui a pour tâche d'extraire des contours sur des images. L'algorithme QGA s'est avéré performant pour extraire les règles aboutissant à la tâche finale désirée.
Cette contribution a fait l'objet des productions scientifiques suivantes :
• **Book chapter** : " A Hybrid Metaheuristic Algorithm Based on Quantum Genetic Compu-

ting for Image Segmentation", book chapter in "Hybrid Metaheuristics for Image Analysis", Springer, Juillet 2018.
•**Colloque** : " Hybridation d'un Algorithme Génétique Quantique et d'un Système Complexe pour le Traitement d'Images ", Safia Djemame, Mohamed Batouche, COSI 2016 , $13^{eme}$ Colloque sur l'Optimisation et Systèmes d'Information 30 Mai - 01 Juin, 2016 Sétif, Algérie.

3) La troisième contribution a consisté à explorer les possibilités de l'algorithme PSO quantique pour trouver des solutions au problème de l'émergence inversée. Son application reste toujours en traitement d'images. Le processus évolutionnaire guidé par l'algorithme QPSO extrait les bonnes règles capables de réaliser une bonne détection de contours et un bon filtrage d'images bruitées. Cette contribution a fait l'objet d'un article de revue dans le journal "Soft Computing", Springer.
•**Article** : "Solving Reverse Emergence with Quantum PSO - Application to Image Processing", Safia Djemame, Mohamed Batouche, Hamouche Oulhadj, Patrick Siarry, in Soft Computing journal, Springer, juin 2018.

# PLAN DE LA THESE

Le travail présenté dans cette thèse est organisé en six chapitres.
Le chapitre 1 est consacré aux systèmes complexes et les notions qui leur sont inhérentes telles que : l'émergence, l'auto-organisation, l'adaptation, l'apprentissage...etc. Nous donnons un aperçu sur les différents domaines d'application des systèmes complexes (Biologie, Economie, Chimie,...etc). Des exemples de systèmes complexes naturels et artificiels sont explicités. Du tas de sable aux boids de Reynolds, en passant par les automates cellulaires et les réseaux de neurones, la richesse et l'étendue des systèmes complexes est mise en évidence.

Le chapitre 2 présente les automates cellulaires, le système complexe sur lequel se porte notre intérêt dans cette thèse. Nous expliquons ses principes de fonctionnement, nous donnons ses différents types et classifications. Nous montrons les différents domaines d'application des automates cellulaires (Physique, Electronique, Mathématiques, Codage ...etc). Nous discutons l'usage des automates cellulaires en traitement d'images, et exposons l'état de l'art des travaux de la littérature ayant utilisé les automates cellulaires pour différentes tâches de traitement d'images. Nous abordons ensuite le point crucial qui est la problématique posée par les AC, et qui motive ce travail. Nous définissons le problème inverse, l'émergence inversée, et les différents problèmes inverses d'automates cellulaires.

Dans le chapitre 3, nous exposons un état de l'art détaillé sur les métaheuristiques d'optimisation. Nous définissons heuristique et métaheuristique, et nous séparons métaheuristiques à solution unique (recuit simulé..etc), de métaheuristiques à population de solutions (colonies de fourmis, systèmes immunitaires artificiels..etc). La dernière partie de ce chapitre expose en détail la métaheuristique sur laquelle porte une de nos contributions : le PSO.

Le chapitre 4 introduit l'informatique quantique comme domaine émergent en informatique. Il présente ses principes de base, ses avantages et ses limites. Il expose également des algorithmes inspirés du quantique tout en présentant trois modèles qui ont un lien fort avec nos contributions : les algorithmes évolutionnaires quantiques, les algorithmes génétiques quantiques et les boids de Reynolds quantiques. Nous terminons ce chapitre par une synthèse des avantages des algorithmes inspirés du quantique.

Les chapitres 5 et 6 sont consacrés à nos propres contributions.
Dans le chapitre 5, la première partie décrit l'utilisation de la métaheuristique PSO pour la résolution de l'émergence inversée, nous expliquons le format des règles adopté, les fonctions de fitness utilisées. Ensuite, l'application du modèle proposé pour résoudre le problème de la détection de contours est détaillée, et les résultats obtenus exposés. Dans la deuxième partie, nous détaillons l'utilisation du PSO quantique pour résoudre le problème de l'émergence inversée, et son application en traitement d'images, pour la détection de contours et le filtrage de bruit.

Dans le chapitre 6, nous exposons notre troisième contribution : l'utilisation de l'algorithme génétique quantique pour la résolution du problème de l'émergence inversée et son application au problème de détection de contours sur images.

Nous terminons par une conclusion générale et des perspectives d'avenir pour enrichir ce travail.

Chapitre 1

# Introduction aux Systèmes Complexes

## 1.1 Introduction

Les systèmes complexes sont omniprésents en physique, économie, sociologie, biologie, informatique et plusieurs autres domaines scientifiques. Un système complexe est composé de petits composants agrégés, leur interaction et leur interconnectivité ne sont pas triviales, les interactions peuvent être de grande dimension, et non-linéaire, la connectivité peut exhiber des caractéristiques topologiques non triviales, ceci conduit aux propriétés émergentes du système non « anticipées » par ses composants isolés. De plus, quand le comportement du système est étudié d'après une perspective temporelle, des patterns self-organisés apparaissent. L'étude des systèmes complexes nécessite des stratégies composites qui emploient des algorithmes variés pour résoudre un seul problème difficile. Les composants de telles stratégies peuvent résoudre des phases consécutives amenant au but principal, peuvent être utilisés pour approcher des sous-tâches à partir de différentes perspectives, ou peuvent résoudre le problème principal de plusieurs manières qui sont agrégées pour former la solution finale (hyper-heuristiques, Generalized Island Models).

## 1.2 Systèmes Complexes

### 1.2.1 Introduction

Un système complexe est un système composé d'un grand nombre d'éléments simples, qui affichent un haut niveau d'interactivité. La nature de ses interactions est principalement non-

linéaire, du fait de l'existence de boucles de rétroaction (feedback loops), qui empêchent toute association (affiliation) entre les causes et les effets [RCL01].
Malgré leur nature non linéaire, les règles d'interaction locales (niveau microscopique) sont simples et prédictibles. Mais le nombre élevé des composants, ainsi que le bouclage des sorties sur les entrées, provoque un comportement complexe au niveau macroscopique.
Le comportement d'un Système Complexe (SC) ne peut être réduit à la composition linéaire des comportements de ses éléments constitutifs. On dit que "le tout est plus que la somme de ses parties".
En résumé, un SC exhibe un comportement au niveau macroscopique, qui ne peut être expliqué par la simple composition des comportements de ses parties au niveau microscopique.
Exemple : Le système nerveux central des animaux est un SC qui produit un comportement de haut niveau, qui ne peut être réduit à la somme des interactions entre les neurones.
Les composants d'un SC peuvent être eux-mêmes des sous systèmes complexes, ce qui n'enlève en rien la propriété de simplicité des composants d'un SC, mais conduit à une sorte de hiérarchisation des composants d'un SC (stratification, hiérarchisation organisationnelle).

### 1.2.2 Niveaux de Complexité

On peut distinguer trois niveaux de complexité, selon la nature des interactions entre les éléments du système (Fig.1.1).


FIGURE 1.1 – Types d'interactions entre les éléments d'un système

Or, il apparaît assez rapidement que la situation (3) n'est pas si complexe que cela : elle renferme en réalité des symétries qui permettent de l'appréhender facilement. Ceci n'est pas le cas de la seconde situation, pour laquelle il faut dresser la liste de toutes les interactions autorisées l'une après l'autre.

## 1.3 Historique et Définitions

### 1.3.1 Historique

Historiquement, le domaine des systèmes complexes a évolué selon les trois phases suivantes :

**1. Des années 40 aux années 60** :
Apparition de la science de la complexité grâce aux initiatives de plusieurs chercheurs provenant de disciplines variées. Ceci était le résultat de l'insatisfaction de plusieurs chercheurs des outils offerts par la science réductionniste pour la modélisation des phénomènes rencontrés dans leurs domaines de recherche.
Les premières initiatives ont été les travaux du cybernéticien Ashby sur les propriétés du cerveau humain, et des informaticiens Turing et Kolmogorov pour la conception d'une machine universelle, ainsi que les travaux de Von Neumann et Hofstadter portant sur la vie et l'intelligence artificielle [Tur50] ; [AP57].

**2. Au cours des années 70 et 80** :
Les mathématiciens et les physiciens du non-linéaire et de la mécanique statistique se sont intéressés aux phénomènes d'auto-organisation en généralisant aux systèmes biologiques l'idée que les propriétés globales émergentes de ce type de systèmes ne pouvaient être obtenues en effectuant une moyenne des propriétés des éléments en interaction, comme le font les méthodes classiques de la physique [HW03].

**3. A partir des années 90** :
Durant cette période, l'apparition des approches bioinspirées a fortement contribué dans la compréhension et la mise en ?uvre des systèmes complexes. Parmi ces approches, on cite : les systèmes multi-agents, les approches basées essaims et les systèmes immunitaires artificiels [Lan90] ; [HW03].

### 1.3.2 Définitions

Dans ce qui suit, on aborde deux définitions de base pour les systèmes complexes.

**Définition 1.1 (Système complexe)** *Un système complexe peut être vu comme un système dynamique composé de plusieurs systèmes simples, et de parties interagissant d'une manière non linéaire* [Ila04].
Dans cette définition, Ilachinski ne prend pas en compte une caractéristique clé des systèmes complexes : c'est la différence de comportement existante entre le bas niveau (composants et parties) et le haut niveau du système (le système lui même). Cette différence est bien cernée dans la définition suivante.

**Définition 1.2 (Système complexe)** *Un système complexe est un système composé d'un ensemble d'éléments homogènes ou hétérogènes, interagissant entre eux de façon non linéaire (interactions rétroactives), mettant ainsi en oeuvre une dynamique permettant à l'ensemble du système d'exister comme un tout, différent de la simple somme de ses composants* [Has03].

On voit bien que cette définition étend la définition 1.1 par la mise en relief de la possibilité que les éléments du système peuvent être hétérogènes, et par le fait que le bas niveau (niveau micro) et le haut niveau (niveau macro) sont différents l'un de l'autre. Ici, on doit noter que l'existence de différence entre les niveaux macro et micro n'est pas obligatoire, et même si elle existe, des propriétés clé des composants sont toujours présentes au niveau global. Un bon exemple de l'existence de similitude entre le niveau micro et le niveau macro est le système social : des caractéristiques clé des composants (soit humains, entreprises, etc) dans un système social sont toujours inhérentes au système lui même.
Dans un système complexe, on distingue deux niveaux : un niveau micro, représentant le niveau des composants, avec des propriétés locales à chacun d'eux ; et un niveau macro, représentant l'ensemble du système, avec des propriétés nouvelles, que l'on ne retrouve dans aucun des composants pris individuellement. On parle alors d'émergence de nouvelles propriétés. Le caractère complexe d'un système tient aux émergences induites par les interactions rétroactives.

## 1.4 Classification des Systèmes Complexes : types de complexité

[DD00] distinguent quatre types de base de la complexité :

### 1.4.1 Type 1. Complexité statique

C'est l'ensemble des aspects statiques de la complexité du système. Ceci inclut les notions de hiérarchie, la connectivité, les détails, la complication, la variété et les niveaux/forces des interactions. Ce type de complexité est facilement visualisé comme un réseau de patterns complexes de liens et de noeuds. La complexité statique est à un certain degré dépendante du contexte car elle va apparaitre très différemment aux niveaux micro et macro, et change quand les frontières du système changent. Exemple de hiérarchie dans un système réel : particules, électrons, atomes, molécules, acides aminés, protéines, cellules, organismes ...etc.

### 1.4.2 Type 2. Complexité dynamique

Ce type de complexité englobe les idées de complexité liées au comportement, aux processus de causes et effets, aux feedbacks, aux variations et stabilité, aux cycles et aux échelles de temps. Essentiellement, la théorie des SC se base sur l'étude de cette notion de changement de comportement au fil du temps qui est liée à un aspect important : *la complexité évolutionnaire*. Cette évolution de la complexité dynamique peut résulter de l'adaptation du système et de ses agents. La complexité d'un système peut évoluer sans l'existence d'une adaptation au niveau de ses agents. Exemple : orbites des planètes, battements de coeur, cycle des saisons, phases de développement du foetus...etc.

### 1.4.3 Type 3. Complexité évolutive

Qualifie la capacité de changement et d'adaptation des systèmes, caractérisés par l'attracteur chaotique (ergodique). Ces systèmes effectuent une recherche dans l'espace des états. Plus encore, ils étendent l'espace de recherche en ajoutant de nouvelles possibilités (création de nouveaux composants dans le système). Inversement ils peuvent réduire l'espace de recherche en éliminant les cas de défaillance. Exemple : sélection naturelle, système immunitaire...etc.

### 1.4.4 Type 4. Complexité auto-organisatrice

C'est la forme la plus intéressante de complexité. Elle combine les contraintes internes des systèmes clos et l'évolution créative des systèmes ouverts. Le système est en co-évolution avec son environnement. Toute la question est de comprendre comment les composantes du système s'auto-organisent pour atteindre un objectif de haut niveau, dans le cadre des contraintes

environnementales. Ces systèmes opèrent aux frontières du chaos. Ils sont caractérisés par des boucles de rétroaction, qui leur donnent une dynamique non-linéaire. En combinant les trois types d'attracteurs, ils ont une structure plus riche et plus complexe. Ce type de systèmes diffère de la catégorie des systèmes purement évolutifs, par le fait que, l'espace de recherche est canalisé par la nature auto-organisatrice (causalité descendante : le macro-niveau influence le micro-niveau), de leurs processus internes émergents. Ainsi le système autolimite ses possibilités. Ces systèmes se trouvent dans des situations dissipatives, semi-stables, loin-de l'équilibre, qui exhibent la distribution en loi puissance des événements, familière aux systèmes critiques en transition de phase. Ils sont structurellement à la fois ouverts et clos, avec des frontières semi-perméables. Exemples : colonies de fourmis, termites bâtisseuses, SMA...etc

## 1.5 Caractéristiques des Systèmes Complexes

Malgré la grande diversité de SCs existants dans la nature, on distingue des règles récurrentes (universelles) qui apparaissent dans la plupart d'entre eux. Nous présentons dans la suite, de manière concise, les propriétés les plus courantes qu'on attribue aux SCs. Tous les SCs ne sont pas censés englober obligatoirement la totalité de ces propriétés :

### 1.5.1 Complexité

Il n'existe pas à ce jour une définition unique et acceptée par tous de la notion de complexité. Ceci n'est pas seulement dû à la jeunesse du domaine des systèmes complexes, mais aussi à sa pluridisciplinarité. Chaque domaine scientifique étudie les systèmes complexes avec ses propres outils et prérogatives de sorte que des définitions propres à chaque discipline ont été formulées. Cependant, on trouve la définition suivante qui synthétise à un certain degré quelques définitions connues dans la littérature de la notion de complexité.

**Définition 1.3 (Complexité)** : La complexité est la propriété d'un modèle qui rend difficile la formulation de son comportement global même s'il y a assez d'information pour décrire ses parties et les relations entre elles [Edm00].
La complexité dans un SC, est dûe aux interactions locales. Elle résulte principalement de la diversité des agents du système et du grand nombre de connexions entre ces agents. Selon [Mor08] : "Penser en termes de complexité signifie reconnaître le Principe Holistique selon lequel l'unité de l'organisme n'est pas réductible à la somme de ses parties constitutives". Il propose la formulation suivante pour la description d'un système complexe :

$$Système = \sum parties + \sum relations\ entre\ parties$$

[Moi99] fait la distinction entre systèmes complexes et systèmes compliqués. Ils disent que les systèmes compliqués sont des systèmes que l'on peut réduire en éléments plus simples que l'on peut analyser séparément pour comprendre le système global, par exemple un avion. Alors que dans le cas des systèmes complexes, la somme des éléments fait émerger de nouvelles propriétés qui ne sont pas présentes dans les éléments composant le système, par exemple le système immunitaire.

### 1.5.2 Changement du niveau d'intégration

Lorsque l'on étudie les SCs, on remarque une sorte de mutation du comportement observé lors du changement d'échelle d'intégration (du niveau microscopique vers le niveau macroscopique) : par exemple le fonctionnement des cellules et des constituants biochimiques du cerveau, ne permet en aucun cas d'augurer des phénomènes tels que la mémoire, l'intelligence ou la conscience. Il est clair, que lorsque l'on change de niveau de description, il se produit des phénomènes de réorganisation et de simplification. Par exemple, l'observation des états de la matière, de l'infiniment petit vers l'infiniment grand, nous fait apercevoir de nombreuses simplifications : pour savoir comment va évoluer une étoile il ne faut pas étudier les combinaisons de chacun des quarks qui constituent chacun des noyaux de ses atomes, il suffit, en gros, de connaître sa masse totale. Ainsi, selon que l'on considère une échelle atomique ou cellulaire, ou planétaire, des lois nouvelles émergent qui permettent de ne pas tenir compte des éléments individuels du système de l'échelle immédiatement inférieure.

### 1.5.3 Boucles de rétroaction et non-linéarité

Lorsque le système considéré est linéaire, on peut effectuer des calculs sur la stabilité, la dynamique...etc. Mais lorsque le système n'est pas linéaire tous les comportements possibles sont envisageables.
Parfois les bouclages interviennent de manière indirecte, et n'apparaissent pas dans le système lui même : c'est le cas lorsque la sortie d'un système influence des variables diverses qui, à leur tour, influencent l'entrée. C'est le cas de tous les systèmes écologiques, mais aussi de nombreux systèmes sociaux ou humains (ex : la bourse).

**Non linéarité** : "Un SC est vu comme un système non linéaire, dont le résultat des interactions entre composants excède les contributions individuelles de chaque composant"

[MAR97].
"Les propriétés du système ne sont pas une composition linéaire des propriétés de ses composants. Les interactions épistatiques entre les parties du système, imposent une étude globale et non-réductionniste (analytique) de celui-ci" [Luc02].

### 1.5.4 Indéterminisme

Un SC qui fait intervenir un grand nombre de paramètres interdépendants, ne peut être réduit à un ensemble d'équations simples prévoyant l'état futur du système à partir de son état actuel.

Poincaré en 1908 écrivait dans son livre : "Science et méthodes" : *"Si nous connaissions exactement les lois de la nature et la situation de l'univers à l'instant initial, nous pourrions prédire exactement la situation de ce même univers à un instant ultérieur. Mais, alors même que les lois naturelles n'auraient plus de secret pour nous, nous ne pourrions connaître la situation initiale qu'approximativement. Si cela nous permet de prévoir la situation ultérieure avec la même approximation, c'est tout ce qu'il nous faut, nous disons que le phénomène a été prévu, qu'il est régi par des lois ; mais il n'en est pas toujours ainsi, il peut arriver que de petites différences dans les conditions initiales en engendrent de très grandes dans les phénomènes finaux ; une petite erreur sur les premières produirait une erreur énorme sur les derniers. La prédiction devient impossible et nous avons le phénomène fortuit."*

Ce phénomène est connu sous le nom de *"chaos déterministe"*. De par la très grande sensibilité des SCs aux conditions initiales, il n'est pas possible de déterminer avec exactitude leur comportement futur car les conditions initiales ne sont jamais connues avec une précision suffisante.
On l'appelle *chaos déterministe* car si on refait une expérience avec exactement les mêmes conditions initiales, on observe exactement le même comportement. Néanmoins, dans la réalité il est impossible de refaire une expérience identique, de même qu'il est impossible de connaître toutes les entrées d'un système et toutes les conditions initiales. Le chaos déterministe est essentiellement causé par le retour des sorties du système sur ses entrées. Ce phénomène appelé Feed-Back (rétroaction) produit une forte nonlinéarité et conduit à un équilibre dynamique (semi-stable et fragile).

**Indéterminisme** *"Les boucles de rétroaction induisent des divergences de comportement du système, à partir de données d'entrée très similaires. En plus des convergences peuvent êtres*

*observées pour des entrées sensiblement distinctes. Ceci est dû au fait que le système est dans un état proche du chaos, caractérisé par un mélange d'attracteurs"* [Luc02].

**Instabilité** *"Des changements soudains d'attracteur sont possibles, lorsque le système approche des frontières de l'un d'eux. L'évolution a lieu par bonds plutôt que graduellement"* [Luc02].

### 1.5.5 Irréversibilité

Les SCs sont irréversibles et dépendent de leur passé. De ce fait ils rompent avec la tradition des systèmes physiques classiques pour lesquels les équations sont symétriques par rapport au temps ([PS96].
Au XIXème siècle le débat était vigoureux : comment expliquer la diversité du vivant avec les lois de la physique si déterministes ? Les scientifiques ont oscillé entre deux attitudes opposées. La première considère que la diversité vient d'un épiphénomène se rajoutant aux phénomènes physiques bien expliqués. La seconde considère que ces phénomènes sont bien réels, mais proviennent d'un supplément ontologique . Aujourd'hui, on considère la complexité comme un phénomène à part entière.
L'irréversibilité des systèmes physiques prend sa justification dans le second principe de la thermodynamique : l'entropie d'un système ne peut que rester stable ou augmenter. Ainsi, si l'entropie augmente, le système ne pourra jamais, s'il reste isolé, revenir à son état initial.
Considérons par exemple l'univers, il est actuellement en expansion. Si l'on adhère à l'hypothèse qu'il sera un jour dans une nouvelle contraction, il est clair que cette contraction ne sera pas l'image inversée de l'expansion qui a précédé du fait de l'irréversibilité. "*Les particules qui ont laissé leur traces dans les papiers photosensibles ne vont pas effacer leurs traces, les êtres vivants ne vont pas régurgiter ce qu'ils ont mangé au fur et à mesure qu'ils vont rajeunir*" [Luc02].
Bref, la flèche du temps ne va pas s'inverser, alors que nombre de modèles de la physique sont symétriques par rapport au temps. Ces modèles sont en général linéaires alors que les SCs ne le sont pas. Les SCs ne sont pas réversibles, et en particulier les systèmes complexes biologiques.

### 1.5.6 Emergence

Comme le résument Ali et Zimmer [1997] dans leur article *"The Question Concerning Emergence"* [AZ97], l'origine de l'émergence pourrait bien être le postulat datant de la Grèce antique : "*le tout est plus que la somme de ses parties*".
Dès lors, de nombreux philosophes, mathématiciens, biologistes et physiciens participent à ce

courant de pensée. Au milieu du XIXe siècle un nouveau courant est apparu autour du terme de l'émergence [MV80]. Ce nouveau courant a été appelé plus tard le Proto-émergentisme.

En 1875, le philosophe anglais *G. H. Lewis* fait la distinction entre *résultant* et *émergent*. Il explique que pour le résultant la séquence d'étapes qui produisent un phénomène est traçable, alors que pour *l'émergent* nous ne pouvons pas tracer les étapes du processus. Ainsi, nous ne pouvons pas voir dans le produit le mode d'opération de chaque facteur. L'émergence ne peut pas être réduite ni à la somme ni à la différence des forces coopérantes. Les proto-émergentistes cherchaient surtout à définir l'émergence afin de reconnaître un phénomène émergent et le différencier de phénomènes explicables grâce à d'autres théories ou modèles [GEO03]. Dans ce qui suit, on donne quelques définitions formelles de l'émergence suivies d'une brève explication de leurs contenus.

**Définition 1.4 (Émergence)** *La propriété d'émergence réfère à l'apparition inattendue de patterns spatiaux et temporels dans la structure et la dynamique du système [Par02]. Ceci est équivalent au fait que le comportement du système n'est pas déduit des comportements de ses parties.*

**Définition 1.5 (Émergence)** *La notion d'émergence peut être définie d'une manière intuitive comme une propriété macroscopique d'un système qui ne peut pas être inférée à partir de son fonctionnement microscopique [Gle04].*

**Définition 1.6 (Émergence)** *Le principe de l'émergence correspond à l'apparition d'un tout cohérent à partir d'interactions locales d'agents non explicitement informés de la manière d'obtenir ce tout cohérent [Gle04]*, comme illustré en figure 1.2.

En effet, le comportement global d'un système complexe ne peut être compris par la simple somme des comportements individuels des entités qui le composent. Il est impossible de comprendre les systèmes complexes sans penser que les agents simples d'une manière ou d'une autre vont donner des comportements complexes. Donc, dans l'étude de l'émergence dans les SCs, nous devons distinguer l'émergence locale de l'émergence globale. La première concerne une partie du système et elle est simple à analyser. Cependant la deuxième concerne les propriétés observées seulement au niveau du système global [GEO03].

•Types d'émergence : [Fro05] identifie quatre types de base de l'émergence. Selon lui, Dans le premier type, aucun 'feedback' n'existe au niveau du système, il n y'a que des 'feedforwards'. La propriété majeure de l'émergence de deuxième type est l'existence de feedbacks simples, soit positifs ou négatifs. Les feedbacks multiples, l'apprentissage et l'adaptation sont

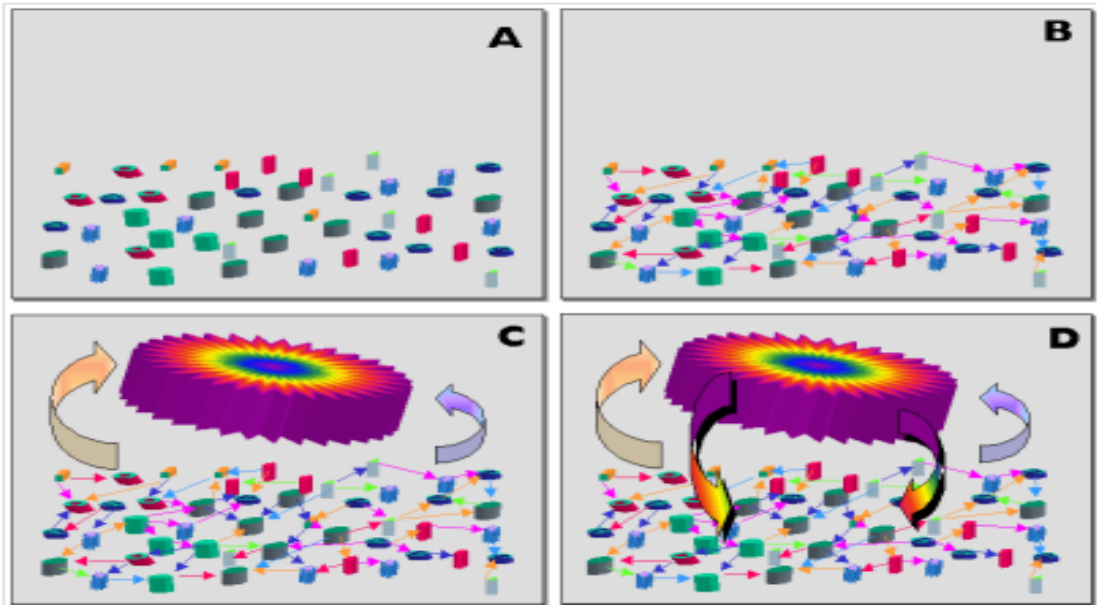


FIGURE 1.2 – L'émergence. a)Un système d'agents divers b)richement connecté c)fait apparaitre un pattern émergent d)qui renvoie un feed-back au système

des points clé dans le troisième type de l'émergence. On trouve ce type d'émergence dans les SCs intelligents ayant une interaction intensive avec leurs environnements externes. Finalement, le quatrième type d'émergence est caractérisé par le multi-niveau et la diversité des patterns créés (figure1.3).

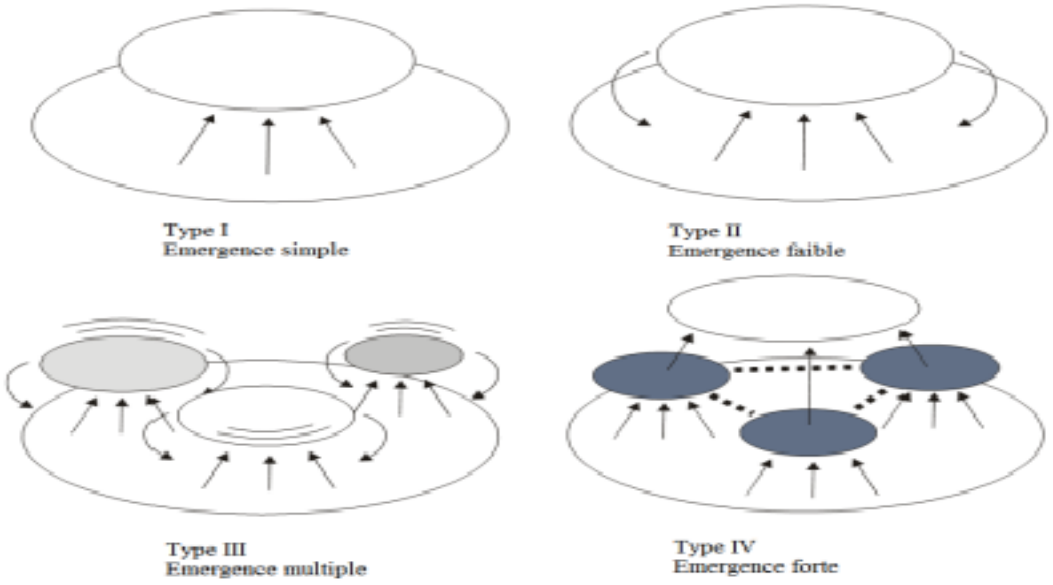


FIGURE 1.3 – Types d'émergence. [Fro05]

### 1.5.7 Auto-organisation et systèmes auto-organisés

Selon [Par02], l'auto-organisation est le mécanisme responsable de l'émergence. C'est le processus par lequel l'effet collectif des interactions locales entre les entités du système, bien qu'apparemment désorganisé, forme une structure et un comportement ordonné émanant au niveau global.

L'auto-organisation peut aussi être vue comme une collaboration entre les entités du système qui modifient leur structure interne dans le but d'améliorer la viabilité et l'efficacité des relations que ce dernier entretient avec son environnement [Man01].

Il n'existe pas une hiérarchie de commandes et de contrôle dans un système complexe adaptatif. Donc, il n'y a pas de planning du comportement global et pas de gestion des tâches. Mais, il existe une auto-organisation continue agissant pour donner le meilleur rendement par rapport aux changements de l'environnement. Le système s'auto-organise par les processus d'émergence et de feedback.

•**Différence entre émergence et auto-organisation** :
Selon [Fro06], les termes 'émergence' et 'auto-organisation' sont utilisés des fois d'une façon interchangeable, mais la différence entre les deux concepts peut être cernée comme suit. Le terme auto-organisation réfère plus à un processus de dynamique résidant au frontières entre le système et son environnement. Tandis que le terme 'émergence' est concerné par les frontières entre le niveau micro et le niveau macro du système. En pratique, les deux phénomènes se trouvent ensemble comme caractéristiques de plusieurs systèmes complexes naturels.

### 1.5.8 Adaptation

Le terme 'adaptation' est défini différemment selon le domaine considéré. Dans ce qui suit, on adopte deux définitions extraites respectivement des sciences de la nature et des sciences sociales. Ce choix est basé sur notre intérêt dans le cadre de cette thèse au systèmes inspirés de systèmes sociaux naturels.

**Définition 1.7 (Adaptation)** L'adaptation dans les systèmes naturels réfère au développement des caractéristiques génétiques et comportementales qui permettent aux organismes ou systèmes de faire face aux changements de l'environnement afin de survivre et se reproduire [SW06].

**Définition 1.8 (Adaptation)** Le processus d'adaptation est "un processus par lequel des groupes d'êtres humains ajoutent des méthodes nouvelles et améliorées pour faire face à l'environnement de leur répertoire culturel" [OH92].

On distingue principalement deux types d'adaptation : l'adaptation spontanée et l'adaptation planifiée. Dans la première, le système s'adapte d'une manière immédiate et non réfléchie d'un point de vue stratégique. Par contre, l'adaptation planifiée résulte de décisions stratégiques intentionnelles, fondées sur une perception claire des conditions qui ont changé - ou qui sont sur le point de changer - et sur les mesures qu'il convient de prendre pour revenir, s'en tenir ou parvenir à la situation souhaitée.

#### 1.5.8.1 Système adaptatif

Selon Grisogono [Gri10], pour qu'un système soit adaptatif, il doit satisfaire les conditions suivantes :

1. l'existence de la notion de succès (techniquement connu sous le nom de 'fitness') et d'échec pour le système dans son environnement.

2. l'existence d'une source de variation dans les détails du système.

3. l'existence d'un processus de sélection au niveau des variations offertes pour l'amélioration de la fitness du système. Ceci nécessite :

4. l'existence d'un outil d'évaluation de l'impact d'une variation sur la fitness du système qui est généralement réalisée à travers un cycle d'interaction et feedback. Tout le temps, le système génère et intériorise des variations qui tendent à augmenter sa fitness. Ce processus exploite la diversité, la variabilité, et la sélection basée succès, sous l'influence inhérente des changements de l'environnement affectant le système et son environnement. Pour l'utilisation pratique de la notion d'adaptation dans le cadre de l'ingénierie des systèmes complexes, on doit bien savoir choisir les décisions à prendre dans les premiers stades et celles à laisser ouvertes pour une sélection adaptative réalisée par le système en cours d'exécution (quand plus d'information sera disponible sur l'environnement du système).

### 1.5.9 Apprentissage

L'apprentissage est une notion très liée au domaine de la psychologie. Donc, il est clair que dans ce domaine on peut trouver les meilleures définitions de ce concept. Dans ce qui suit, on donne deux définitions de l'apprentissage et ses types, et on présente la différence entre L'apprentissage et l'adaptation.

**Définition 1.9 (Apprentissage)** L'apprentissage est le processus continu de transformation de l'information et de l'expérience en connaissances, compétences, comportements et attitudes [Cob11].

**Définition 1.10 (Apprentissage)** L'apprentissage est défini comme un changement relativement permanent dans le comportement comme un résultat de l'expérience [Lt11].

#### 1.5.9.1 Types d'apprentissage

On distingue principalement trois types d'apprentissage : l'apprentissage supervisé, l'apprentissage non supervisé et l'apprentissage par renforcement. Dans le premier type, un superviseur externe est présent. Ce dernier fournit les réponses désirées (exemples et contre-exemples) au système. Dans le deuxième type, non supervisé, l'apprentissage se fait sans guide, l'apprenant fait des hypothèses, puis les teste, et ensuite il les évalue selon certains critères. Le troisième type d'apprentissage, apprentissage par renforcement, est réalisé à travers une séquence de perceptions de l'environnement, d'actions du système et de récompenses comme feedback de l'environnement dans lequel le système évolue [CM02].

#### 1.5.9.2 Différence entre apprentissage et adaptation

A partir des définitions ci-dessus de l'apprentissage et de l'adaptation, on peut bien voir qu'il existe un chevauchement entre les deux concepts du point de vue but avec des différences au niveau façon et durabilité. En ce qui concerne la similitude, le système (ou l'agent) dans les deux cas s'adapte ou apprend pour pouvoir survivre ou être plus performant dans un environnement donné [ES07]. En ce qui concerne les différences, on peut citer les suivantes. L'apprentissage se fait au niveau global du système, ce qui le rend intelligent, mais l'adaptation s'applique seulement à une partie du système. Donc, un système intelligent apprend et un système adaptatif s'adapte. Aussi, une différence importante entre les deux concepts se voit dans le fait que l'apprentissage se fait avec une présence de la conscience alors que l'adaptation ne nécessite pas sa présence. Une autre différence est que l'apprentissage est relativement

durable, alors que l'adaptation ne l'est pas.

## 1.6 Domaines d'application des systèmes complexes

Les systèmes complexes offrent un squelette flexible qui permet d'analyser des problèmes interdisciplinaires. Ils utilisent des techniques générales issues de la physique, des mathématiques, et de l'informatique afin d'approcher des sujets en Biologie, Médecine, Sociologie, Psychologie, Neuroscience, Chimie, Economie, et Ingénieurie. En 2011, lors de "International Convention on Complex Systems", les présentations faites ont varié de la photosynthèse artificielle, à la formation du gouvernement de l'ancienne Egypte, aux réseaux régulateurs de gènes dans les embryons d'oursins...et bien d'autres [Woo11].

### 1.6.1 Biologie

La biologie est un domaine qui a hautement bénéficié des autres sciences, et y a en contrepartie, fortement contribué. Alors que les biologistes ont utilisé les autres disciplines pour réaliser des avancées significatives afin de comprendre des processus, organismes et écosystèmes biologiques compliqués, d'autres champs ont trouvé inspiration à partir des systèmes biologiques. Nous citons quelques uns de ces domaines :
•Bioinformatique : c'est un domaine où les chercheurs essaient de comprendre les processus biologiques, en appliquant les techniques de l'informatique et les théories de l'information sur la grande quantité de données disponibles en Biologie. Ces champs de recherche incluent :
o Développement pharmaceutique
o Evolution
o Mutations cancéreuses
o Génétique
•Systèmes biologiques : c'est une stricte application des techniques des systèmes complexes sur les systèmes biologiques, les chercheurs étudient les interactions entre les entités biologiques pour modéliser le comportement du système entier. Ces champs de recherche incluent :
o Dynamiques cellulaires
o Processus métaboliques
o Réseaux régulateurs de gènes
o Réseaux cancéreux
•Biomimétique : Biomimétique et Biomimétisme sont l'union entre l'ingénierie et la biologie, à travers lesquels les chercheurs étudient les propriétés physiques et structurelles des organismes biologiques, comme source d'inspiration pour les ingénieurs, notamment en informatique Soft

et Hard. Les champs de recherche incluent :
o Organes artificiels
o Robotique
o Dynamique des essaims
o Conception évolutionnaire
o Programmation évolutionnaire
o Intelligence artificielle

### 1.6.2 Sciences sociales

C'est un domaine qui connait un rapide développement grâce aux techniques inspirées des systèmes complexes, beaucoup de travaux de recherche s'intéressent au développement des organisations sociales et leur comportement.
•Analyse de réseaux sociaux : tous les organismes sociaux forment de manière inhérente des réseaux sociaux qui peuvent être analysés et révéler plusieurs aspects incluant : Les membres influents, susceptibilité aux épidémies et défaillance de systèmes, diffusion d'information.
•Anthropologie évolutionnaire : certains chercheurs s'intéressent à l'étude de la manière dont les civilisations et sociétés se forment, et aussi comment les organismes développent des habitudes et comportements sociaux. Ceci cause un « feedback » où le comportement de l'organisme aura nécessairement un impact sur la structure sociale, et inversement celle-ci aura un impact sur le comportement du système. Des chercheurs essaient d'appliquer les techniques des systèmes complexes pour comprendre la co-évolution de ces phénomènes, ceci inclut l'évolution des états, de la coopération, la robustesse des organisations sociales.

### 1.6.3 Psychologie et neuroscience

Le cerveau est souvent considéré comme un système complexe très étudié par la science moderne, il exhibe des phénomènes intéressants tels que la conscience, la compréhension sémantique, mémoire, prédictions, logique, créativité, apprentissage...etc. Des chercheurs s'attachent à comprendre ces caractéristiques comme des propriétés émergentes de la structure et interaction entre les différentes parties et cellules du cerveau.
•Psychologie computationnelle : des chercheurs de différents domaines s'intéressent à la manière dont le cerveau travaille, et comment il arrive à des conclusions. Des modèles mathématiques et des simulations informatiques basées sur des interactions neuronales ont été développées, et se montrent prometteurs pour l'apprentissage et reconnaissance des formes. D'autres chercheurs focalisent sur la compréhension des causes de la dégénérescence des fonctions cérébrales liées à l'âge, maladies mentales. Les domaines spécifiques de recherche incluent :

* Intelligence artificielle
* Génération d'idées créatives par computer
* Reconnaissance d'images
* Prédiction de données
* Linguistique computationnelle

Le développement et l'évolution des langages a certaines contraintes, comme la non-ambiguité, facilité de remémoration, facilité orale, compréhension facile. Les modèles computationnels de linguistique considèrent un langage comme un réseau reliant des mots, des idées et des règles de grammaire dans un réseau complexe. D'autres recherches s'orientent vers la compréhension des phonèmes du langage, les sons fondamentaux assemblés pour former des mots. Les axes de recherche sont :

* Processeurs de langage naturel
* Orateurs de langage naturel
* Traducteurs
* Linguistique Théorique

### 1.6.4 Economie

Le domaine de l'économie est composé de structures hautement complexes, développées et adaptatives. toute économie est basée sur des réseaux commerciaux multiples et dynamiquement interconnectés. Comprendre la manière dont de tels réseaux interagissent aiderait à comprendre comment optimiser la production et le développement durable. Des recherches spécifiques ciblent : Economie et développement durable, Distribution optimale des ressources, Théorie des jeux, et comment les choix économiques sont faits ?, Co-évolution à travers compétition et coopération.

### 1.6.5 Chimie

La chimie implique souvent l'étude d'interactions moléculaires afin de former de nouvelles molécules, et les propriétés qu'elles ont. Dans les réactions chimiques, plusieurs réactifs, catalyseurs et produits interagissent entre eux et avec leur environnement, aboutissant à un système dynamique. Des recherches spécifiques incluent :

◦ Vie artificielle
◦ Production industrielle de composants chimiques
◦ modélisation écologique

### 1.6.6 Géologie

Les scientifiques géologues tentent de comprendre comment la Terre évolue, comment elle est arrivée à l'état actuel, et ce qu'elle pourrait devenir dans le futur. Le système Terre est hautement complexe et concerne non seulement la planète Terre solide, mais aussi l'hydrosphère, l'atmosphère, la biosphère et la magnétosphère. Les géologues ont opté pour une vue réductionniste de ces systèmes, pour essayer de mieux cerner la dynamique de chacun d'eux. La recherche inclut :

- Comprendre comment les formes de vie ont influé sur la composition de l'atmosphère.
- Comprendre les changements climatiques
- Comprendre l'équilibre, distribution et flots d'énergie entre les sphères terrestres.

### 1.6.7 Logistique

Déterminer comment obtenir et allouer des ressources dans un problème commun, face à toutes les organisations et entreprises. De plus, trouver la solution optimale à tous ces problèmes peut rendre le système vulnérable aux perturbations au delà du contrôle de l'organisation. L'analyse de le robustesse et de l'optimalité de réseaux complexes interdépendants relève de l'analyse des systèmes complexes. La recherche inclut :

- Défaillances consécutives de réseaux
- Optimisation évolutionnaire de réseaux.

La figure 1.4 résume la majorité des domaines d'application des SCs.

## 1.7 Exemples de systèmes complexes naturels et artificiels

On présente ici une liste d'exemples de SCs naturels et artificiels, pour illustrer les définitions précédentes.


FIGURE 1.4 – Domaines d'application des systèmes complexes

### 1.7.1 Systèmes complexes naturels

#### 1.7.1.1 Le tas de sable

Le sable est considéré à mi-chemin entre l'état liquide et l'état solide (Fig.1.5). Si on retourne un seau de sable mouillé vigoureusement, le contenu se déverse en un coup, mais le tas formé reste ensuite immobile (solide), en revanche si on le retourne plus doucement, il s'effondre et s'écoule en vagues successives (liquide). Lorsqu'on applique une contrainte de

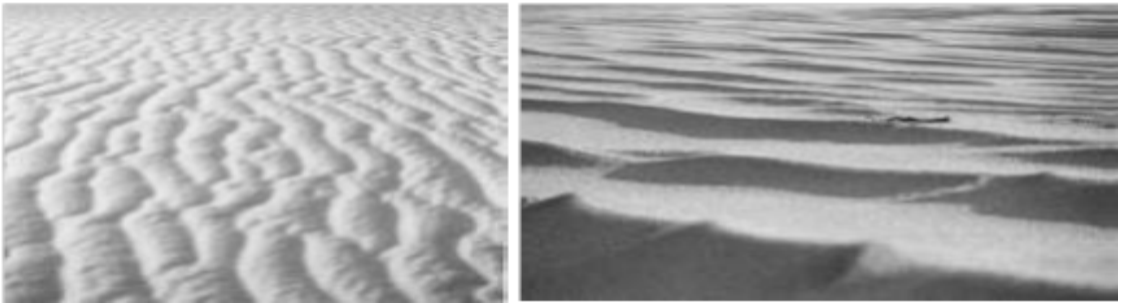

FIGURE 1.5 – Les ondulations de sable dans un état semi-solide

cisaillement, le sable cède à partir d'un certain seuil qui dépend de la profondeur sur laquelle est appliquée la contrainte dans le tas : le tas est donc de plus en plus solide en profondeur. Les écoulements dans le sable, par exemple dans un sablier, ne sont pas uniformes : il arrive qu'une sorte de voûte se construise et fasse obstacle à la progression des grains vers le sous tirage.

Un tas de sable est formé par de nombreux grains reposant les uns sur les autres ; c'est à dire en interactions. Ces interactions ne sont pas en général linéaires, à cause des frottements des grains de forme quelconques. On est donc bien dans la configuration d'un SC. Certains scientifiques étudient ces systèmes en effectuant des simulations numériques. Ils ont noté deux caractéristiques :

- De nombreuses voûtes se forment spontanément (structures dynamiques) et diminuent le poids apparent du sable en dessous d'elles mêmes,
- Ces voûtes sont en perpétuel réarrangement : il suffit qu'un grain glisse pour que la voûte se réarrange (aux frontières du chaos) d'une autre manière.

#### 1.7.1.2 Les colonies d'insectes sociaux

Les colonies d'insectes sociaux présentent des caractéristiques évidentes de complexité, elles en contiennent tous les "ingrédients" : grand nombre d'entités autonomes, nombreuses interactions, fonctionnement simple des entités, capacités cognitives limitées, système confronté à un environnement dynamique, fonctionnement de haut niveau remarquable. Voici deux exemples de SCs naturels : les colonies de fourmis et les termites bâtisseuses.

##### 1.7.1.2.1. Construction de nids chez les termites

Après une certaine phase de déplacement non-coordonnée, quelques termites récoltent de la terre, en façonnant des boulettes et déposent celles-ci de manière aléatoire sur le sol. Cette phase d'incoordination est suivie d'une autre, durant laquelle une coordination plus précise des dépôts entraîne progressivement la reconstruction d'une loge royale ayant sa forme spécifique [BT94].

##### 1.7.1.2.2 Fourragement chez les fourmis

Il a été montré que les fourmis étaient capables de sélectionner le plus court chemin pour aller du nid vers une source de nourriture [Bru00], grâce au dépôt et au suivi de pistes de phéromones. Lorsque des fourmis doivent emprunter un pont à deux branches de longueurs différentes pour exploiter une source de nourriture, elles sélectionnent automatiquement la branche la plus courte, si la différence entre les longueurs des branches est suffisamment importante. Au départ, le choix est aléatoire mais la branche courte devient vite la plus marquée, car les fourmis qui l'empruntent arrivent plus vite au nid et auront statistiquement plus de chance de l'emprunter lorsqu'elles retourneront vers la source de nourriture.

#### 1.7.1.3 Autres exemples naturels

De nombreux systèmes naturels composés d'individus autonomes exhibent des aptitudes à effectuer des tâches qualifiées de complexes, sans contrôle global. De plus, ils peuvent s'adapter à leur milieu, soit pour y survivre, soit pour améliorer le fonctionnement du collectif.

- *Déplacement collectif* : le déplacement d'oiseaux migrateurs ou de bancs de poissons exhibe le fait que la tâche collective est le résultat des interactions d'individus autonomes (Fig.1.6).
- *Système immunitaire* : ensemble de cellules dont le rôle est de défendre l'organisme contre les infections, lorsqu'il est envahi par un organisme étranger (virus, bactérie) ou par des molécules étrangères (pollen). Le système immunitaire réagit très vite pour s'en débarrasser. Le fonctionnement du système immunitaire est représentatif du fonctionnement d'un SC naturel.

FIGURE 1.6 – Déplacements collectifs d'oiseaux migrateurs et de bancs de poissons

### 1.7.2 Systèmes complexes artificiels

La vie artificielle fait jonction entre la biologie et le domaine informatique. Elle s'inspire aussi beaucoup de la physique, la chimie, l'économie, la philosophie...etc. Chris Langton qui est un de ses principaux partisans la définit : "C'est un domaine d'étude destiné à abstraire les principes sous-jacents à l'organisation du vivant et de les implanter dans un ordinateur afin de pouvoir les étudier et les tester" [L+92]. La recherche dans le domaine de la vie artificielle a donc, deux objectifs principaux. D'abord, chercher à comprendre le fonctionnement et l'origine de la vie, et puis essayer de produire des modèles informatiques (des SCs artificiels) pour l'utilisation scientifique. Le développement de ces modèles s'appuie sur les concepts d'autonomie, de comportement individuel répondant à des stimuli de l'environnement, de viabilité, d'adaptation et de reproduction. Il n'y a pas de processus de réflexion symbolique comme en intelligence artificielle. Les composants n'ont pas de capacités cognitives évoluées mais obéissent à des règles de comportement simples. Il est ainsi possible d'obtenir un comportement collectif complexe par l'interaction de plusieurs éléments relativement simples. L'intelligence en essaim

(intelligence de groupe) offre donc un moyen de concevoir des systèmes capables de s'adapter rapidement à des conditions changeantes.

#### 1.7.2.1 Systèmes multi-agents

Le domaine des systèmes multi-agents est né à la fin des années 70 et début des années 80, de l'idée de distribuer les connaissances et le contrôle dans les systèmes d'intelligence artificielle. Cette idée a émergé d'une part du besoin de faire face à la complexité croissante de ces systèmes et a été favorisée d'une autre part par l'émergence des modèles et machines parallèles, rendant possible la mise en oeuvre opérationnelle de la distribution [Has03].

**Définition 1.11 (Agent)** Un agent est une "entité computationnelle", comme un programme informatique ou un robot, qui peut être vue comme percevant et agissant d'une façon autonome sur son environnement. On peut parler d'autonomie parce que son comportement dépend au moins partiellement de son expérience [Wei99].

**Définition 1.12 (Système multi-agents)** Un système multi-agents est un ensemble d'entités (physiques ou virtuelles) appelées agents, partageant un environnement commun (physique ou virtuel), qu'elles sont capables de percevoir et sur lequel elles peuvent agir [BDT99]. Les perceptions permettent aux agents d'acquérir des informations sur l'évolution de leur environnement, et leurs actions leur permettent entre autres de modifier l'environnement. Les agents interagissent entre eux directement ou indirectement, et exhibent des comportements corrélés créant ainsi une synergie permettant à l'ensemble des agents de former un collectif organisé.

##### 1.7.2.1.1 Types d'agents dans un SMA

[Pot04] distingue trois types d'agents dans un SMA :

- **Agent simple** : n'importe quelle partie d'un programme assez simple à comprendre, même si les interactions entre groupes formés de ce type d'agents sont largement très difficiles à comprendre.

- **Agent autonome** : c'est un programme informatique situé dans un environnement, qui est capable de faire des actions autonomes flexibles afin de réaliser ses objectifs.

- **Agent adaptatif** : c'est un agent qui peut s'améliorer au fil du temps, i.e. il devient meilleur avec le temps dans l'atteinte de ses objectifs. Donc, c'est un agent capable de changer et d'améliorer son comportement au fil du temps.

Actuellement, il existe deux tendances de recherche dans le domaine des multi-agents : la tendance cognitive, influencée par l'IA symbolique, et la tendance réactive influencée par la vie artificielle et l'intelligence "sans représentation", appelée aussi *embodied intelligence* [Has03]. Cette dernière approche est la plus adaptée pour la modélisation des systèmes complexes adaptatifs car elle inclut l'émergence et l'imprévisibilité, caractéristiques clé dans les SCs.
Le point clé des systèmes multi-agents réside dans la formalisation de la coordination entre les agents. La recherche sur les agents est ainsi une recherche sur :

1. **La décision** : On cherche à répondre aux questions suivantes : Quels sont les mécanismes de décision de l'agent ? Quelle est la relation entre les perceptions, les représentations et les actions des agents ? Comment décomposent-ils leurs buts et tâches ? Comment construisent-ils leurs représentations ?

2. **Le contrôle** : On cherche à répondre aux questions suivantes : Quelles sont les relations entre les agents ? Comment sont-ils coordonnés ?

3. **La communication** : On cherche à répondre aux questions suivantes : Quels types de message s'envoient-ils ? A quelle syntaxe obéissent ces messages ?

Les SMAs ont été largement appliqués pour la résolution de problèmes, la simulation de systèmes intelligents, la construction de mondes synthétiques et la robotique. Ils ont également des applications dans le domaine de l'intelligence artificielle où ils permettent de réduire la complexité de la résolution d'un problème en divisant le savoir nécessaire en sous-ensembles, en associant un agent intelligent indépendant à chacun de ces sous-ensembles et en coordonnant l'activité de ces agents. On parle ainsi d'intelligence artificielle distribuée.

#### 1.7.2.1.2 Systèmes multi-agents et systèmes complexes

Les motivations derrière le choix des systèmes multi-agents comme étant un modèle puissant pour la mise en oeuvre des systèmes complexes adaptatifs ne sont pas souvent introduites dans la littérature des SMAs pour la raison principale que ces travaux sont orientés application. Une autre raison qui n'a pas permis l'adoption des SMAs comme modèle des SCs est la jeunesse de la théorie des SCs et de la non standardisation des concepts. Un bon exemple de cette non standardisation est que chacun des domaines d'étude et d'utilisation des SCs utilise ses propres définitions et outils, contrairement au domaine des SMAs qui est très lié à l'informatique. Dans ce qui suit, on essaie de mettre en relief le lien entre les SMAs et les SCs à travers l'analyse des définitions et des caractéristiques des SMAs données dans les sections précédentes.
En analysant la définition des SMAs donnée par [BDT99] (Définition 1.12) on peut extraire les points suivants :

* L'agent : Comme dans un SC, l'agent dans un SMA est un composant du système (soit

un individu, une partie ou même un SC composant du système). Cependant, il existe une différence principale entre l'agent d'un SC et celui d'un SMA : un agent dans un SC est plus simple que l'agent dans un SMA.

* L'environnement : c'est la même notion d'environnement dans les deux types de systèmes. Selon la définition de [BDT99], il existe une possibilité que les agents changent l'environnement dans lequel ils évoluent, en plus du fait qu'ils sont influencés par lui. Ceci nous rappelle la notion de "co-évolution" d'un SC avec son environnement.

* Les interactions : l'interaction entre les agents du système à bas niveau dans les deux modèles mène à une complexité au niveau global.

* L'organisation : Ici, on doit souligner le fait que dans un SMA, il doit exister une auto-organisation et pas une organisation pour qu'on puisse parler d'un SC. Cette auto-organisation doit être créée par le système et pas imposée par un agent ou une entité externe, par exemple un être humain.

#### 1.7.2.2 Boids de Reynolds

L'exemple le plus célèbre de système de vie artificielle distribuée, est bien le modèle du comportement de meutes [Rey87]. L'entité individuelle est appelée "Boid". Il représente un oiseau virtuel avec des capacités de vol de base. La meute (flocking) est une collection de Boids dans un monde simulé (Fig. 1.7).

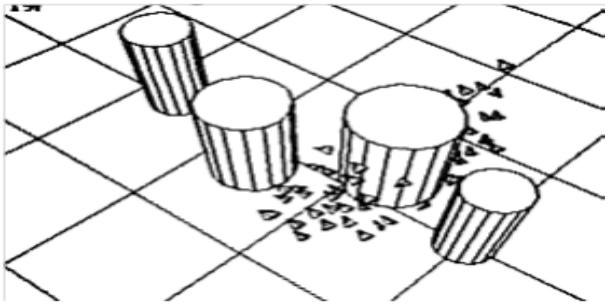

FIGURE 1.7 – Les Boids de Reynolds

Chaque Boid a une perception locale de son environnement. Il peut seulement distinguer les voisins ou les obstacles qui l'entourent (dans un petit voisinage autour de lui). Le voisinage local d'un Boid est caractérisé par une distance (mesurée depuis le centre du Boid) et un angle (mesuré à partir du vecteur de direction du Boid) (Fig.1.8). Les voisins à l'extérieur de ce voisinage sont ignorés. Le comportement de déplacement de Boids dans l'espace virtuel est

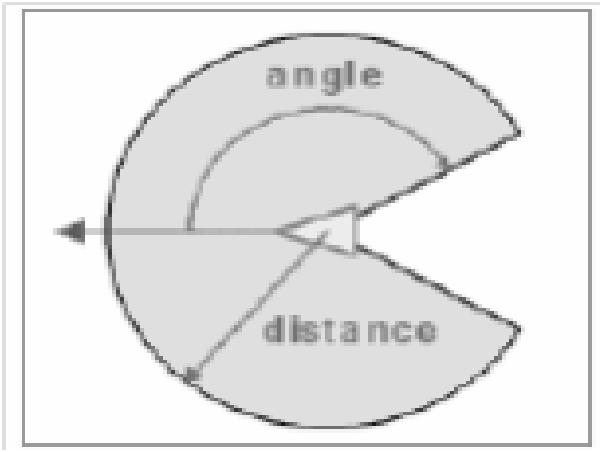


FIGURE 1.8 – Le voisinage local d'un Boid

un phénomène émergent qui résulte de leur interaction, chacun d'eux se contentant de suivre trois règles comportementales simples suivantes (Fig.1.9) :

1. Alignement : adapter sa vitesse à la moyenne de celle de ses voisins.

2. Cohésion : aller vers le centre de gravité des Boids voisins.

3. Séparation : maintenir une distance minimale par rapport aux autres objets dans l'environnement et en particulier les autres Boids.

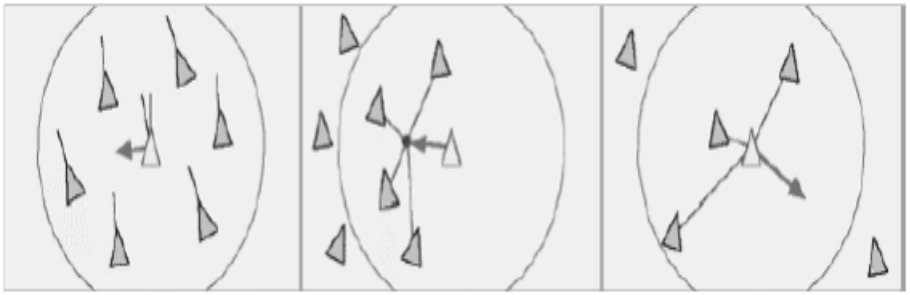

FIGURE 1.9 – Comportements de direction : alignement, cohésion et séparation

Ces règles sont suffisantes pour que les Boids adoptent un comportement semblable à un vol d'oiseaux migrateurs. Les Boids évoluent sans leader et sans contrôle global. Leur comportement est particulièrement fluide et semble très naturel lorsqu'ils évitent tous ensemble des obstacles en restant groupés. Parfois la meute se divise et produit deux groupes qui se meuvent indépendamment l'un de l'autre.Parfois aussi, deux sous-groupes se réunissent pour produire un nouveau groupe.

Les Boids ont été utilisés dans des films d'animation comme "Batman" et "Le roi Lion",

et leurs principes ont été appliqués à des domaines aussi divers que l'analyse géographique, l'exploration spatiale et le traitement d'images [Car02].

#### 1.7.2.3 L'intelligence en essaim

*Swarm intelligence*, est une forme particulière d'intelligence collective qui se base sur un essaim d'individus pourvus d'un intellect minimal mais totalement autonome. Elle est largement définie comme "*n'importe quelle tentative de concevoir des algorithmes ou des dispositifs de résolution de problèmes distribués inspirés par le comportement collectif de colonies d'insectes sociaux et d'autres sociétés animales*" [BDT99].
Les caractéristiques de la logique fonctionnelle de l'essaim peuvent être résumées dans les points suivants [BT94] :

- le fonctionnement de l'essaim se caractérise par une gestion locale des informations. Chaque agent (au travers d'une boucle réactive comportementale) exploite la physique des signaux perçus localement dans l'environnement pour opérer une action sur celui-ci.
- Le fonctionnement de l'essaim est hautement décentralisé. Il n'y pas de contrôleur central des activités individuelles qui, en intégrant l'ensemble des informations, dirigerait les actions des autres individus.
- Le fonctionnement de l'essaim n'implique, ni représentation globale de la tâche à accomplir, ni planification de celle-ci, ni représentation codée de l'environnement sous forme de carte. L'intelligence en essaim offre un moyen de concevoir des systèmes capables de s'adapter rapidement à des conditions changeantes.

#### 1.7.2.4 Les réseaux de neurones

Très librement inspirés de l'organisation "électrique" des cellules du cortex. On retrouve dans leur architecture certaines caractéristiques du système nerveux. Composés de nombreuses cellules interconnectées (les neurones) en un graphe analogue à celui formé par les dendrites et les axones (d'où le terme générique de "connexionnisme"). Le plus souvent la ressemblance entre les réseaux de neurones et le cortex se limite à ce seul principe organisationnel.

##### 1.7.2.4.1 Neurone formel

Un réseau de neurones formels est constitué d'un graphe de cellules logiques échangeant des messages à travers un graphe de connexions. Il peut alors être entièrement défini par trois caractéristiques :

- Neurones : Les noeuds de la structure. Ce sont des automates très simples dont le fonctionnement est indépendant de l'état global du réseau.

- Topologie : Correspond à l'organisation des connexions. En général, fixée a priori mais certains algorithmes permettent de la modifier dynamiquement.
- Lois dynamiques : Règles d'apprentissage, qui régissent l'évolution et le comportement global du réseau (essentiellement les règles de mise à jour). Elles définissent comment le réseau apprend puis restitue l'information.

Le neurone formel est un automate caractérisé par un petit nombre de fonctions mathématiques. Il traite un signal d'entrée recueilli à travers ses connexions entrantes pondérées, pour fournir un signal de sortie calculé par la fonction de transfert. Son état est caractérisé par trois variables E, A et S calculées au moyen de trois fonctions successives : la fonction d'entrée f(), la fonction d'activation g() et la fonction de sortie h().

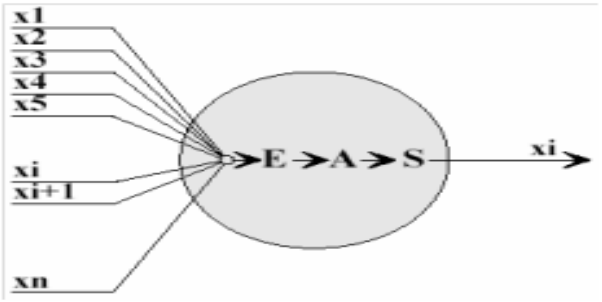


FIGURE 1.10 – Modèle général de neurone formel

On a donc successivement : E = f(x), A = g(E) et S = h(A). La définition de ces trois fonctions permet de définir le comportement du neurone (Fig.1.10).

#### 1.7.2.4.2 Principes d'apprentissage des RNs

C'est leur capacité d'apprentissage par l'exemple qui fait l'intérêt principal des réseaux connexionnistes [Bou03].

*Apprentissage supervisé* : on soumet au réseau un grand nombre d'exemples pour lesquels l'entrée et la sortie associée sont connues et les poids sont modifiés de façon à corriger l'erreur commise par le réseau.

*Apprentissage non supervisé et auto-organisation* : Contrairement aux modes supervisés, seule une base d'entrées est ici fournie au réseau. Celui-ci doit donc déterminer lui-même ses sorties en fonction des similarités détectées entre les différentes entrées.

*Apprentissage par renforcement* : Dans de nombreux problèmes on ne dispose que d'une information qualitative permettant l'évaluation de la réponse calculée, sans pour autant connaître

la réponse la plus adaptée. Les algorithmes d'apprentissage par renforcement, permettent alors d'entraîner le réseau pour qu'il fournisse, à chaque stimulus entrant, la sortie la plus adéquate (apprentissage par pénalité/récompense ou par essai/erreur).

#### 1.7.2.5 Les algorithmes évolutionnaires

Les Algorithmes Evolutionnaires (AEs) font partie du champ de l'intelligence artificielle. Ils sont apparus à la fin des années 1950. Il s'agit d'une intelligence de bas niveau inspirée de l'intelligence naturelle. Ces algorithmes sont basés sur la théorie de l'évolution et de la sélection naturelle élaborée par Charles Darwin, qui annonce que dans un environnement quelconque, seules les espèces les mieux adaptées survivent, les autres étant condamnées à disparaître. Au sein de chaque espèce, le renouvellement des populations est essentiellement dû aux meilleurs individus de l'espèce.

Les AEs constituent une approche originale : il ne s'agit pas de trouver une solution analytique exacte, ou une bonne approximation numérique, mais de trouver des solutions satisfaisant au mieux certains critères, souvent contradictoires. S'ils ne permettent pas de trouver à coup sûr la solution optimale de l'espace de recherche, du moins peut-on constater que les solutions fournies sont généralement meilleures que celles obtenues par des méthodes plus classiques, pour un même temps de calcul. Trois classes d'algorithmes évolutionnaires ont été développées indépendamment, entre les années 1960 et 1970 :

- *les algorithmes génétiques (AGs)* [Gol89] : Une autre catégorie des AEs est généralement incluse dans les AGs, c'est la programmation génétique ayant comme espace de recherche un ensemble d'instructions et de programmes au lieu de bits ou de réels.

- *la programmation évolutionnaire* [Fog97] : conçue comme une méthode d'intelligence artificielle pour la conception d'automates à états finis.

- *les stratégies évolutionnaires* [BS02].

Dans les années 90, ces trois champs de recherche ont été regroupés dans ce qui est devenu "Evolutionary Computation" [Fog97].

##### 1.7.2.5.1 Principe d'un algorithme évolutionnaire

Un algorithme évolutionnaire opère sur une population d'individus représentant des solutions potentielles du problème à résoudre. A chaque génération, l'algorithme applique les trois opérations suivantes en 4 phases :

**1. La sélection** : choisir les individus les plus performants pour la reproduction et la survie.

**2. La reproduction** : permet le brassage, la recombinaison et les variations des caractères héréditaires des parents pour former les descendants.

**3. L'évaluation** : offre à chaque individu une qualité d'adaptation qu'on appelle 'fitness'.

La sélection est appliquée en premier lieu pour choisir des individus à reproduire, et puis elle est appliquée à la fin de l'itération pour choisir les individus qui vont survivre, comme montré par l'algorithme 1

**Data**:
**Output**:

$t \leftarrow 0$
Initialiser la population $P(t)$
Evaluer $P(t)$
**while** *(le critère d'arrêt n'est pas atteint)* **do**
    $t \leftarrow (t+1)$
    $P_0(t) \leftarrow Slectionner - parents(P(t))$
    $Reproduire(P0(t))$
    Evaluer $(P0(t))$
    $P(t+1) \leftarrow$ Sélectionner-suivants $(P(t), P_0(t))$
**end**

**Algorithm 1:** Algorithme évolutionnaire

#### 1.7.2.6 Les automates cellulaires

Les Automates cellulaires ont été inventés par Stanislaw Ulam (1909-1984) et John Von Neumann (1903-1957) à la fin des années quarante, à Los Alamos National Laboratory (États-Unis) [Lan90]. Un automate cellulaire est une grille D-dimensionnelle composée d'agents appelés cellules. Ces cellules interagissent et évoluent dans le temps en changeant leurs états continuellement. Les règles de transition de chaque cellule sont locales et simples. Elles ne prennent en compte que l'état des cellules voisines.

Formellement, un automate cellulaire est un quintuplet $A = (L,D,S,N,R)$. $L$ est la grille cellulaire $D$ dimensionnelle. A chaque instant discret $t$, chaque cellule ne peut avoir qu'un seul état d'un ensemble fini $S$. A chaque étape de temps, les états de toutes les cellules sont mis à jour simultanément. Le processus de transition est guidé par une règle de mise à jour locale $R$, qui est la même pour toutes les cellules (équation 1.1). Cette règle reçoit comme entrée le voisinage local $N$ de la cellule (états des cellules voisines), et produit l'état futur de la cellule [GSD+03].

$$S^{t+1}{}_{i,j} = R(N(S^{t}{}_{i,j})) \tag{1.1}$$

#### 1.7.2.6.1 Automates cellulaires et systèmes complexes

Pour simuler le comportement des systèmes complexes, les automates cellulaires ont été utilisées comme un modèle mathématique puissant [Lan90]. Ils ont l'avantage d'être faciles à comprendre et à mettre en oeuvre [Fro05]. Dans un contexte informatique pour la simulation et l'exploitation des SCs, les automates cellulaires ont les avantages suivants par rapport aux techniques conventionnelles :

1. Ils offrent des coûts faibles de calcul pour des modèles à haute complexité.
2. Ils sont faciles à programmer.
3. Ils permettent une contrôlabilité élevée du modèle.
4. Ils sont faciles à visualiser.
5. Ils permettent une possibilité de paralléliser un grand nombre d'entrées.
6. Ils ont une grande portabilité d'autres modèles mathématiques de calcul.
7. Ils sont bien adaptés à la "siliconisation".

Le "jeu de la vie" est l'un des modèles des ACs les plus proches du comportement d'un SC. l'évolution des objets manipulés, bien que totalement déterminée par les fonctions de transition, est hautement imprévisible. On remarque aussi que le simple fait d'ajouter ou d'enlever une cellule dans une configuration change son évolution d'une façon radicale, c'est exactement le même phénomène observé dans le système écologique dans lequel n'importe quelle petite modification, même très locale, va influencer tout le système d'une façon imprévisible.

Pour synthétiser les points forts et les points faibles des deux modèles, ACs et SMAs, le tableau 1.1 (extrait de [Fro05]) donne une petite comparaison entre les deux approches pour la modélisation des SCs. Ce tableau reflète bien les avantages des ACs par rapport aux SMAs pour la simulation et l'exploitation des SCs.

TABLE 1.1 – Une comparaison entre AC et SMA pour la modèlisation des SCs

| Propriétés/Système | **Automate cellulaire (CA)** | **Système multi-agent (SMA)** |
|---|---|---|
| Phénomènes décrits | Patterns et structures complexes | Structures simples |
| Agents, Objets et Composants | Agents simples, machines d'états finis ou automates | Agents complexes |
| Règles locales de base | Table de transition | Langage de communication et intentions entre agents |
| Phénomènes émergents | Structures préparées | Culture puissante...etc |

## 1.8 Conclusion

Le maintien de l'organisation dans la nature ne peut être réalisé par une gestion centralisée. Les SCs permettent l'adaptation aux circonstances environnementales ; ils s'adaptent dynamiquement aux modifications de l'environnement, ce qui les rend extraordinairement flexibles et robustes face aux perturbations externes. Nous soulignons donc la supériorité des systèmes naturels par rapport à la technologie humaine classique, qui évite la complexité et gère de manière centralisée la grande majorité des systèmes techniques. Une technologie entièrement nouvelle devra être développée pour exploiter le grand potentiel d'idées et de règles des SCs. Lorsqu'un système réunit toutes les caractéristiques nécessaires, pour qu'il soit qualifié de complexe (composants simples et en nombre important, interactions locales et nonlinéaires), il peut fournir un comportement de haut niveau pertinent et adaptatif, face aux problématiques qu'il rencontre. L'utilisation de cette capacité à fournir un comportement global complexe, à partir de règles d'interaction locales simples, fait des SCs un modèle à suivre pour la conception de systèmes artificiels (logiciels en particulier). De cette façon on pourra résoudre des problémes concrets et difficiles, simplement en exploitant la dynamique induite par un ensemble d'éléments simples, qu'on met en interaction, pour faire émerger le fonctionnement requis du système. Comme on a pu l'observer dans ce chapitre, l'avenir appartient sans conteste aux SCs. Mais, leurs applications réelles, sont pour ainsi dire balbutiantes. Il n'y a que récemment, que l'ordinateur nous a permis de commencer à les exploiter plus profondément. On a aussi parlé d'émergence, une caractéristique primordiale des SCs, que l'on veut utiliser pour résoudre des problématiques concrètes. C'est l'émergence donc, qui nous fournira le moyen de résoudre des tâches complexes sans pour autant répercuter cette complexité sur le processus de conception (faire émerger un comportement complexe à partir d'ingrédients simples). Les automates cellulaires seront revus avec plus de détails dans le chapitre suivant.

Chapitre 2

# Les Automates Cellulaires

## 2.1 Introduction

Les Automates Cellulaires (ACs) sont un outil mathématique puissant, pour modéliser les Systèmes Complexes. Inventés par deux des plus renommés mathématiciens de notre siècle, Ulam et Von Neumann.Reprenant la théorie des automates autoreproducteurs de Von Neumann, Christopher Langton créa en 1984 le premier automate cellulaire. Un réseau d'automates cellulaires est une matrice régulière de machines très simples. Ces machines sont parmi les plus élémentaires que l'on puisse imaginer, car elles ne peuvent se trouver à un instant donné que dans un seul état choisi parmi un nombre restreint d'états possibles. Les règles de transitions permettent de définir exactement l'état d'un automate en fonction de celui de ses voisins.
Malgré cette simplicité, les réseaux d'automates cellulaires peuvent être considérés comme des approximations discrètes des systèmes physiques dynamiques non-linéaires. Ils sont ainsi, à plus d'un titre, une approche intéressante pour la vie artificielle et la modélisation des systèmes complexes.

## 2.2 Définitions

### 2.2.1 Définition générale

Un automate cellulaire consiste en une grille régulière de cellules contenant chacune un état choisi parmi un ensemble fini d'états et qui peut évoluer au cours du temps. L'état d'une cellule au temps $t+1$ est fonction de l'état au temps t d'un nombre fini de cellules appelé son voisinage. À chaque nouvelle unité de temps, les mêmes règles sont appliquées simultanément à toutes les cellules de la grille, produisant une nouvelle génération de cellules dépendant

entièrement de la génération précédente.
L'automate cellulaire est caractérisé par :
- nombre de dimensions de la grille. Exemples : $1D$, $2D$, $3D$...etc
- nombre d'états ou couleurs. Exemples : allumé et éteint, 256 niveaux de gris...etc
- voisinage considéré d'une cellule. Exemples : Von Neumann : 4 cellules en $2D$, Moore : 8 cellules en $2D$...etc (Fig. 2.1).
- règles définissant l'automate et le déroulement des générations. Exemple : jeu de la vie...etc

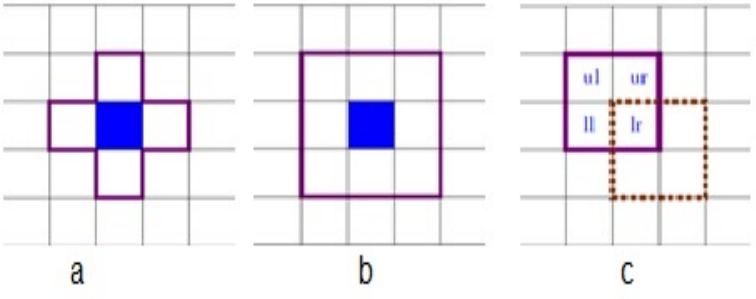


FIGURE 2.1 – Les voisinages d'un automate cellulaire. a) Voisinage de Von Neumann b) Voisinage de Moore c) Voisinage de Margolus

### 2.2.2 Définition formelle

Formellement, un AC est un quintuplé A = (L, d, S, N, R), d est la dimension de l'AC, L est la grille de cellules. S est l'ensemble des états possibles d'une cellule. A chaque top d'horloge t, toutes les cellules changent simultanément d'état. Cette transition est gérée par une règle locale R. Cette règle de transition a comme argument les cellules du voisinage N de la cellule concernée, et fournit son état futur (à l'instant $t + 1$).

$$S^{t+1}{}_{i,j} = R(N(S^{t}{}_{i,j})) \tag{2.1}$$

On distingue trois périodes dans l'histoire des ACs :
1) leur création, qui fut une retombée des travaux de Von Neumann et d'Ulam sur l'auto-reproduction
2) le développement de l'AC 'Jeu de la Vie'
3)le renouveau, où l'on a commencé à explorer de nouvelles pistes pour l'exploitation des ACs pour des applications de simulation (physique, biologie, sciences sociales...etc) et pour la résolution de problèmes en général. La puissance de cet outil est de plus en plus évidente et promet un avenir prospère.

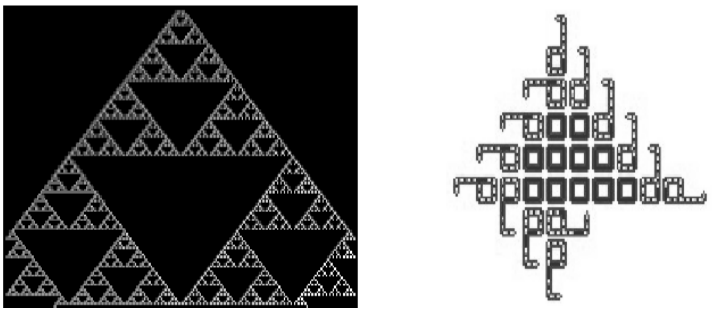

FIGURE 2.2 – Exemples d'automates cellulaires à une dimension. a) Triangle de Sierpinski b)Boucles de Langton

#### 2.2.2.1 Naissance des Automates Cellulaires

Les ACs ont été inventés par Stanislas Ulam (1909-1984) et John Von Neumann (1903-1957) à la fin des années 40 au Los Alamos National Lab. En 1948 Von Neumann, proposa un article intitulé "Théorie générale et logique des automates". Une des questions centrales abordées était de savoir s'il était possible de concevoir une machine capable de s'auto-reproduire. La solution à ce problème vint de son collègue Ulam, qui s'intéressait aux "objets géométriques définis de façon récursive" (fractales). Ces objets provenaient de jeux aux règles simples dans lesquels on pouvait voir des figures (patterns) se développer, se reproduire, combattre pour une portion de territoire et mourir.
Ces jeux se déroulaient dans un univers "cellulaire", composé d'une matrice infinie où les cellules, régulièrement réparties, peuvent être dans un état passif ou actif. Les figures de cet univers étaient composées des cellules actives, et à tout moment, l'état futur de chaque cellule était dicté par l'état des cellules avoisinantes.

**Le problème de l'autoreproduction** : Von Neumann aboutit à l'idée qu'un automate autoreproducteur devrait comporter un "constructeur universel" capable de fabriquer n'importe quelle machine (cellulaire) à partir d'une "description" qui lui serait fournie. Dans le cas particulier où l'on fournirait la description du constructeur universel au constructeur universel lui-même, il y aurait autoreproduction. Le système autoreproducteur (SAR) est constitué du constructeur universel (CU) et de sa propre description (DESC). Or cette description ne peut être la description du constructeur universel seulement, elle doit être la description de tout le système, et donc être en particulier une description de la description. En équations, nous avons : SAR = CU + DESC (SAR), ce qui paraît à priori insoluble étant donné l'existence d'une régression à l'infini (récursivité à gauche : quand on remplace SAR à droite par le contenu de l'équation).
Le problème fut résolu par Von Neumann en rajoutant une troisième unité : une machine de Turing, appelée superviseur, qui orchestre le processus. L'utilisation du superviseur évitait la

régression à l'infini en distinguant deux phases : L'ensemble (constructeur universel + superviseur), le "copieur-superviseur" réalise une copie de lui-même dans une région vide de l'espace en lisant la description, c'est la phase d'interprétation (phase 1).
La phase 1 étant terminée, le superviseur comprend qu'il ne faut plus que la description soit interprétée ; celle-ci est considérée comme un ensemble de données et recopiée littéralement pour rebâtir le système initial.
En 1952, Von Neumann proposait une version qui utilise 29 types de cellules différentes. L'état de chaque cellule au temps t était déterminé uniquement par l'état des quatre cellules adjacentes et celui de la cellule centrale au temps $t-1$. Ce voisinage est d'ailleurs nommé voisinage de Von Neumann.

### 2.2.3 Exemples d'Automates Cellulaires

#### 2.2.3.1 Automate Cellulaire Elémentaire

L'automate cellulaire trivial le plus simple que l'on puisse concevoir consiste en une grille unidimensionnelle (1D) de cellules ne pouvant prendre que deux états : 0 ou 1, avec un voisinage constitué, pour chaque cellule, d'elle-même et des deux cellules qui lui sont adjacentes. Chacune des cellules pouvant prendre deux états, il existe $2^3 = 8$ configurations (ou motifs) possibles d'un tel voisinage. Pour que l'automate cellulaire fonctionne, il faut définir quel doit être l'état, à la génération suivante, d'une cellule pour chacun de ces motifs. Il y a $2^8 = 256$ façons différentes de s'y prendre, soit donc 256 automates cellulaires différents de ce type.
Souvent on désigne les automates de cette famille par un entier entre 0 et 255 dont la représentation binaire est la suite des états pris par l'automate sur les motifs successifs 111, 110, 101, etc. À titre d'exemple, considérons l'automate cellulaire défini par la table 2.1, qui donne la règle d'évolution :

TABLE 2.1 – Tableau représentant le fonctionnement de la règle de transition 30

| Motif initial | 111 | 110 | 101 | 100 | 011 | 010 | 001 | 000 |
|---|---|---|---|---|---|---|---|---|
| Valeur suivante de la cellule centrale | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 |

Cela signifie que si par exemple, à un instant t donné, une cellule est à l'état « 1 », sa voisine de gauche à l'état « 1 » et sa voisine de droite à l'état « 0 », au temps t+1 elle sera à l'état « 0 ». Si l'on part d'une grille initiale où toutes les cellules sont à l'état « 0 » sauf une, on aboutit à la configuration suivante (Fig. 2.3 ) où chaque ligne est le résultat de la ligne précédente. Par convention cette règle est nommée « règle 30 », car 30 s'écrit 00011110 en binaire et 00011110 est la deuxième ligne du tableau ci-dessus, décrivant la règle d'évolution.

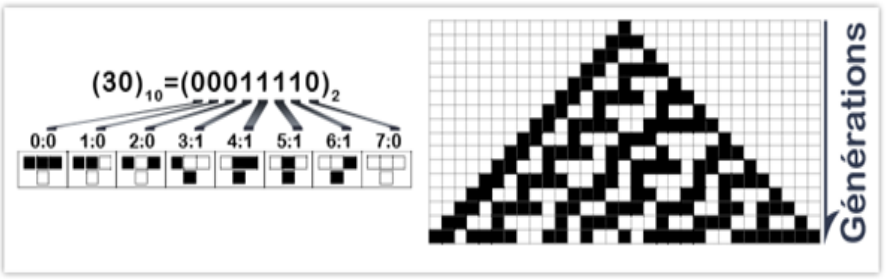


FIGURE 2.3 – Automate cellulaire élémentaire produit par la règle 30

#### 2.2.3.2 L'automate cellulaire "Jeu de la Vie"

Les règles du Jeu de la Vie sont simples. Les cellules peuvent se trouver dans deux états qui sont : vivante / morte. L'espace cellulaire est composé de cellules qui se trouvent dans l'état mort au départ, sauf pour un nombre fini d'entre elles. L'évolution de chaque cellule est déterminée en fonction du nombre $N_v$ de cellules vivantes se trouvant dans les huit cases adjacentes à une cellule. Les règles sont :

- Une cellule vivante meurt, si $N_v \leq 1$ : isolement.
- Une cellule vivante meurt, si $N_v \geq 4$ : surpeuplement.
- Une cellule morte peut devenir vivante pour $N_v = 3$ : reproduction.

L'évolution des objets manipulés, bien que totalement déterminée par les fonctions de transition, est hautement imprévisible. On remarque ainsi qu'il n'y a pas de correspondance apparente entre la taille d'une configuration initiale et le temps qu'elle met pour se stabiliser. Par ailleurs, le simple fait d'ajouter ou d'enlever une cellule dans une configuration change son évolution de façon radicale.
La figure 2.4 illustre la suite des générations d'une configuration initiale aléatoire à l'aide d'un automate "Jeu de la Vie".

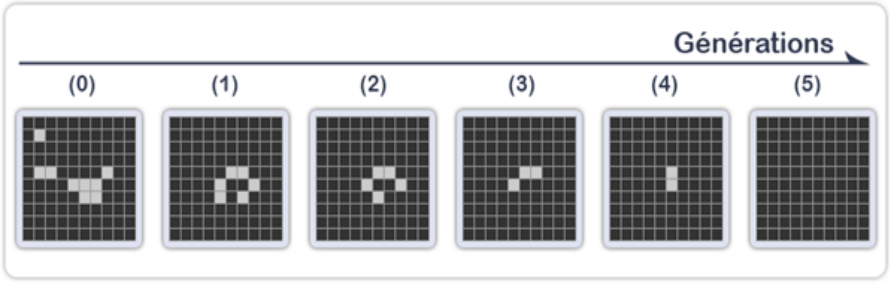


FIGURE 2.4 – Suite de générations produites par l'automate "Jeu de la Vie"

#### 2.2.3.3 Exploration des possibilités des AC

Parallèlement aux recherches tardives qui se déroulaient sur le Jeu de la Vie, des chercheurs du MIT étudiaient d'autres ACs ayant des règles simples et qui conduisaient aussi à des comportements imprévisibles.
L'intérêt pour les ACs va connaître un regain après la publication d'un article de fond émanant de Stephen Wolfram [Wol83]. Le point fort de la recherche de Wolfram est d'avoir proposé une classification des ACs en quatre catégories, en s'inspirant de la théorie des systèmes dynamiques : Classe I (systèmes statiques), Classe II (systèmes cycliques), Classe III (systèmes chaotiques) et Classe IV (systèmes complexes).

### 2.2.4 La classification de Wolfram

Wolfram a classé les ACs en quatre catégories :

**Classe 1 : les systèmes statiques :** ils produisent des diagrammes espace-temps uniformes. Ce type d'automates convergent vers un espace d'état monotone (attracteur fixe).

**Classe 2 : les systèmes cycliques :** ils produisent des phénomènes qui s'inscrivent sur des orbites périodiques. Cette périodicité n'est généralement qu'une continuité de l'état initial.

**Classe 3 : les systèmes chaotiques :** Ils produisent des phénomènes chaotiques. Ils sont soumis à la sensibilité aux conditions initiales et produisent des structures sans organisations apparentes.

**Classe 4 : Les systèmes complexes :** à ce jour encore ils sont incompris mathématiquement, ils sont présents dans des phénomènes et des structures qui semblent localement organisés. Ces automates font par exemple apparaitre des formes ayant une existence propre à travers les générations, du même type que les structures émergentes dans le jeu de la vie (comme le planeur).

Ces classes sont illustrées sur la figure 2.5

Si les automates de classe I et II sont à ce jour bien compris, les automates de classe III et IV posent encore nombre de problèmes. Pour isoler les caractéristiques essentielles des structures et des phénomènes émergents de ce type d'automates, les scientifiques ont développé des

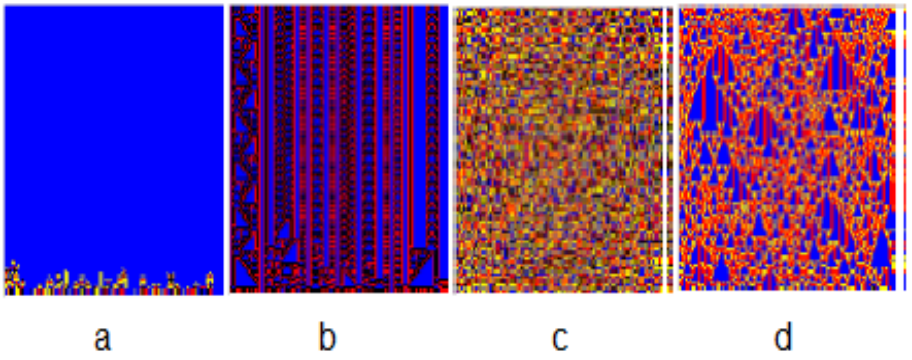


FIGURE 2.5 – Différentes classes d'automates : a) classe I b) classe II c) classe III d) classe IV

méthodes de groupage. Le groupage consiste à s'abstraire de l'information liée à la nature des structures qui apparaissent dans les automates cellulaires pour n'en extraire que le modèle général. Ainsi, en remplaçant au sein de l'automate les structures détectées par de nouveaux motifs uniformes, il est possible de faire apparaitre de nouvelles structures sous-jacentes. Ces méthodes de groupage ont permis de déterminer une échelle de complexité entre les automates cellulaires. En effet, certains automates présentent des dynamiques et des structures qui, observées à un plus haut niveau (après le groupage par exemple) se retrouvent dans d'autres automates. La complexité de l'automate inclus dans le premier est donc plus faible.
Nous pouvons dire qu'à partir du début des années 90, le domaine des ACs a atteint une phase de maturité. Cette maturité se traduit au niveau théorique par l'exploration systématique de grandes classes d'AC comme le font Toffoli, Wolfram et Langton. Au niveau pratique, les ACs ont des applications dans des domaines aussi variés que la reconnaissance de formes, la cryptographie, l'étude du développement des systèmes urbains, [Bou03] etc. Il semble donc qu'un nouveau paradigme scientifique se soit développé. Ses caractéristiques principales étant de traiter les problèmes selon une approche ascendante (du plus simple au plus complexe), parallèle et en déterminant les comportement des entités élémentaires de façon locale.

## 2.3 Domaines d'Application des Automates Cellulaires

Les applications pratiques des automates cellulaires sont nombreuses et diverses. Fondamentalement, ils constituent des univers dont on fixe les lois. Notre Univers est soumis aux lois de la Physique. Ces lois ne sont que partiellement connues et apparaissent hautement complexes. Dans un automate cellulaire, les lois sont simples et complètement connues. On peut ainsi tester et analyser le comportement global d'un univers simplifié. Voici quelques exemples d'application.

### 2.3.1 Modélisation en Physique

Parmi les modèles qui ont été très étudiés, on peut citer les automates de gaz sur réseau (Lattice Gas Automata) qui modélise des particules de gaz régies par une version discrète des équations de Navier Stokes. La première formulation de ce modèle s'agit d'un automate cellulaire de deux dimensions (2D), connu sous le nom de HPP, par J. Hardy, Y. Pomeau et O. Pazzis dans les années 70. Une seconde formulation, FHP, a été proposée dans les années 80 par U. Frisch, B. Hasslacher et Y. Pomeau : elle améliore la précédente en remplaçant le réseau de cellule Z2 par un réseau hexagonal.
- Étude des matériaux magnétiques selon le modèle d'Ising (1925) : ce modèle représente le matériau à partir d'un réseau dont chaque noeud est dans un état magnétique donné. Cet état, en l'occurrence l'une des deux orientations du moment magnétique, dépend de l'état des noeuds voisins.
- Simulation des processus de percolation.
- Simulation des processus de cristallisation.

### 2.3.2 Codage

- Cryptographie : l'automate cellulaire a été proposé pour la cryptographie des clés publiques. La règle 30 a été suggérée comme une possibilité de chiffrement par bloc dans la cryptographie.
- Dans le domaine de codage/décodage, les automates cellulaires ont été utilisés pour la conception des correcteurs d'erreurs rapides.

### 2.3.3 Mathématiques

Les automates cellulaires peuvent être utilisés comme alternative aux équations différentielles.

### 2.3.4 Électronique

- Pour la conception d'ordinateurs massivement parallèles.

### 2.3.5 Phénomènes Biologiques

Des formes complexes composées de formes fractales qui se ressemblent plus ou moins en échelles variées, existent partout dans la nature, par exemple : Les côtes des continents, les cours des rivières, les nuages du ciel, les branches des plantes et les veines de leurs feuilles, les vaisseaux sanguins dans les poumons, la forme des coquillages, et les flocons de neige. Toutes ces fractales sont des modèles auto-similaires qui abondent. Ces modèles naturels de formes fractales, d'une grande complexité apparente, peuvent être simulés par des programmes informatiques simples basés sur des automates cellulaires, générant des résultats qui ressemblent à ceux observés dans la nature (Fig.2.6).

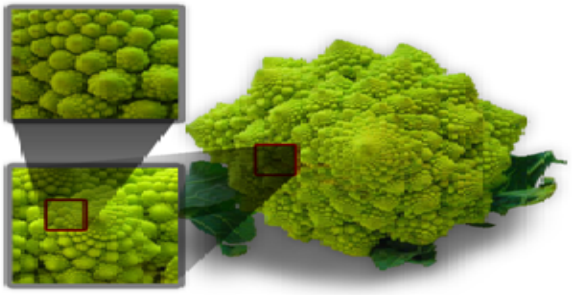

FIGURE 2.6 – Image en 3D d'un brocolis, développée par un AC

Les motifs de certains coquillages, comme les *cônes* et les *Cymbiolae*, sont générés par des mécanismes s'apparentant au modèle des automates cellulaires. Les cellules responsables de la pigmentation sont situées sur une bande étroite le long de la bouche du coquillage. Chaque cellule sécrète des pigments selon la sécrétion (ou l'absence de sécrétion) de ses voisines et l'ensemble des cellules produit le motif de la coquille au fur et à mesure de sa croissance. Par exemple, l'espèce *Conus textile* présente un motif ressemblant à la règle 30 décrite plus haut (Fig.2.7).

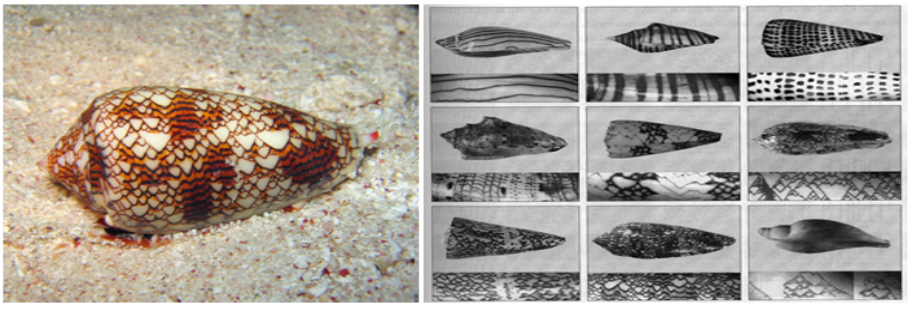

FIGURE 2.7 – Coquillage avec motif similaire au résultat de la règle 30

### 2.3.6 Autres applications

-Simulation et étude du développement urbain.
-Simulation de la propagation des feux de forêt.
-Dans un domaine plus quotidien, les automates cellulaires peuvent être utilisés comme générateur des effets graphiques.

## 2.4 les Automates Cellulaires en Traitement d'images

### 2.4.1 Intérêt des AC en traitement d'images

Un automate cellulaire est une grille de cellules, facilement assimilable à une grille de pixels, la cellule représente le pixel, la dimension de l'automate est $2D$, l'état de la cellule représente la couleur du pixel en question (noir et blanc, niveaux de gris ou couleur), et le voisinage du pixel peut être considéré comme celui de Moore ou Von Neumann. L'état initial de l'automate cellulaire représente l'image en entrée (image à traiter), et après plusieurs générations de transition de l'automate cellulaire, on aboutit à la configuration finale qui est l'image résultat attendue (image segmentée, débruitée...etc), Le problème réside dans la recherche des règles de transition adéquates, susceptibles de faire évoluer l'automate cellulaire jusqu'à la configuration finale. Le domaine de traitement d'images est prédisposé à la recherche avec les automates cellulaires, vu que l'image est une représentation triviale de l'AC. Leur nature locale et le parallélisme intrinsèque lié à leur implémentation en fait un outil naturel pour le traitement d'images. À partir des années 90, il y'a eu un intérêt croissant de la part des chercheurs pour les propriétés des AC en traitement d'images. Dans la littérature, on trouve plusieurs travaux qui exploitent les propriétés intéressantes des AC, ainsi que leur grande puissance de calcul, pour effectuer des opérations de traitement d'images.

### 2.4.2 Etat de l'art des AC en traitement d'images

Dans ce paragraphe, nous allons décrire quelques travaux utilisant les AC pour accomplir diverses tâches de traitement d'images :
- Dans [PP02], les auteurs présentent un algorithme basé sur un automate cellulaire symétrique, avec une fonction de transition dynamique, pour le filtrage de bruit et la détection de contours sur images binaires.

-[WT07] a présenté un algorithme d'automate cellulaire uniforme pour le traitement d'images médicales. Cet algorithme réalise le filtrage de bruit et la détection de tumeurs dans des mammographies. Une extension de ce travail dans [WT08] a porté sur l'identification du muscle pectoral dans les images de mammographies.
- Dans [BMAH09], les auteurs présentent une approche pour le filtrage de bruit basée sur un AC et qui évolue avec un algorithme évolutionnaire quantique.
- Dans [ZYY07], les auteurs ont utilisé une approche automate cellulaire avec la distance euclidienne pour l'extraction de contours.
- Dans [KS10], les auteurs ont repris l'algorithme de [WT07] et l'ont appliqué avec succès à la détection de contours sur images à niveaux de gris.
- Dans [SH09], les auteurs décrivent une méthode simple de débruitage d'images, basée sur un AC évoluant avec une règle de transition simple.
- Travaux de [SBM07], [BMA06] : les auteurs ont réalisé une détection de contours sur images binaires, en utilisant un paquet de règles, extrait par un algorithme génétique.
- Travaux de [Ros06], [Ros10] : l'auteur a opéré différentes tâches simples de traitement d'images : filtrage de bruit sur images binaires et à niveaux de gris, amincissement, trouver les régions convexes dans une image, détection de contours.

## 2.5 Problématique et motivations

Malgré leur utilité et leur capacité de résolution dans différents domaines, les ACs présentent un problème majeur, sur lequel butent beaucoup de scientifiques et de chercheurs. En effet, comment extraire parmi un immense ensemble de règles de transition, l'ensemble minimal de règles qui réalisent la fonction finale désirée : filtrage, ou détection de contours ou autre chose.

### 2.5.1 Formulation du problème

Formellement, le problème est posé comme suit : Soit A un AC, f est sa fonction de transition et V est l'ensemble des états cellulaires. On note que f est une fonction de $V_n$ dans V, n étant le nombre de cellules par voisinage. Il en résulte que l'ensemble des fonctions de transitions s'il est défini de manière extensionnelle est fini (on suppose ici que l'ensemble V est fini). Le cardinal de l'ensemble $V_n$ est égal à $card(V)^n$ ce qui représente en fait le nombre de toutes les configurations possibles du voisinage. Formellement, le cardinal de l'ensemble des fonctions de transitions est donc donné par la formule suivante :

$$(f : V^n \to V) \wedge (card(V^n) = card(V) \wedge n) \Longrightarrow card(f) = card(V) \wedge (card(V) \wedge n)$$

Par exemple, si l'on considère un automate unidimensionnel à deux états et un voisinage local de 7 cellules, nous avons donc $card(V) = 2$ et $n = 7$, d'où : $card(f) = 2^{2^7} = 2^{128}$.
Si l'on considère un AC bidimensionnel à deux états dont le voisinage est de 9 cellules constitué d'une cellule au centre et ses huit voisines immédiates. Nous avons donc $card(V) = 2$ et $n = 9$, d'où : $card(Vn) = (2^9) = 512$ et $card(f) = 2^{2^9} = 2^{512}$, f et V étant respectivement la fonction de transition et l'ensemble des états cellulaires.
En effet, le nombre de règles augmente exponentiellement avec le nombre d'états d'une cellule, et son type de voisinage, par exemple : Pour une cellule à 256 états et 8 voisins, on a $256^{256^8}$ règles possibles. La question qui se pose est : comment extraire dans un tel ensemble, les seules règles capables de réaliser l'objectif voulu ?
Notre travail de thèse se situe dans cette optique : on veut trouver une solution au problème d'extraction de règles d'un automate cellulaire en appréhendant des tâches de traitement d'images. Pour cela, on utilise les automates cellulaires comme système complexe pour modéliser une image.On démarre d'une configuration initiale de l'automate cellulaire, qui correspond à l'image originale à traiter, et on aboutit après plusieurs évolutions de l'automate à une configuration finale qui est l'image traitée (segmentée, filtrée..).
Quelle démarche allons nous adopter pour isoler, dans un ensemble immense de règles de transition, les seules règles qui nous permettent d'aboutir au résultat désiré ? Ce problème est connu sous le nom de : Problème Inverse.

### 2.5.2 Problèmes Inverses d'Automates Cellulaires

Dans cette partie, nous donnons dans un premier temps une définition du problème inverse de façon générale puis du problème inverse des automates cellulaires. Ensuite, nous présentons quelques problèmes inverses des automates cellulaires traités dans la littérature.

#### 2.5.2.1 Définitions

De façon générale, dans un problème direct, on déduit les effets connaissant les causes [Kel76]. A l'inverse du problème «direct», un problème est «inverse» si les effets sont connus et on cherche à connaitre les causes.

**1. Définition du problème inverse**
D'un point de vue mathématique, un problème inverse consiste à inverser un ou plusieurs

opérateurs [Cav11]. Un problème inverse classique est défini comme suit : soit $A$ un opérateur de l'espace de Hilbert $H$ dans $K$.

$$\text{Pour } k \in K \text{ donné, déterminer } f \in H \text{ telle que } Af = k$$

Le problème est donc de gérer cette inversion pour obtenir une reconstruction précise. Il existe des cas où l'on ne peut pas inverser cet opérateur. C'est le cas des problèmes *mal posés*.

**2. Définition du problème bien posé**

Un problème est bien posé [Had32], s'il respecte les conditions suivantes :
- Il existe une solution au problème ;
- La solution si elle existe est unique ;
- La solution dépend constamment des données.

Un problème inverse peut aussi se définir lorsqu'une des trois conditions précédentes n'est pas satisfaite.

**3. Définition de l'émergence inversée**

L'émergence inversée réfère à un problème inverse dont les sorties du problème direct sont obtenues par émergence.

En règle générale, la simulation des systèmes par automates cellulaires consiste à créer l'environnement le plus proche de celui du phénomène étudié (conditions initiales, paramètres, règles d'évolution,...etc). Ce type de problèmes est dit « direct » car on veut connaitre l'état futur de l'automate cellulaire connaissant sa configuration initiale et les règles qui régissent son évolution.

Dans certains cas, afin d'obtenir une structure à une génération donnée ou une dynamique particulière de l'automate cellulaire, il est important de déterminer et choisir une fonction de transition qui permettrait à l'automate d'évoluer et de donner cette structure ou cette dynamique. Ce problème est connu sous le nom du problème inverse ou problème d'identification des automates cellulaire. On peut distinguer différents problèmes inverses :

1. Déterminer la fonction de transition de l'automate cellulaire permettant de retrouver des structures dynamiques particulières (comportement) de l'automate cellulaire.
2. Déterminer la fonction de transition de l'automate cellulaire en connaissant ses configurations initiale et finale.
3. Retrouver la configuration initiale de l'automate cellulaire en connaissant sa fonction de transition et sa configuration finale.

Dans ce travail, nous nous intéressons uniquement au deuxième cas des problèmes inverses des automates cellulaires. [GSD$^+$03] ont qualifié ce problème d'extrêmement difficile.

#### 2.5.2.2 Quelques Problèmes Inverses d'Automates Cellulaires

Le problème inverse des automates cellulaires consiste à déterminer les règles d'évolutions locales des cellules (fonction de transition) permettant d'obtenir une configuration de l'automate cellulaire donnée.
Cette partie présente quelques problèmes inverses des automates cellulaires traités dans la littérature dans différentes spécialités allant de la théorie à la biologie en passant par la physique (Fig.2.8). Tous ces problèmes ont pour objectif de déterminer la fonction de transition de l'automate cellulaire.

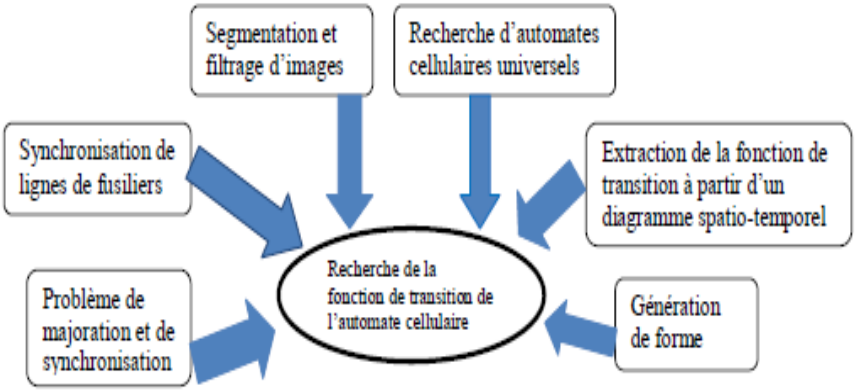


FIGURE 2.8 – Quelques problèmes inverses des automates cellulaires

**a) Problème de majoration et de synchronisation :**

* Problème de majoration : également appelé problème de classification de densité.Il consiste à chercher une fonction de transition d'un automate cellulaire binaire permettant de classifier le maximum de configurations initiales. Pour résoudre ce problème, Mitchell [MCD+96] a utilisé un algorithme génétique pour la recherche d'une fonction de transition qui permet de classifier le maximum de configurations initiales.

* Problème de synchronisation : Il consiste à trouver la fonction de transition de l'automate cellulaire dont, après un nombre de générations important, la configuration finale oscille entre une configuration ne comportant que des cellules à l'état « 0 » et une configuration où toutes les cellules sont à l'état « 1 ». Un algorithme génétique a été également proposé pour résoudre ce problème par [MCD+96].

**b) Synchronisation des lignes de fusiliers :**

Le problème de lignes de fusiliers a été inventé par Myhill en 1957 mais formalisé par [Moo64] comme suit : « Soit un automate cellulaire 1D où toutes les cellules sont à un même état sauf

les deux qui sont aux bords. L'état d'une cellule dépend de son état et de celui de ces deux voisines (une de chaque côté). Au départ, toutes les cellules sont dans un état appelé « latent » sauf l'une des deux cellules situées aux bords. Le problème consiste à déterminer le nombre d'états et à trouver une fonction de transition de l'automate de telle façon à ce que au bout d'un certain nombre de générations, toutes les cellules se mettent en même temps à un état jamais apparu avant appelé état de « feu » ».
La définition suivante [Yun93] n'est qu'une définition imagée du problème mais qui lui a donné son nom : « comment synchroniser une ligne de fusiliers pour qu'ils se mettent à tirer tous en même temps, alors que l'ordre donné par un général situé à l'un des deux bords de l'escadron met un temps pour se propager ? ». En d'autres termes, ce problème consiste à déterminer la fonction de transition avec le minimum de nombre d'états en un minimum de temps. Plusieurs travaux se sont intéressés à la résolution de ce problème. Les algorithmes de résolution sont majoritairement des heuristiques [UHS05].

**c) Filtrage et segmentation d'images :**

Plusieurs travaux dans la littérature proposent d'utiliser les automates cellulaires pour segmenter des images en utilisant l'approche par contours. La segmentation par automate cellulaire consiste à déterminer la fonction de transition permettant à l'automate cellulaire d'évoluer d'une configuration initiale qui est l'image à segmenter vers une configuration finale représentant les contours de l'image à segmenter.
Dans [SBM07] et [BMA06], les auteurs utilisent un algorithme génétique pour extraire les règles d'un automate cellulaire pour la détection de contours sur images binaires. Une population initiale d'AC candidats est assimilée à des chromosomes. Un processus de sélection d'individus par croisement et mutation se fait itérativement. De ce processus cyclique, va émerger un ensemble d'individus (règles) intéressants, qui ont une fitness élevée. L'un d'eux sera la solution retenue.
Dans [Ros06] l'auteur utilise une approche SFFS (Sequential Floating Forward Search), basée sur le Hill Climbing, pour sélectionner les règles. Cette approche a l'avantage d'être simple et facile à implémenter. Et elle ne nécessite pas de paramètres à régler comme pour les AGs. Elle a été appliquée pour le filtrage d'images binaires, amincissement de formes..
* Dans [Ros10], l'auteur décrit une version modifiée de SFFS pour traiter directement et efficacement des images à niveaux de gris. Comme le nombre de règles augmente dans ce cas, la méthode utilisée vise à trouver un codage dans le voisinage immédiat d'un pixel, pour réduire le nombre d'états de 256 à 3. le nombre de règles à considérer a été réduit à 954. les applications en traitement d'images sont : le débruitage, détection de contours, détection d'arêtes et vallées (creux).

**d) Recherche d'automates cellulaires universels :**

Turing [Tur48] a introduit la notion d'une caractéristique fondamentale appelée universalité. Un automate cellulaire est universel s'il est capable de calculer toutes les fonctions calculables, c'est ce que l'on appelle l'Universalité de Turing. Une fonction $f$ est calculable s'il existe un algorithme permettant de calculer $f(x)$ pour chaque argument $x$ en un nombre d'états fini.
[BCG82]ont montré que le jeu de la vie est un automate cellulaire universel car il peut réaliser n'importe quelle fonction calculable et que toutes les portes (opérateurs) logiques peuvent être réalisées à l'aide d'interactions entre figures connues, stables ou périodique. Le planeur (les planeurs sont des formes qui réapparaissent dans des positions différentes après un certain nombre de générations de l'automate cellulaire) est équivalent à l'unité de base bit qui permet à l'information de circuler. Chaque nombre est une chaine de bits obtenue par un faisceau de planeurs générés par des canons à planeurs.
[Sap03] s'est basé sur cette notion pour rechercher des automates cellulaires universels. Un automate cellulaire est universel s'il accepte :
- des planeurs, qui se déplacent dans la grille et permettent la représentation et la transmission d'informations ;
- des canons à planeurs qui émettent des flux de planeurs.

[Sap03] ont proposé un couplage d'algorithmes évolutionnaires et d'automates cellulaires pour rechercher des automates cellulaires universels. Ils ont considéré un automate cellulaire binaire $2D$ et un système d'interaction de Moore de rayon $R = 1$.

**e) Extraction de la fonction de transition d'un automate cellulaire à partir d'un diagramme spatio-temporel :**

[II08] ont proposé une méthode pour extraire la fonction de transition d'un automate cellulaire $1D$ à partir d'un diagramme spatio-temporel (espace-temps) et de définir le rayon du système d'interaction $R$. Ils ont étudié deux automates cellulaires : un automate cellulaire déterministe et un automate cellulaire stochastique.

**f) Problème de génération de formes :**

* Pour étudier l'embryogénèse et les phénomènes de différenciation cellulaire, [DG+99] ont proposé d'utiliser un algorithme génétique pour générer des formes pleines représentant une colonie de cellules avec des automates cellulaires. L'algorithme génétique permet de déterminer la fonction de transition de l'automate cellulaire ainsi que les conditions correspondant à la mise à jour d'une cellule (direction, activation de la règle d'évolution locale).
* Chavoya et Duthen [CD06] ont proposé un modèle basé sur un automate cellulaire pour la génération de formes $2D$ pleines (carré, triangle, losange et cercle) à partir d'une cellule placée au centre de la grille. La fonction de transition ainsi que le nombre de générations de l'automate cellulaire sont déterminés par un algorithme génétique. Ce modèle est implémenté

sur un environnement programmable de modélisation NetLogo qui est basé sur des systèmes multi-agents. Cet environnement est utilisé pour modéliser et simuler certains phénomènes naturels et sociaux.
*Breukelaar et Back [BB05] se sont également intéressés à la génération de formes $2D$ et $3D$ par automate cellulaire. Parmi les formes à générer, ils ont traité le problème du damier. Ils ont proposé d'utiliser un algorithme génétique pour déterminer la fonction de transition de l'automate.

g)Dans [APV05], les auteurs ont effectué l'évolution d'un automate cellulaire par algorithme génétique pour la prédiction de séquences binaires complexes.

### 2.5.3 Approches utilisées dans la littérature

Dans la littérature, nous avons recensé peu de travaux qui traitent le problème inverse des automates cellulaires, particulièrement en traitement d'images.

La plupart des travaux de la littérature présentés utilisent les algorithmes évolutionnaires pour la recherche de la fonction de transition de l'automate cellulaire, particulièrement l'algorithme génétique.

A notre connaissance, aucune solution basée sur le PSO, sa variante quantique (QPSO) ou génétique quantique (QGA) n'a été étudiée.

### 2.5.4 Les Algorithmes Evolutionnaires pour Résoudre l'Emergence Inversée

Les algorithmes évolutionnaires en tant que métaheuristiques, sont utilisés majoritairement dans le domaine de l'optimisation pour résoudre des problèmes inverses. Il s'agit d'une classe de problèmes, où l'on peut évaluer une solution pour dire qu'elle est bonne ou mauvaise, sans pour autant avoir de méthodes directes conduisant à la construction d'un ensemble de paramètres permettant d'y aboutir. Ces algorithmes permettent de trouver des paramètres adaptés, en partant de solutions qui sont générées aléatoirement à l'initialisation de l'algorithme. Mis à part la définition des paramètres que l'on souhaite optimiser et l'ordre dans lequel ils apparaissent sur le chromosome, les connaissances sur le problème, nécessaires à la définition d'un algorithme évolutionnaire, sont peu nombreuses. L'un des intérêts des algorithmes évolution-

naires pour l'optimisation est que le fonctionnement basé sur une population, c'est-à-dire un ensemble de solutions, a la capacité de couvrir l'ensemble des paramètres dans l'espace du problème sans le faire réellement. Ensuite, comparer l'ensemble des solutions et choisir celles qui donnent le meilleur résultat. Un autre intérêt des algorithmes évolutionnistes c'est que la solution est optimisée dans sa totalité représentant l'ensemble des paramètres et les relations qui peuvent exister entre eux.

## 2.6 Conclusion

Dans ce chapitre, nous avons présenté les automates cellulaires. Ce sont des outils très utilisés pour simuler des systèmes dynamiques complexes. Ils sont basés sur le principe d'auto-organisation qui est une caractéristique de plusieurs systèmes biologiques. Ils permettent de faire émerger des comportements complexes à partir de règles locales simples.

Un état de l'art des problèmes inverses des automates cellulaire a été également présenté dans ce chapitre. Ces problèmes partagent l'objectif de déterminer les règles d'évolution locales (fonction de transition) permettant d'aboutir au comportement global souhaité. La plupart de ces travaux proposent de résoudre ce problème par des algorithmes évolutionnaires, particulièrement l'algorithme génétique.

Parmi ces travaux, nous nous intéressons en particulier aux problèmes de traitement d'images par automate cellulaire. Les travaux présentés dans la littérature pour la résolution de problèmes de traitement d'images (filtrage, segmentation)utilisent des méthodes d'optimisation pour chercher la fonction de transition de l'automate cellulaire. Les solutions proposées sont basées sur des algorithmes évolutionnaires, en majorité l'algorithme génétique.

Dans cette thèse, nous proposons de résoudre ces problèmes par des méthodes d'optimisation à base de métaheuristiques et leurs variantes quantiques. Ces méthodes seront présentées dans les chapitres suivants.

Chapitre 3

# Métaheuristiques d'Optimisation : Etat de l'Art

## 3.1 Introduction

Les problèmes d'optimisation occupent actuellement une place grandissante dans la communauté scientifique. Ces problèmes peuvent être combinatoires (discrets) ou à variables continues, avec un seul ou plusieurs objectifs (optimisation mono ou multi-objectif), statiques ou dynamiques, avec ou sans contraintes. Cette liste n'est pas exhaustive et un problème peut être, par exemple, à la fois continu et dynamique.
Un problème d'optimisation est défini par un ensemble de variables, une fonction objectif (ou fonction de coût) et un ensemble de contraintes. L'espace de recherche est l'ensemble des solutions possibles du problème. Il possède une dimension pour chaque variable. Pour des raisons pratiques et de temps de calcul, l'espace de recherche des méthodes de résolution est en général fini. Cette dernière limitation n'est pas gênante, puisqu'en général le décideur précise exactement le domaine de définition de chaque variable. La fonction objectif définit le but à atteindre, on cherche à minimiser ou à maximiser celle-ci. L'ensemble des contraintes est en général un ensemble d'égalités et d'inégalités que les variables doivent satisfaire. Ces contraintes limitent l'espace de recherche.
Les méthodes d'optimisation recherchent une solution, ou un ensemble de solutions, dans l'espace de recherche, qui satisfont l'ensemble des contraintes et qui minimisent, ou maximisent, la fonction objectif. Parmi ces méthodes, les méta-heuristiques sont des algorithmes génériques d'optimisation : leur but est de permettre la résolution d'une large gamme de problèmes différents, sans nécessiter de changements profonds dans l'algorithme. Elles forment une famille d'algorithmes visant à résoudre des problèmes d'optimisation difficile, pour lesquels on ne connaît pas de méthode classique plus efficace. Les méta-heuristiques s'inspirent généralement d'analogies avec la physique (recuit simulé), avec la biologie (algorithmes évolutionnaires) ou encore l'éthologie (colonies de fourmis, essaims particulaires). Toutes sortes d'extensions ont

été proposées pour ces algorithmes, notamment en optimisation dynamique.

## 3.2 Les Algorithmes d'Optimisation Approchée

### 3.2.1 Heuristiques

Une heuristique d'optimisation est une méthode expérimentale se voulant simple, rapide et adaptée à un problème donné. Sa capacité à optimiser un problème avec un minimum d'informations est contrebalancée par le fait qu'elle n'offre aucune garantie quant à l'optimalité de la meilleure solution trouvée. Du point de vue de la recherche opérationnelle, ce défaut n'est pas toujours un problème, tout spécialement quand seule une approximation de la solution optimale est recherchée.

### 3.2.2 Métaheuristiques

Parmi les heuristiques, certaines sont adaptables à un grand nombre de problèmes différents sans changements majeurs dans l'algorithme, on parle alors de méta-heuristiques. La plupart des heuristiques et des métaheuristiques utilisent des processus aléatoires comme moyens de récolter de l'information et de faire face à des problèmes comme l'explosion combinatoire. En plus de cette base stochastique, les métaheuristiques sont généralement itératives, c'est-à-dire qu'un même schéma de recherche est appliqué plusieurs fois au cours de l'optimisation, et directes, c'est-à-dire qu'elles n'utilisent pas l'information du gradient de la fonction objectif. Elles tirent en particulier leur intérêt de leur capacité à éviter les optima locaux, soit en acceptant une dégradation de la fonction objectif au cours de leur progression, soit en utilisant une population de points comme méthode de recherche (se démarquant ainsi des heuristiques de descente locale).
Souvent inspirées d'analogies avec la réalité (physique, biologie, éthologie, . . . ), elles sont généralement conçues au départ pour des problèmes discrets, mais peuvent faire l'objet d'adaptations pour des problèmes continus.
Les métaheuristiques, du fait de leur capacité à être utilisées sur un grand nombre de problèmes différents, se prêtent facilement à des extensions. Pour illustrer cette caractéristique, citons notamment :
- l'optimisation multiobjectif (dite aussi multicritère), où il faut optimiser plusieurs objectifs contradictoires. La recherche vise alors non pas à trouver un optimum global, mais un ensemble d'optima "au sens de Pareto" formant la "surface de compromis" du problème.
- l'optimisation multimodale, où l'on cherche un ensemble des meilleurs optima globaux et/ou

locaux.
- l'optimisation de problèmes bruités, où il existe une incertitude sur le calcul de la fonction objectif. Incertitude dont il faut alors tenir compte dans la recherche de l'optimum.
- l'optimisation dynamique, où la fonction objectif varie dans le temps. Il faut alors approcher au mieux l'optimum à chaque pas de temps.
- la parallélisation, où l'on cherche à accélérer la vitesse de l'optimisation en répartissant la charge de calcul sur des unités fonctionnant en parallèle. Le problème revient alors à adapter les métaheuristiques pour qu'elle soient distribuées.
- l'hybridation, qui vise à tirer parti des avantages respectifs de métaheuristiques différentes en les combinant.

Enfin, la grande vitalité de ce domaine de recherche ne doit pas faire oublier qu'un des intérêts majeurs des métaheuristiques est leur facilité d'utilisation dans des problèmes concrets. L'utilisateur est généralement demandeur de méthodes efficaces permettant d'atteindre un optimum avec une précision acceptable dans un temps raisonnable. Un des enjeux de la conception des métaheuristiques est donc de faciliter le choix d'une méthode et de simplifier son réglage pour l'adapter à un problème donné.

Du fait du foisonnement de la recherche, un grand nombre de classes de métaheuristiques existent. Les principales seront présentées plus en détail dans la suite de ce chapitre. On peut distinguer les métaheuristiques qui font évoluer une seule solution sur l'espace de recherche à chaque itération et les métaheuristiques à base de population de solutions. En général, les métaheuristiques à base de solution unique sont plutôt axées sur l'exploitation de l'espace de recherche, on n'est donc jamais sûr d'obtenir l'optimum optimorum. Les métaheuristiques à base de population sont plutôt exploratoires et permettent une meilleure diversification de l'espace de recherche.

### 3.2.3 Les métaheuristiques à solution unique

Dans cette section, nous présentons les métaheuristiques à base de solution unique, aussi appelées méthodes de trajectoire. Contrairement aux métaheuristiques à base de population, les métaheuristiques à solution unique commencent avec une seule solution initiale et s'en éloignent progressivement, en construisant une trajectoire dans l'espace de recherche.

Les méthodes de trajectoire englobent essentiellement la méthode de descente, la méthode du recuit simulé, la recherche tabou, la méthode GRASP, la recherche à voisinage variable, la recherche locale itérée, et leurs variantes.

#### 3.2.3.1 Le Recuit Simulé

L'origine de la méthode du recuit simulé vient de l'analogie avec la métallurgie, où la méthode, pour atteindre des états de basse énergie d'un solide, consiste à monter la température du solide à des valeurs élevées, puis à le laisser refroidir lentement. Ce processus est appelé "recuit".
L'algorithme d'optimisation appelé "recuit simulé" a été établi indépendamment par Kirkpatrick et al. en 1983 [KGV+83], et Cerny en 1985 [Čer85]. L'idée de base est la suivante : à des paliers de températures décroissantes, l'algorithme utilise la procédure itérative de Metropolis [MRR+53], pour atteindre un état de quasi-équilibre thermodynamique. Cette procédure permet de sortir des minima locaux avec une probabilité d'autant plus grande que la température est élevée. Quand l'algorithme atteint les très basses températures, les états les plus probables constituent en principe d'excellentes solutions au problème d'optimisation.
Le recuit a connu un grand succès dans différents domaines d'application [CL99], grâce a ses deux atouts principaux : un comportement type boite noire et une facilité de "réglage" des paramètres internes. D'un point de vue théorique, le recuit simulé permet d'approcher de prés la solution optimale du problème plus rapidement qu'une exploration exhaustive dans l'espace de recherche. En pratique, un bon réglage des paramètres internes de l'algorithme permet d'accélérer la convergence vers une solution pseudo-optimale, avec une précision prédéfinie. Le recuit simulé est aussi adapté pour résoudre les problèmes d'optimisation continue [SBDH97]. L'inconvénient majeur de cet algorithme est sa lenteur. Toutefois, plusieurs tentatives de parallélisation de l'algorithme ont été proposées dans la littérature, au détriment de sa convergence théorique. Il est à signaler qu'il existe des approches rapides qui sont simples à mettre en oeuvre qui conservent les propriétés de convergence, comme par exemple dans [RRD90], où le recuit simulé est réparti sur plusieurs calculateurs en parallèle.

**Data**:
**Output**:

1. $s := s_0$
2. $e := E(s)$
3. $k := 0$
4. **while** $k < k_{max}$ *et* $e > e_{max}$ **do**
   - 4.1. $s_n := voisin(s)$
   - 4.2. $e_n = E(sn)$
   - **if** $e_n < e$ *ou* $alatoire() < P(e_n - e, temp(k/k_{max}))$ **then**
     - $s := s_n; e := e_n$
   - **end**
   - $k := k + 1$

**end**
Retourner s

**Algorithm 2:** La méthode du recuit simulé

**Data**:
**Output**:

1. Choisir $s \in S$
2. Choisir $\lambda \in E$
3. Calcul de la variation d'énergie $\Delta E$ lors du passage de la valeur en s de $x_n$ à $\lambda$
4. **if** $\Delta E \leq 0$ **then**
   | le changement est accepté : $x_n = \lambda$
**end**
**else**
   | le changement est accepté avec la probabilité $p = exp(-\Delta E/T)$
**end**

**Algorithm 3:** L'algorithme de Metropolis

#### 3.2.3.2 L'Algorithme de Recherche Tabou

La recherche tabou est une métaheuristique itérative qualifiée de « recherche locale » au sens large. L'idée de la recherche tabou consiste, à partir d'une position donnée, à en explorer le voisinage et à choisir le voisin qui minimise la fonction objectif. L'algorithme de recherche tabou a été introduit par Fred Glover en 1986 [Glo86]. Le principe de base de cet algorithme est de pouvoir poursuivre la recherche de solutions même lorsqu'un optimum local est rencontré et ce, en permettant des déplacements qui n'améliorent pas la solution, et en utilisant le principe de la mémoire pour éviter les retours en arrière (mouvements cycliques). Il est essentiel de noter que cette opération peut conduire à augmenter la valeur de la fonction (dans un problème de minimisation) ; c'est le cas lorsque tous les points du voisinage ont une valeur plus élevée. C'est à partir de ce mécanisme que l'on sort d'un minimum local. En effet, comme l'algorithme de recuit simulé, la méthode de recherche tabou fonctionne avec une seule configuration courante, qui est actualisée au cours des itérations successives. La nouveauté ici est que, pour éviter le risque de retour à une configuration déjà visitée, on tient à jour une liste de mouvements interdits (ou de solutions interdites), appelée « liste tabou ». Le rôle de cette dernière évolue au cours de la résolution pour passer de l'exploration (aussi appelée « diversification ») à l'exploitation (également appelée « intensification »). Cette liste contient m mouvements $(t \rightarrow s)$ qui sont les inverses des m derniers mouvements $(s \rightarrow t)$ effectués. L'algorithme modélise ainsi une forme primaire de mémoire à court terme. L'algorithme de recherche tabou peut être résumé par l'algorithme 4.

Dans sa forme de base, l'algorithme de recherche tabou présente l'avantage de comporter moins de paramètres que l'algorithme de recuit simulé. Cependant, l'algorithme n'étant pas toujours performant, il est souvent approprié de lui ajouter des processus d'intensification et/ou de diversification, qui introduisent de nouveaux paramètres de contrôle [GL97].

**Data**:
**Output**:

1. Déterminer une configuration aléatoire s
2. Initialiser une liste tabou vide
3. **while** *(le critère d'arrêt n'est pas satisfait)* **do**
   4. Perturbation de s suivant N mouvements non tabous
   5. Evaluation des N voisins
   6. Sélection du meilleur voisin t
   7. Actualisation de la meilleure position connue $s^*$
   8. Insertion du mouvement $t \rightarrow s$ dans la liste tabou
   9. s = t

**end**

**Algorithm 4:** Algorithme de recherche tabou

### 3.2.4 Les métaheuristiques à population de solutions

Contrairement aux algorithmes partant d'une solution singulière, les métaheuristiques à population de solutions améliorent, au fur et à mesure des itérations, une population de solutions. On distingue dans cette catégorie, les algorithmes évolutionnaires, qui sont une famille d'algorithmes issus de la théorie de l'évolution par la sélection naturelle, énoncée par Charles Darwin [Dar59] et les algorithmes d'intelligence en essaim qui, de la même manière que les algorithmes évolutionnaires, proviennent d'analogies avec des phénomènes biologiques naturels.

#### 3.2.4.1 Les algorithmes évolutionnaires

Les algorithmes évolutionnistes ou algorithmes évolutionnaires (EC : Evolutionary Computation), sont une famille d'algorithmes s'inspirant de la théorie de l'évolution « darwinienne » pour résoudre des problèmes divers. Selon la théorie du naturaliste Charles Darwin, énoncée en 1859 [Dar59], l'évolution des espèces est la conséquence de la conjonction de deux phénomènes : d'une part la sélection naturelle qui favorise les individus les plus adaptés à leur milieu à survivre et à se reproduire, laissant une descendance qui transmettra leurs gènes et d'autre part, la présence de variations non dirigées parmi les traits génétiques des espèces (mutations). Le terme Evolutionary Computation englobe une classe assez large de métaheuristiques telles que les algorithmes génétiques [Hol75], les stratégies d'évolution [Rec73], la programmation évolutive [FOW66], et la programmation génétique [Koz94]. La figure 3.1 décrit le squelette d'un algorithme évolutionnaire type, commun à la plupart des instances classiques d'EAs. Chaque itération de l'algorithme correspond à une génération, où une population constituée de plusieurs individus, représentant des solutions potentielles du problème considéré, est capable de se reproduire. Elle est sujette à des variations génétiques et à la pression de l'environnement qui est simulée à l'aide de la fonction d'adaptation, ce qui provoque la sélection naturelle (la survie du plus fort). Les opérateurs de variation sont appliqués (avec une probabilité donnée) aux individus parents sélectionnés, ce qui génère de nouveaux descendants appelés enfants

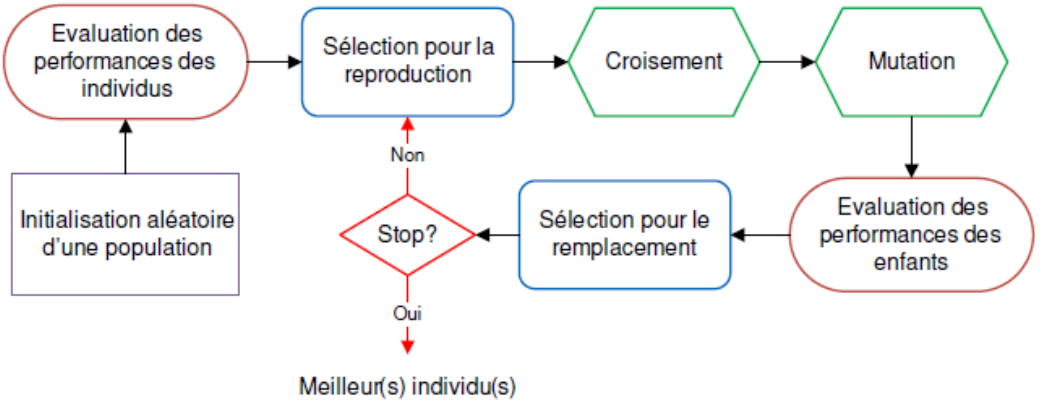


FIGURE 3.1 – Principe d'un algorithme évolutionnaire (EA)[DPST03]

(ou offsprings) ; on parlera de mutation pour les opérateurs unaires, et de croisement pour les opérateurs binaires (ou n-aires). Les individus issus de ces opérations sont alors insérés dans la population. Le processus d'évolution est itéré, de génération en génération, jusqu'à ce qu'une condition d'arrêt soit vérifiée, par exemple, quand un nombre maximum de générations ou un nombre maximum d'évaluations est atteint.
Parmi les algorithmes évolutionnaires cités, on va détailler les algorithmes génétiques.

- **Les Algorithmes Génétiques**

Les algorithmes génétiques (GA : Genetic Algorithms) sont l'une des techniques les plus populaires des EAs. Les origines de ces algorithmes remontent au début des années 1970, avec les travaux de John Holland et ses élèves à l'Université du Michigan sur les systèmes adaptatifs [Hol75]. L'ouvrage de référence de David E.Goldberg [Gol89] a fortement participé à leur essor. Ces algorithmes se détachent en grande partie par la représentation des données du génotype, initialement sous forme d'un vecteur binaire et plus généralement sous forme d'une chaîne de caractères. Chaque étape de GA est associée à un opérateur décrivant la façon de manipuler les individus :
- *Sélection* : Pour déterminer quels individus sont plus enclins à se reproduire, une sélection est opérée. Il existe plusieurs techniques de sélection, les principales utilisées sont la sélection par tirage à la roulette (roulette-wheel selection), la sélection par tournoi (tournament selection), la sélection par rang (ranking selection)...etc.
- *Croisement* : L'opérateur de croisement combine les caractéristiques d'un ensemble d'individus parents (généralement deux) préalablement sélectionnés, et généré de nouveaux individus enfants. La encore, il existe de nombreux opérateurs de croisement, par exemple le croisement en un point, le croisement en n-points ($n \geq 2$) et le croisement uniforme (Fig. 3.2).
- *Mutation* : Les descendants sont mutés, c'est-à-dire que l'on modifie aléatoirement une partie de leur génotype, selon l'opérateur de mutation.
- *Remplacement* : Le remplacement (ou sélection des survivants), comme son nom l'indique,

remplace certains des parents par certains des descendants. Le plus simple est de prendre les meilleurs individus de la population, en fonction de leurs performances respectives, afin de former une nouvelle population (typiquement de la même taille qu'au début de l'itération).

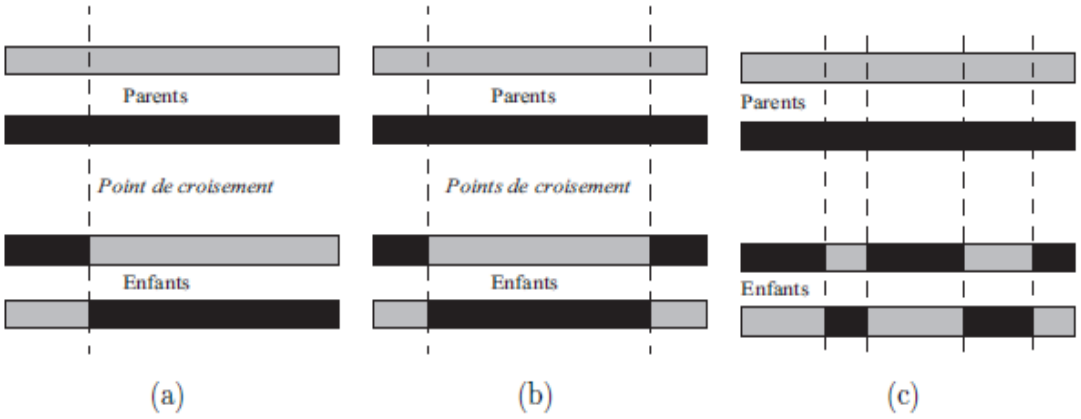


FIGURE 3.2 – Exemples de croisement : (a) croisement simple en un point (b)croisement en deux points (c) croisement uniforme

#### 3.2.4.2 Les Algorithmes des Colonies de Fourmis

En plus des sociétés animales, les sociétés d'insectes telles que les fourmis, les termites, les araignées et certaines espèces d'abeilles et de guêpes, constitues-elles aussi des sources d'inspiration pour le développement de systèmes artificiels « Les sociétés d'insectes nous proposent un modèle de fonctionnement bien différent du modèle humain : un modèle décentralisé, fondé sur la coopération d'unités autonomes au comportement relativement simple et probabiliste, qui sont distribuées dans l'environnement et ne disposent que d'informations locales. » [DGF+91]. L'exemple des fourmis est répandu dans la littérature à cause de leur capacité à réaliser des tâches hautement complexes à partir des interactions d'insectes simples à l'intelligence très rudimentaire. [DAGP90] ; [DMC96] ; [BT97] ; [Par97] ; [BDT99].

En particulier deux comportements collectifs ont été principalement étudiés chez les fourmis : l'optimisation de chemin et le tri des éléments du couvain. Le premier comportement met en évidence la capacité des fourmis à optimiser la procédure de fourragement. En effet, au départ les fourmis se déplacent d'une manière aléatoire de la fourmilière vers une source de nourriture en laissant des traces de phéromones sur le chemin qu'elles empruntent. Cette trace tend à attirer les congénères qui, en la suivant, vont parvenir à la nourriture. Il a été constaté alors que les fourmis qui choisissent l'itinéraire le plus court, le terminent le plus vite. Les fourmis vont alors préférer progressivement cet itinéraire car il est marqué avec une plus

FIGURE 3.3 – Une colonie de fourmis ramenant la nourriture vers le nid

grande quantité de phéromones [GADP89]. De plus, l'évaporation naturelle de la phéromone renforce davantage ce choix des fourmis par l'affaiblissement des chemins plus longs. L'illustration de ce phénomène est donnée par la figure 3.4. Le second comportement collectif des

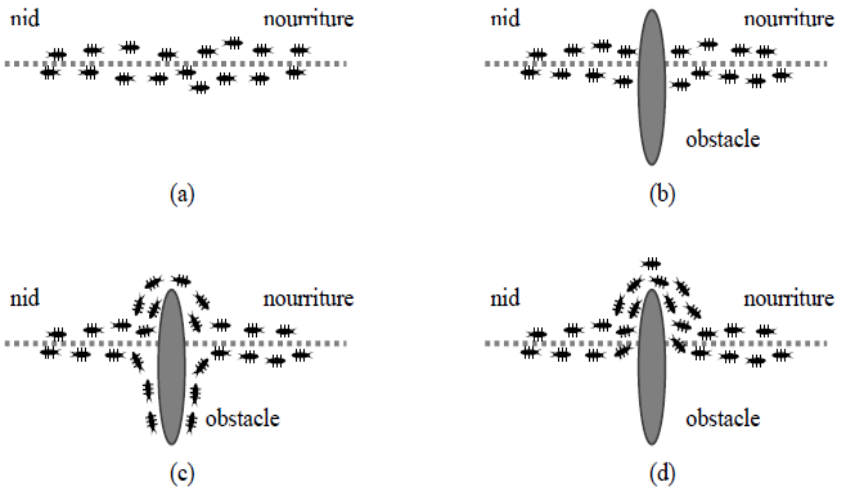


FIGURE 3.4 – Illustration de la capacité des fourmis à chercher de la nourriture en minimisant leur parcours. (a)Recherche sans obstacle, (b)Apparition d'un obstacle, (c)Recherche du chemin optimal, (d)Chemin optimal trouvé

fourmis concerne leur aptitude à nettoyer leur nid en organisant collectivement des cimetières composés de cadavres empilés les uns sur les autres. Le principe est le suivant : plus un cadavre est isolé, plus la fourmi a de chances de ramasser ce cadavre. La probabilité pour une fourmi porteuse de déposer ce qu'elle transporte suit une règle inverse : plus le monticule observé est important, plus la probabilité de déposer le corps au sol sera grande.

Les premières modélisations algorithmiques de ces comportements collectifs et auto-organisés sont dues à Deneubourg et son équipe [DAGP90] et furent repris par d'autres chercheurs pour donner naissance à différents algorithmes pour la résolution de problèmes par stigmergie.

#### 3.2.4.3 L'algorithme des colonies d'abeilles artificielles (ABC)

C'est une méthode d'optimisation inspirée du comportement intelligent des abeilles lors de la récolte du nectar. Il existe trois types d'abeilles : les ouvrières, les faux-bourdons et une seule reine. La communication entre les abeilles se fait via une danse, pour transmettre des informations sur les sources de nourriture. Il y'a deux types de danse : la danse en rond signifie que le pollen est à faible distance, la danse frétillante signifie que le pollen est à moins de 10 km.
L'algorithme suivant illustre le principe de ABC [Kar10]

1. Initialiser la population avec n solutions aléatoires
2. Evaluer la fitness de la population
3. **while** *(le critère d'arrêt n'est pas atteint faire)* **do**
   - 3.1. Recruter des abeilles ⇒ rechercher de nouvelles sources de nourriture
   - 3.2. Evaluer la fitness de la population
   - 3.3. SI un membre de la population ne s'est pas amélioré
   - 3.4. faire enregistrer la solution et la remplacer par une solution aléatoire
   - 3.5. Trouver les solutions aléatoires et remplacer les membres de la population qui ont la mauvaise fitness

   **end**
4. Retourner la meilleure solution.

**Algorithm 5:** Principe de l'algorithme des abeilles artificielles

#### 3.2.4.4 Les systèmes immunitaires artificiels (AIS)

AIS cherche à modéliser et à appliquer des principes d'immunologie pour résoudre des problèmes d'optimisation. Le système immunitaire d'un organisme est un système biologique robuste et adaptatif, qui défend le corps contre les « agressions » d'organismes extérieurs. La métaphore dont sont issus les algorithmes AIS met l'accent sur les aspects d'apprentissage et de mémoire du système immunitaire dit adaptatif (par opposition au système dit inné), notamment via la discrimination entre le soi et le non-soi [DPST03]. En effet, les cellules vivantes disposent sur leurs membranes de molécules spécifiques dites antigènes. Chaque organisme possède ainsi une identité unique, déterminée par l'ensemble des antigènes présents sur ses cellules. Certaines cellules du système immunitaire, appelées lymphocytes, possèdent des récepteurs capables de se lier spécifiquement à un antigène donné, permettant ainsi de reconnaître une cellule étrangère à l'organisme. Un lymphocyte ayant reconnu une cellule du non-soi va être incité à proliférer (en produisant des clones de lui-même) et à se différencier en cellule permettant de garder en mémoire l'antigène, ou en cellule permettant de combattre les agressions. Dans le premier cas, son clonage permettra à l'organisme de réagir plus vite à une nouvelle exposition à l'antigène (c'est le principe de la vaccination). A noter qu'un mécanisme d'hyper-mutation des cellules clonées assure la diversité des récepteurs dans l'ensemble de la population des lymphocytes. Dans le second cas, le combat contre les agressions est possible grâce à la production d'anticorps. Ces étapes sont résumées dans la figure 3.5.
AIS s'inspire essentiellement des sélections opérées sur les lymphocytes, et des processus per-

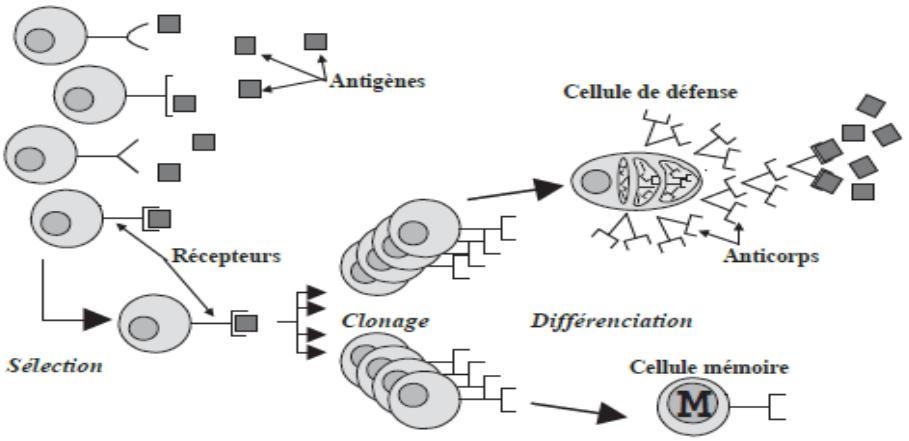


FIGURE 3.5 – La sélection par clonage [DPST03]

mettant la multiplication et la mémoire du système. En effet, ces caractéristiques sont capitales pour maintenir les propriétés auto-organisées du système. Cette approche est assez similaire à celle des algorithmes évolutionnaires. Elle peut également être comparée à celle des réseaux de neurones. Dans le cadre de l'optimisation, on peut considérer les AIS comme une forme d'algorithme évolutionnaire présentant des opérateurs particuliers. Par exemple, l'opération de sélection est basée sur une mesure d'affinité (i.e. entre le récepteur d'un lymphocyte et un antigène). La mutation s'opère, quant à elle, via un opérateur d'hyper-mutation directement issu de la métaphore.
Un exemple d'AIS basique est présenté dans l'algorithme 6, où t désigne l'itération courante. Plus de détails sur AIS sont donnés dans [DCT02].

1. t $\leftarrow$ 1
2. Initialiser aléatoirement une population $P(t)$ d'individus, composée de $m$ cellules mémoires $P_m(t)$ et de $r$ autres cellules $P_r(t) : P(t) = P_m(t) \cup P_r(t)$
3. Evaluer l'affinité de chaque individu de $P(t)$
4. **while** *(le critère d'arrêt n'est pas atteint faire)* **do**
   - 4.1. Sélectionner un sous-ensemble $S(t)$ d'individus de plus haute affinité dans $P(t)$
   - 4.2. Cloner chaque individu de $S(t)$ proportionnellement à son affinité, pour former une population $C(t)$
   - 4.3. Muter chaque individu de $C(t)$ avec un taux inversement proportionnel à son affinité, pour former une population $C^*(t)$
   - 4.4. Evaluer l'affinité de chaque individu de $C^*(t)$
   - 4.5. Sélectionner les $m$ individus de plus haute affinité dans $C^*(t) \cup P_m(t)$ pour former $P_m(t+1)$
   - 4.6. Remplacer un sous-ensemble d'individus de plus faible affinité dans $P_r(t)$ par des nouveaux individus, tirés aléatoirement, pour former $P_r(t+1)$
   - 4.7. Evaluer l'affinité de ces nouveaux individus
   - 4.8. t $\leftarrow$ t+1

**end**

**Algorithm 6:** Exemple d'algorithme AIS basique

#### 3.2.4.5 Algorithme à évolution différentielle

L'algorithme à évolution différentielle (DE : Differential Evolution) a été proposé par R. Storn et K. Price dans les années 1990 [SP97] afin de résoudre le problème d'ajustement par polynômes de Tchebychev (Chebyshev polynomial fitting problem).
Différentes variantes de DE ont été suggérées par Price et al. [PSL06] et sont classiquement appelées *DE/x/y/z*, où DE désigne l'évolution différentielle, x fait référence au mode de sélection de l'individu de référence ou l'individu cible (rand ou best) pour la mutation, y est le nombre de différenciations utilisées pour la perturbation du vecteur cible x, z désigne le schéma de croisement, qui peut être binomial ou exponentiel. À chaque itération du processus d'évolution, chaque individu est d'abord muté, puis croisé avec son mutant. La phase de mutation implique que, pour chaque individu de la population $\overrightarrow{X}_i$ (target individual), un nouvel individu $\overrightarrow{V}_i$ (mutant individual) est généré, en ajoutant à ses composantes la différence pondérée d'autres individus pris aléatoirement dans la population. Le vecteur d'essai $\overrightarrow{U}_i$ (trial individual) est ensuite généré après croisement entre l'individu initial $\overrightarrow{X}_i$ et son mutant $\overrightarrow{V}_i$. La phase de sélection intervient juste après, par compétition entre l'individu père $\overrightarrow{X}_i$ et son descendant $\overrightarrow{U}_i$ , le meilleur étant conservé pour la génération suivante. Ce processus est répété pour chaque individu de la population initiale, et mène donc à la création d'une nouvelle population de taille identique.

#### 3.2.4.6 Les algorithmes à estimation de distribution

Les algorithmes à estimation de distribution (EDA : Estimation of Distribution Algorithms), aussi appelés algorithmes génétiques par construction de modèles probabilistes (PMBGA : Probabilistic Model-Building Genetic Algorithms), ont été proposés en 1996 par Mühlenbein et Paaß [MP96].Ils ont une base commune avec les algorithmes évolutionnaires, mais s'en distinguent, car le coeur de la méthode consiste à estimer les relations entre les différentes variables du problème. Ils utilisent pour cela une estimation de la distribution de probabilité, associée à chaque point de l'échantillon. Ils n'emploient donc pas d'opérateurs de croisement ou de mutation, l'échantillon étant directement construit à partir des paramètres de distribution, estimés à l'itération précédente.
Les EDAs peuvent être divisés en trois classes, selon la complexité des modèles probabilistes utilisés pour évaluer les dépendances entre variables [LL01] : (1) les modèles sans dépendance (Univariate EDAs) ; (2) les modèles à dépendances bivariantes (Bivariate EDAs) ; et (3) les modèles à dépendances multiples (Multivariate EDAs).

## 3.3 L'Optimisation par Essaim de Particules

L'optimisation par essaim particulaire (OEP), ou Particle Swarm Optimization (PSO), est un algorithme évolutionnaire qui utilise une population de solutions candidates pour développer une solution optimale au problème. Cet algorithme a été proposé par Russel Eberhart (ingénieur en électricité) et James Kennedy (socio-psychologue) en 1995 [KE95]. Il s'inspire à l'origine du monde du vivant, plus précisément du comportement social des animaux évoluant en essaim, tels que les bancs de poissons et les vols groupés d'oiseaux. L'optimisation par essaims particulaires est classée dans les techniques d'optimisation stochastiques à population. Une description détaillée de l'algorithme PSO est présentée dans les paragraphes suivants.

### 3.3.1 Principe

PSO est initialisé avec un groupe de particules aléatoires (solutions) and effectue une recherche d'optimum par mise à jour de chaque génération. A chaque itération, une particule est mise à jour par les deux meilleures valeurs suivantes. La première est la meilleure solution locale qu'une particule a obtenu jusqu'à présent. Cette valeur est appelée *personal best solution*. La deuxième meilleure solution est celle que l'essaim en entier a obtenu jusqu'à présent. Cette valeur est appelée *global best solution*. La philosophie du PSO original est l'apprentissage par les expériences des individus (*personal best solution*) et la meilleure expérience globale (*global best solution*) dans l'essaim entier. Notons par N le nombre de particules dans l'essaim.

Soit $X_i(t) = (x_{i1}(t), ..., x_{id}(t), ..., x_{iD}(t))$ une particule avec $D$ bits à l'itération $t$, considérée comme une solution potentielle. Notons la vitesse $V_{i1}(t) = (v_{i1}(t), ..., v_{id}(t), ..., v_{iD}(t))$

Soit $PBest_i(t) = (PBest_{i1}(t), ..., PBest_{id}(t), ..., PBest_{iD}(t))$ la meilleure solution que la particule $i$ a obtenu jusqu'à l'itération $t$, et $GBest(t) = (GBest_1(t), ..., GBest_d(t), ..., GBest_D(t))$ la meilleure solution obtenue de $PBest_i(t)$ dans l'essaim entier à l'itération $t$. Chaque particule ajuste sa vitesse en accord avec ses vitesses précédentes, la partie cognitive et la partie sociale. Les équations suivantes décrivent le mouvement des particules : [KE95] :

$$v_{id}(t+1) = v_{id}(t) + c_1 * r_1 * (pbest_{id}(t) - x_{id}(t)) + c_2 * r_2 * (gbest_d(t) - x_{id}(t)) \quad (3.1)$$

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t+1) \quad (3.2)$$

où $c_1$ est la composante cognitive et $c_2$ la composante sociale ; $r_1$ et $r_2$ sont des nombres aléatoires uniformément distribués sur [0,1].

Dans [CK02], Clerc et Kennedy analysent la trajectoire et prouvent que chaque particule $i$ dans le système PSO converge vers un point attracteur local $g_i$, dont les coordonnées sont : $g_{id} = \phi * pbest_{id} + (1-\phi) * gbest_d$ , de sorte que les meilleures positions précédentes de toutes les particules vont converger vers une position globale exclusive avec $t \to \infty$, où $\phi$ est un nombre aléatoire uniformément distribué sur [0,1]. La collectivité des particules dans le PSO classique confine la recherche de la particule, en conséquence, il n'est pas permis à la particule d'apparaitre dans une position loin de l'essaim, qui pourrait être une solution ayant une meilleure fitness que l'actuelle (global best solution).

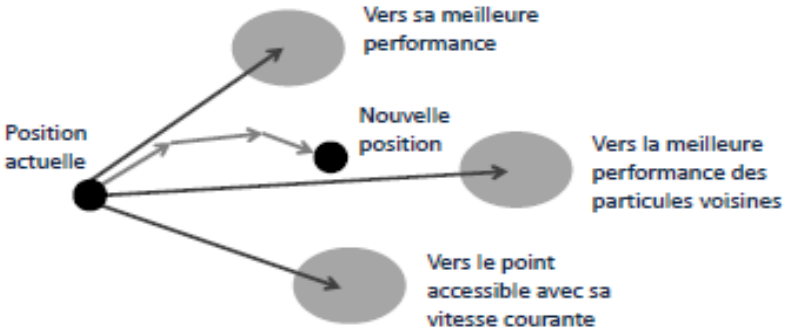


FIGURE 3.6 – Déplacement d'une particule

L'algorithme PSO est décrit par le pseudo-code 7

**Data**:
**Output**:

Répéter pour i=1 à taille-population faire
Si f(Xi) < f(Pi) Alors Pi = Xi ;
G = arg min (f(Pi)) ;
Pour j = 1 à D faire
mise à jour vitesse avec Eq. (3.1) ;
mise à jour position avec Eq. (3.2) ;
Fin//boucle j
Fin//boucle i
Jusqu'à critère d'arrêt atteint.

**Algorithm 7:** Pseudo-code de l'algorithme PSO

### 3.3.2 Comparaison entre l'algorithme génétique et le PSO

Le PSO partage beaucoup de points communs avec l'algorithme génétique (AG). Les deux algorithmes commencent avec un groupe de populations générées aléatoirement, tous les deux ont des valeurs de fonction objective pour évaluer la population. Les deux mettent à jour la population et cherchent de l'optimum avec des techniques aléatoires. Les deux systèmes ne garantissent pas le succès. Cependant, le PSO n'a pas d'opérateurs génétiques comme le croisement et la mutation. Les particules se mettent à jour avec la vitesse interne. Elles ont

également la mémoire, qui est importante pour l'algorithme. Conceptuellement, il y a quelques différences entre le PSO et l'AG : les particules restent vivantes et « habitent » l'espace de recherche pendant l'exécution, alors que dans les AG les individus sont remplacés à chaque génération. De plus, l'objectif est atteint par une recherche coopérative dans les essaims de particules plutôt qu'une recherche compétitive comme dans les AG.
L'algorithme PSO est très simple à implémenter et possède moins de paramètres à ajuster. Il est plus rapide que AG et fournit un bon compromis entre vitesse et efficacité. PSO ne nécessite pas d'opérateurs évolutifs comme la mutation et le croisement, car les solutions potentielles émergent simplement dans l'espace de recherche, en évoluant itérativement dans la direction des solutions optimales. Il a aussi été prouvé que PSO converge plus rapidement et fournit de meilleurs résultats.

### 3.3.3 Domaines d'utilisation du PSO

Le PSO a été largement utilisé pour résoudre des problèmes d'optimisation variés, comme la minimisation de fonctions [SE99], l'évolution de réseau neuronal [vdBE00], les problèmes d'optimisation multi-objectif [Eng06], le clustering d'images [OES05] PSO a aussi été utilisé pour résoudre des problèmes d'optimisation dans les systèmes de puissance électrique [AEH09], ordonnancement d'applications workflow dans un environnement de cloud computing [PWGB10]

L'algorithme PSO a plusieurs points communs avec les autres algorithmes évolutionnaires tels que le recuit-simulé, et les algorithmes génétiques, et a prouvé sa robustesse dans de nombreux cas d'optimisation difficile.

## 3.4 Conclusion

Dans ce chapitre, nous avons présenté un état de l'art sur les algorithmes s d'optimisation. Tout d'abord, nous avons donné la définition d'heuristiques et métaheuristiques. Puis, nous avons distingué les deux types de métaheuristiques : les métaheuristiques à solution unique et les métaheuristiques à population de solutions. Parmi les métaheuristiques présentées pour résoudre ces problèmes, un intérêt particulier a été porté à la méthode d'optimisation par essaim particulaire (PSO). Cette méthode, qui est inspirée du monde du vivant, a rencontré un succès remarquable depuis sa création, grâce à sa simplicité. Elle présente l'avantage d'être efficace sur une vaste gamme de problèmes, sans pour autant que l'utilisateur ait à modifier la structure de base de l'algorithme. Par la suite, et malgré le succès des différentes métaheu-

ristiques, il est devenu évident que ces méthodes ont des limites, lorsqu'il s'agit d'appréhender des problèmes d'optimisation particuliers, comme les problèmes d'optimisation du monde réel, et à large échelle. Une combinaison judicieuse entre les différentes métaheuristiques ou bien avec d'autres algorithmes d'optimisation, techniques de l'intelligence artificielle, informatique quantique, peut aboutir à des solutions meilleures et plus performantes. Le principal but derrière ces combinaisons est d'obtenir des systèmes plus performants, qui exploitent et incluent les avantages de ces algorithmes et techniques. Ces avantages sont complémentaires et la métaheuristique hybride résultante en bénéficie en terme de performances.
Dans ce contexte, et après avoir travaillé avec la métaheuristique PSO et obtenu de bons résultats, on a envisagé d'explorer le domaine de l'informatique quantique, et utiliser la version quantique du PSO pour résoudre le problème posé dans cette thèse. Dans le chapitre 4, nous introduisons l'informatique quantique.

Chapitre 4

# L'Informatique Quantique

## 4.1 Introduction

L'informatique quantique est un nouveau champs de recherche en informatique dont l'objectif est de tirer profit des possibilités offertes par la physique quantique. Elle se base sur l'utilisation des spécificités de la physique quantique pour le traitement de l'information. La particularité de cette nouvelle voie de recherche réside dans l'utilisation d'un nouveau type de support de stockage et de traitement de l'information : l'ordinateur quantique. Un ordinateur quantique est un nouveau type de machines caractérisé par sa puissance de calcul grâce aux possibilités de parallélisme offertes par les principes issus de la physique quantique.

Malgré une certaine maturité de la théorie de l'informatique quantique et l'apparition d'algorithmes quantiques remarquablement moins complexes que leurs contreparties classiques pour la résolution de problèmes, une difficulté cruciale pour l'adoption de ce nouveau modèle de traitement de l'information est l'absence d'ordinateurs quantiques puissants jusqu'à nos jours. Les modèles déjà construits ne sont que des modèle préliminaires et ils sont toujours à des phases expérimentales.

En attendant la réalisation d'ordinateurs quantiques puissants, deux axes de recherche ont vu le jour. Le premier vise à la simulation d'algorithmes quantiques sur des machines classiques. Cette solution réduit les performances des algorithmes quantiques et n'est efficace que pour leur étude. Le deuxième se base sur l'exploitation des principes de l'informatique quantique, tels que le bit quantique, la superposition d'états et les portes quantiques, par des algorithmes conventionnels afin d'augmenter les performances de ces derniers.

Dans ce chapitre, nous donnons une brève introduction à l'informatique quantique. Ensuite, nous décrivons les Algorithmes Evolutionnaires Quantiques (AEQ). Finalement, nous présentons une petite synthèse sur d'autres hybridations entre l'informatique quantique et les

approches d'optimisation.

## 4.2 L'Informatique Quantique

Dans cette section, on présente les principes de base de l'informatique quantique, on donne quelques exemples de ses domaines d'applications, et on présente ses avantages et ses limites.

### 4.2.1 Bit Quantique

Un "bit" représente l'unité élémentaire d'information dans un ordinateur classique. Il peut se trouver dans l'un de deux états, le 1 ou le 0, à l'instar d'un interrupteur, ouvert ou fermé. Appelée "qubit" (de l'anglais : QUantum BIT), la version quantique d'un bit d'information peut être dans les deux états en même temps. Ces états peuvent être représentés par une flèche, qui pointe une position à la surface d'une sphère comme montré dans la figure 4.1 [DE06] ; [Hey99] ; [NC10].

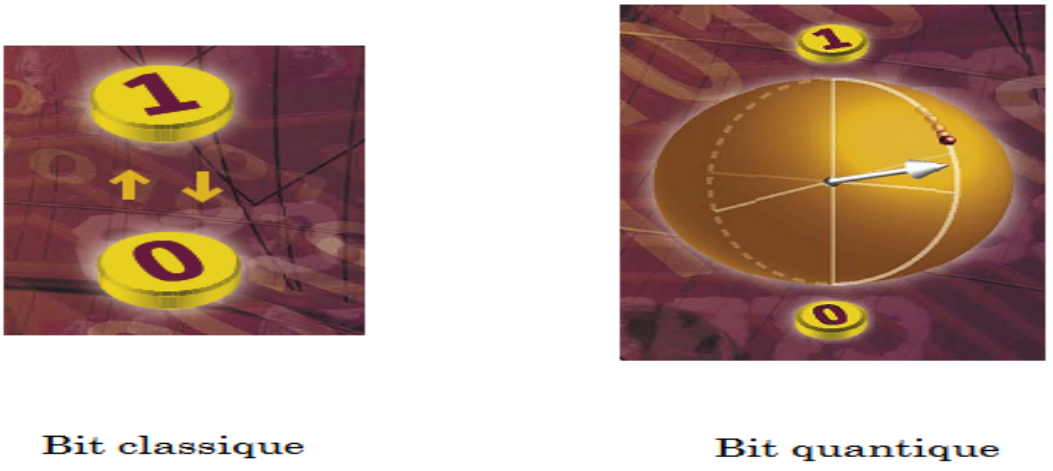


FIGURE 4.1 – Bit classique et bit quantique

Un qubit est un système physique basé sur deux valeurs logiques distinctes : 0 et 1, ou "oui" et "non". Ces deux valeurs représentent les états de base notés par $|0>$ et $|1>$.

L'état du système $\varphi$, qui est une superposition des deux états fondamentaux, est donné par l'équation (4.1) [RP00].

$$\varphi = \alpha|0> + \beta|1> \tag{4.1}$$

L'observation de cet état va donner la valeur 0 avec une probabilité égale à $|\alpha|^2$ et la valeur 1 avec une probabilité égale à $|\beta|^2$ où $\alpha$ et $\beta$ sont deux nombres complexes appelés amplitudes et vérifiant la contrainte donnée par l'équation 4.2.

$$|\alpha|^2 + |\beta|^2 = 1 \tag{4.2}$$

### 4.2.2 Registre quantique

Un registre quantique est un ensemble de qubits, une superposition arbitraire de $n$ qubits. L'état d'un registre quantique peut être donné par l'équation (4.3) [DE06] ; [RP00].

$$\psi = \sum_{x=0}^{2^n-1} C_x \mid x > \tag{4.3}$$

Les amplitudes $C_x$ gouvernant les probabilités des différents états satisfont la propriété donnée par l'équation (4.4).

$$\sum_{x=0}^{2^n-1} \mid C_x \mid^2 = 1 \tag{4.4}$$

Dans un système classique, un registre de $n$ bits peut contenir une seule valeur parmi les $2^n$ valeurs possibles. Par contre, un registre quantique représente toutes les $2^n$ valeurs en même temps [DE06].

* **Mesure quantique** La mesure de l'état d'un qubit le force à pointer soit vers le '1' , soit vers le '0'. Le résultat dépend des amplitudes du qubit. La figure 4.2 donne un exemple de mesure d'un qubit qui aura 70% de chances de se trouver dans l'état '1' et 30% de se trouver dans l'état '0'.

### 4.2.3 Les cinq principes de l'informatique quantique

L'informatique quantique est basée principalement sur les cinq principes suivants qui sont inspirés de la mécanique quantique : la superposition d'états, l'interférence, l'enchevêtrement, le non déterminisme et la non duplication, et la réversibilité [Bel03].

**•Superposition d'états**

Un qubit peut être dans l'état 0 ou 1, mais également les deux en même temps. Donc, un registre de $n$ qubits pourra contenir $2^n$ valeurs distinctes. Une conséquence importante de cette

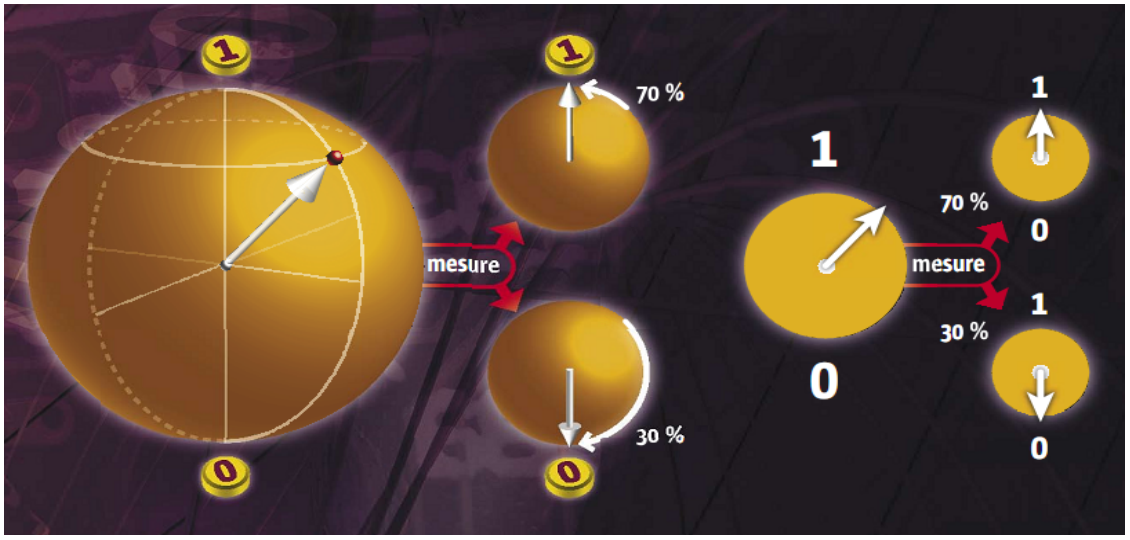


FIGURE 4.2 – Mesure quantique

propriété, la superposition d'états, est que le système peut manipuler $2^n$ valeurs en parallèle, ce qui offre une rapidité énorme aux machines quantiques.

**•Interférence**

C'est l'action qui permet d'augmenter ou diminuer la probabilité d'avoir un état. Il existe deux types d'interférence : *interférence constructive* et *interférence destructive* ; l'interférence constructive permet d'augmenter la probabilité d'obtenir un état, alors que l'interférence destructive permet de diminuer la probabilité d'obtenir un état.

**•Enchevêtrement ou intrication**

Deux bits quantiques sont dits enchevêtrés si l'un dépend de l'autre et toute modification de l'un affecte l'autre.

**•Non déterminisme et non duplication**

La mécanique quantique est non déterministe : le hasard quantique est authentique et irréductible, c'est une loi fondamentale. La copie des qubits n'est pas toujours possible. Un état inconnu ne peut pas être dupliqué.

**•Réversibilité**

L'évolution dans le temps d'un système quantique isolé est décrite par une transformation unitaire. En d'autres termes, ça veut dire que l'aboutissement d'une condition à une action peut

aussi mener à l'action inverse dans des circonstances inverses. Pour préserver la superposition et l'intrication on a besoin d'évolutions unitaires. Pour ceci, une condition nécessaire est que les opérations soient réversibles.

### 4.2.4 Calcul quantique et algorithmes quantiques

Il s'agit d'utiliser des processus quantiques pour réaliser des opérations logiques. Les unités de calcul sont des qubits évoluant dans des superpositions d'états [Pre98] ; [RP00].

L'état d'un qubit est un vecteur dans un espace vectoriel à deux dimensions. Pour manipuler ces données quantiques (représentées sous forme de bits quantiques), on distingue deux types d'opérations : opérations simples dites portes quantiques simples et successions d'opérations quantiques appelées circuits quantiques [Tal09].

**•Portes quantiques**

Les portes quantiques sont des opérations élémentaires opérant sur un qubit ou un ensemble de qubits (registre quantique). Les portes quantiques célèbres permettant des transformations unitaires sur des qubits sont les suivantes [RP00] :
- les porte quantiques simples : la transformation identité, la négation, le décalage en phase, la combinaison du décalage en phase et de la négation.
- la porte racine carrée du NOT $\sqrt{NOT}$.
- la porte controlled-NOT (CNOT).

**•Circuits quantiques**

Un circuit quantique est une combinaison de deux ou plusieurs portes quantiques permettant d'effectuer un traitement plus compliqué sur un système quantique.

**•Algorithmes quantiques**

Un algorithme quantique est une succession d'application d'opérations quantiques sur des systèmes quantiques. Ces opérations quantiques sont soit des portes logiques quantiques simples, soit des circuits quantiques. Les algorithmes quantiques offrent une complexité algorithmique nettement inférieure à celle des algorithmes classiques, grâce au concept de superposition d'états. Cette diminution de complexité a attiré l'attention de beaucoup de chercheurs qui ont fourni de grands efforts pour concevoir des algorithmes quantiques résolvant des problèmes connus du monde réel, citant l'algorithme de Shor [Sho94] pour la factorisation de grands nombres, l'algorithme de Grover [Gro96] pour la recherche dans une base de données

non triée et l'algorithme de Kitaev pour la factorisation [RP00].

Dans le contexte de l'informatique quantique, on ne peut pas vraiment parler des domaines d'applications actuels vu l'absence d'ordinateurs quantiques puissants. Malgré cela, les futures applications de l'informatique quantique sont très prometteuses, vu les algorithmes quantiques déjà existants et les modèles préliminaires d'ordinateurs quantiques, Les domaines les plus actifs selon cette perspective sont la recherche de l'information (recherche de sous groupes cachés) et la cryptographie quantique.

### 4.2.5 Avantages et limites de l'informatique quantique

L'informatique quantique offre des possibilités de calcul et des fonctionnalités qui sont rares en informatique classique. On peut les résumer dans les points suivants [Bel03].
- une puissance de calcul énorme grâce à la superposition d'états et des opérateurs quantiques permettant le traitement d'une grande quantité d'informations en parallèle.
- l'indéterminisme et la non duplication, caractérisant ce paradigme, ont des applications très utiles dans le domaine de la cryptographie.
- une complexité algorithmique remarquablement diminuée.

Malgré les possibilités offertes par l'informatique quantique, les limites suivantes restent à résoudre pour pouvoir exploiter ses atouts. Parmi les limites actuelles de l'informatique quantique, on trouve les suivantes [CNM08] ; [Bel03] :

- les algorithmes quantiques ne sont réalisables que sur des machines quantiques dont la construction est toujours en état de recherche.
- la décohérence, l'interférence destructive, cause des problèmes au processus de calcul, la moindre perturbation du système causera l'effondrement du processus de calcul et du système lui même. Pour cette raison, l'ordinateur quantique doit être totalement isolé de son environnement dans la phase de traitement.
- la difficulté de correction d'erreurs : ceci est dû à la nature quantique de la machine. Donc, plus d'effort doit être fourni pour la correction d'erreurs, causées par la décohérence, pour chaque algorithme.
- l'opération de lecture (mesure) elle même peut changer l'état du système ou causer sa corruption. Pour ceci, on doit utiliser des mécanismes bien spécifiés pour lire les résultats d'un traitement quantique avant qu'ils soient perdus. Malgré que ceci a été fourni par [Gro96], ça reste très basique et demande un effort considérable pour pouvoir être utilisé au niveau algorithmique.

## 4.3 Algorithmes Evolutionnaires Quantiques

L'exploitation de toutes les opportunités qu'offre l'informatique quantique ne sera possible qu'après la mise en place de vrais ordinateurs quantiques. En attendant la construction de machines quantiques puissantes, l'idée de simulation des algorithmes quantiques sur ordinateurs classiques ou de les combiner à d'autres méthodes conventionnelles est apparue. Parmi ces hybridations, qui ont prouvé d'être meilleures que leurs contreparties classiques, on cite : les réseaux de neurones quantiques, les colonies de fourmis quantiques, les essaims de particules quantiques et les algorithmes évolutionnaires quantiques [CNM08] qu'on présente dans cette section.

**Définition 4.1 (AEQ)** *Un algorithme évolutionnaire quantique est un algorithme évolutionnaire enrichi par les concepts et les principes de l'informatique quantique, tel que le bit quantique, la superposition d'états et les opérateurs quantiques.*

Historiquement, le premier algorithme s'inspirant de l'informatique quantique a été celui de [NM96]. Cependant, cet algorithme n'a pas suscité beaucoup d'attention à cause de son interprétation différente de l'interprétation standard de l'informatique quantique et à cause de la pauvreté de formalisme et d'évidence offerts par les auteurs.

Quatre ans plus tard, Han et Kim [HK00] ont proposé un nouveau modèle d'algorithmes génétiques s'inspirant des concepts de l'informatique quantique tels que le bit quantique (qubit), la superposition d'états et l'opérateur quantique (la porte D). Ils ont appliqué l'algorithme proposé sur un problème d'optimisation combinatoire, celui du sac-à-dos binaire (0/1 knapsack) et ont conclu que l'hybridation entre algorithmes génétiques et l'informatique quantique est une voie très prometteuse.

Dans l'algorithme de Han et Kim, il n' y a ni mutation ni croisement. Selon les auteurs, l'utilisation de ces deux opérations conduit vers une détérioration des performances. Dans ce qui suit, l'algorithme original de Han et Kim sera présenté. D'autres variantes des algorithmes évolutionnaires quantiques peuvent être trouvés dans [HK02], [HK04] ; [dCPVB05], [dCVP07] ; [ZJL03].

### 4.3.1 Individu quantique

Han et Kim [HK00] ont utilisé dans leur algorithme des individus quantiques représentés par des registres quantiques ; c'est une chaine de bits quantiques (formule 4.5) où $\alpha_1$ et $\beta_1$

représentent les amplitudes des qubits, et donc, doivent vérifier la condition $|\alpha|^2 + |\beta|^2 = 1$.

$$\begin{pmatrix} \alpha_1 & \alpha_2 & ... & \alpha_m \\ \beta_1 & \beta_2 & ... & \beta_m \end{pmatrix} \tag{4.5}$$

Grâce à cette superposition d'états, un individu quantique peut représenter en même temps toute une population d'individus, chacun avec une probabilité. Donc, il ne sera pas nécessaire dans une telle représentation d'individus d'avoir une grande taille de la population, ce qui offre plus de diversité à l'AEQ, tout en utilisant une petite population. Un peu plus tard, [Han03] démontre que les performances offertes par un nombre d'individus un peu supérieur sont meilleures que celles données par un seul individu.

En plus de la mise à jour de la meilleure solution trouvée, deux opérations quantiques sont appliquées d'une façon itérative sur les individus quantiques jusqu'à l'atteinte d'une condition d'arrêt. Il s'agit de *la mesure quantique* et *la porte D*.

### 4.3.2 Mesure quantique d'individus

Cet opérateur permet d'extraire un individu classique à partir d'un individu quantique en appliquant des mesures quantiques individuelles (comme vu dans la section 4.2.2 sur les qubits). La figure 4.3 donne une illustration de la mesure quantique appliquée à un individu quantique.

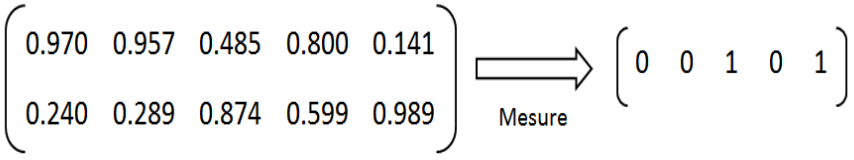


FIGURE 4.3 – Mesure quantique d'individus

### 4.3.3 La porte D

A chaque itération, l'actuelle meilleure solution sert comme guide pour rechercher de nouvelles solutions qui peuvent s'avérer meilleures. Cela est implémenté via une porte quantique, appelée la porte D, qui effectue une rotation du qubit en question de façon à augmenter la probabilité d'avoir la valeur binaire du bit correspondant dans la meilleure solution actuelle (fig.4.4).

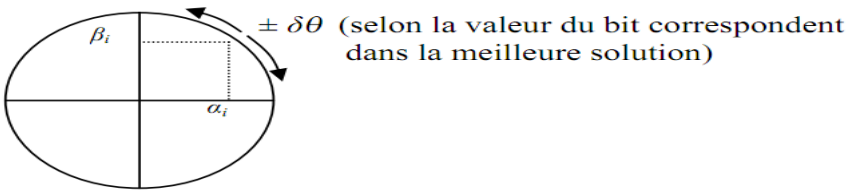


FIGURE 4.4 – Effet de la porte D sur un qubit

La porte D prend la forme matricielle donnée par la formule(4.6).

$$\begin{pmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{pmatrix} \tag{4.6}$$

Le tableau 4.1 donne la direction de rotation en fonction de $\alpha_1$, $\beta_1$ et de la valeur du bit correspondant dans la meilleure solution actuelle.

TABLE 4.1 – Direction de la rotation

| $\alpha$ | $\beta$ | valeur du bit de référence | angle de rotation |
|---|---|---|---|
| $>0$ | $>0$ | 1 | $+\delta\theta$ |
| $>0$ | $>0$ | 0 | $-\delta\theta$ |
| $>0$ | $<0$ | 1 | $-\delta\theta$ |
| $>0$ | $<0$ | 0 | $+\delta\theta$ |
| $<0$ | $>0$ | 1 | $-\delta\theta$ |
| $<0$ | $>0$ | 0 | $+\delta\theta$ |
| $<0$ | $<0$ | 1 | $+\delta\theta$ |
| $<0$ | $<0$ | 0 | $-\delta\theta$ |

Le pseudo-code de l'algorithme de Han et Kim [HK00] est donné dans l'algorithme 8, où : Q(t) est la population composée de chromosomes quantiques à l'instant t, B(t) est la meilleure solution jusqu'à l'instant t, et P(t) est l'ensemble de solutions binaires issues de la mesure de Q(t).

**Data**:
**Output**:
$t \leftarrow 0$
Produire P(t) par observation de Q(t)
Evaluer P(t)
Stocker la meilleure solution parmi P(t) dans B(t)
**while** *(critère d'arrêt non satisfait)* **do**
    $t \leftarrow t+1$
    Produire P(t) par observation de Q(t-1)
    Evaluer P(t)
    Générer Q(t) en appliquant la porte D sur Q(t-1)
    Stocker la meilleure solution parmi B(t-1) et P(t) dans B(t)
**end**

**Algorithm 8:** Algorithme évolutionnaire quantique (AEQ)

### 4.3.4 Discussion de l'algorithme évolutionnaire quantique

Tout algorithme évolutionnaire doit être capable de bien équilibrer entre la diversification et le renforcement, souvent appelés respectivement : *exploration* et *exploitation*. Dans ce contexte, on introduit une petite analyse de l'AEQ de Han et Kim [HK00] vis-à-vis ces deux critères [Tal09].

Dans l'algorithme évolutionnaire quantique de Han et Kim [HK00], le codage quantique des chromosomes constitue le moyen employé par les auteurs pour garantir la diversification de la recherche. Un qubit pourrait, dans la majorité des situations, produire lors de sa mesure la valeur 0 ou la valeur 1. Donc, un chromosome pourrait potentiellement couvrir tout l'espace de recherche.

L'opérateur de rotation, ou la porte D, est l'outil garantissant le renforcement de la recherche autour de la meilleure solution actuelle. Au cours des itérations, cet opérateur augmente, pour chaque qubit, la probabilité d'avoir la valeur du bit correspondant dans la meilleure solution. Dans la majorité des situations, on garde la possibilité, même affaiblie, d'avoir la valeur inverse qui pourrait contribuer à la découverte d'une nouvelle meilleure solution.

Cette stratégie n'est pas tout le temps gagnante. Le problème majeur est qu'en cas de non découverte d'une nouvelle meilleure solution après un certain nombre d'itérations, la probabilité pour un qubit de générer la valeur binaire inverse de la valeur du bit correspondant dans la meilleure solution sera amplement réduite, voir même mise à 0. La généralisation de ce phénomène sur l'ensemble des qubits entraînera l'effondrement de la superposition d'états et le blocage des chromosomes dans un optimum local.

## 4.4 Autres Algorithmes Inspirés du Quantique

En se basant sur le framework de l'algorithme évolutionnaire quantique de Han et Kim [HK00] et en s'inspirant d'autre approches évolutionnaires, d'autres instances et hybridation d'AEQ ont été créées et appliquées avec succès pour la résolution de problèmes d'optimisation. Dans ce qui suit on donne une brève description de quelques algorithmes s'inspirant de l'informatique quantique.

### 4.4.1 Algorithmes génétiques quantiques

Les AEQs ont prouvé leur efficacité pour la résolution de problèmes d'optimisation combinatoire. Cependant, un problème qui les limite est la perte de diversité avec le temps. Pour pouvoir faire un bon équilibre entre l'exploration et l'exploitation, une variante des AEQs a été proposée et utilisée pour résoudre divers problèmes, tels que le recalage d'images [TBD04], [TDB06], le problème du voyageur de commerce [TDB04], la vérification formelle [LS07], la bio-informatique [LMB08] et la segmentation d'images [TBD07]. Dans cette variante, en plus des opérateurs quantiques de mesure et de mise à jour, des opérateurs de croisement et de mutation quantiques sont utilisés.

**•Croisement quantique**

Ce croisement a le même principe qu'un croisement classique. Mais, il opère sur des chromosomes quantiques. Donc, c'est un croisement de matrices de probabilités qui génère comme résultat des matrices de probabilités. Le croisement quantique entre deux individus (parents) en un point donné permet de générer deux nouveaux individus (descendants) dont les gènes proviennent de leurs deux parents (fig.4.5).

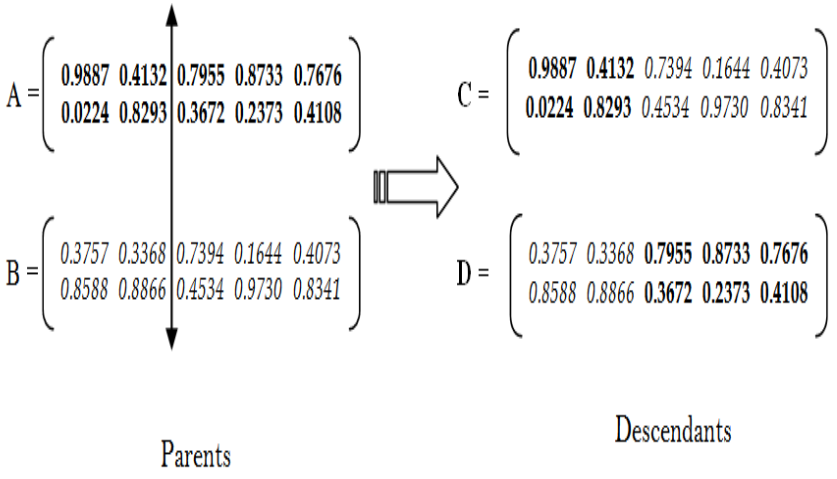


FIGURE 4.5 – Croisement quantique

**•Mutation quantique**

La mutation classique opère comme une petite perturbation qui inverse le bit muté. Dans une mutation quantique, il y a aussi une perturbation, mais elle opère sur les probabilité d'un

qubit du chromosome en question de la façon suivante.
Soit un qubit $A = \alpha|0> + \beta|1>$ . La mutation quantique du qubit A génère le qubit $B = \beta|0> + \alpha|1>$ . La figure 4.6 donne un exemple dans lequel une mutation quantique est appliquée au troisième point du chromosome d'entrée. Après une opération de croisement quantique, le

| | | | |
|---|---|---|---|
| 0.2383 | 0.1079 | **0.5057** | 0.8337 |
| 0.9712 | 0.9942 | **0.8627** | 0.5522 |
| | | ↓ | |
| 0.2383 | 0.1079 | **0.8627** | 0.8337 |
| 0.9712 | 0.9942 | **0.5057** | 0.5522 |

FIGURE 4.6 – Mutation quantique

nombre d'individus augmente. Donc, pour maintenir la taille initiale de la population, une opération de sélection quantique s'impose. Il est difficile, bien que faisable, de considérer les caractéristiques des chromosomes quantiques pour formuler un critère de sélection. Ceci est dû au fait qu'un chromosome quantique représente potentiellement une multitude de solutions éparpillées dans l'espace de recherche.

On a donc suggéré de se baser sur la qualité de la solution binaire issue d'une opération de mesure pour évaluer un chromosome. Une opération de mesure est appliquée sur l'ensemble des chromosomes. Une solution est obtenue de chaque chromosome et les chromosomes desquels émanent les meilleures solutions sont choisis pour former la nouvelle génération [Tal09].

### 4.4.2 Boids de Reynolds quantiques

Les boids de Reynolds quantiques sont le résultat de l'enrichissement du modèle classique des boids de Reynolds [Rey87] (chapitre 1, section 7.2.2) par les concepts de l'informatique quantique tels que la superposition d'états et la mesure quantique.

Un boid de Reynolds quantique est une suite de qubits qui peut être représentée sous la forme d'un registre quantique. Par conséquence, les opérations de base du modèle de Reynlods sont adaptées pour créer des opérations de boids quantiques qui sont nommées : cohésion quantique, séparation quantique et alignement quantique.

La représentation quantique des boids a offert à ce modèle plus de diversité par rapport aux représentations classiques (binaire ou réelle) du fait que chaque boid représente une multitude de boids en superposition.

### 4.4.3 Réseaux de neurones quantiques

S'inspirant de l'informatique quantique, l'approche des réseaux de neurones quantiques vise à améliorer les performances des réseaux de neurones formels classiques avec de nouveaux algorithmes quantiques [San07]. Ventura et Martinez [VM98] ont proposé un modèle basé sur la superposition des poids de connexions du réseau, le principe est simple : lors de l'utilisation du réseau, une mesure est effectuée et les poids de connexions sont fixés de manière probabiliste (phénomène de décohérence), l'exécution du réseau s'effectue ensuite classiquement comme dans un réseau de neurones classique. Les intérêts possibles sont : d'une part, le gain de performances dû aux algorithmes quantiques. D'autre part, la possibilité offerte par l'aspect probabiliste de la mesure quantique d'effectuer certaines fonctions inaccessibles aux réseaux de neurones classiques telles que le problème de l'XOR [VM98] ; [ARL03].

Les réseaux de neurones quantiques offrent des gains non négligeables, tant au niveau de la capacité de stockage que de la rapidité. Les propriétés de superposition d'états d'intrication et de décohérence ont permis aux réseaux de neurones quantiques d'être meilleurs que leur homologue classique [San07].

## 4.5 Avantages des Algorithmes Inspirés du Quantique

Comme on l'a déjà vu dans les sections si-dessus, le paradigme des algorithmes inspirés du quantique est basé sur l'exploitation des concepts quantiques permettant de surmonter les limites des approches d'optimisation classiques, notamment la convergence prématurée et la lenteur de convergence. Ceci a été relativement atteint dans ce type d'algorithmes. On peut résumer les points forts des approches inspirées du quantique dans les points suivants :

- La représentation quantique des individus (boids dans les boids de Reynolds quantiques, poids dans les RNAs quantiques, ou chromosomes dans les AGs quantiques) permet à un algorithme inspiré du quantique une meilleure exploration : une bonne diversité est offerte, ce qui permet de couvrir une grande partie de l'espace de recherche.

- Ces algorithmes sont caractérisés par une complexité réduite : très peu d'individus sont nécessaires pour une bonne représentation de l'espace de recherche.

- La porte D (connue des fois sous le nom d'*interférence*) qui constitue la base de comportement dans la majorité des algorithmes inspirés du quantique permet une bonne exploitation des solutions déjà trouvées.

## 4.6 Conclusion

Dans ce chapitre, on a vu une introduction à l'informatique quantique comme domaine émergent en informatique. On a présenté ses principes de base et ses avantages et limites. On a également parlé des algorithme inspirés du quantique tout en présentant trois modèles qui ont un lien fort avec nos contributions : les algorithmes évolutionnaires quantiques, les algorithmes génétiques quantiques et les boids de Reynolds quantiques. On a aussi introduit les réseaux de neurones quantiques, vus comme un bon modèle artificiel des systèmes complexes adaptatifs. Finalement, une petite synthèse des avantages des algorithmes inspirés du quantique a été donnée.

Dans le prochain chapitre, nous allons introduire deux contributions. La première basée sur la métaheuristique PSO pour la résolution de l'émergence inversée. La deuxième basée sur l'hybridation de PSO avec les principes de l'informatique quantique.

Chapitre 5

# Utilisation du PSO puis QPSO pour résoudre le problème d'émergence inversée

## 5.1 Introduction

Dans ce chapitre,, nous décrivons deux nouvelles approches de résolution de l'émergence inversée. La première méthode utilise la métaheuristique PSO, la deuxième utilise la version quantique du PSO : le quantum PSO. Les deux métaheuristiques font évoluer un système complexe qui modélise une image : l'automate cellulaire. Le processus évolutionnaire guidé par l'algorithme PSO puis QPSO permet d'extraire le sous-ensemble de règles qui réalise une bonne détection de contours (et filtrage de bruit pour le QPSO). Une fonction fitness mesure la qualité de l'image résultat. Les résultats sont comparés à des algorithmes de la littérature, et s'avèrent prometteurs.

## 5.2 Contribution 1 : Application de la Métaheuristique PSO pour la Résolution de l'Emergence Inversée

Dans la littérature relative au domaine du traitement d'images, on trouve une multitude de méthodes de détection de contours qui s'appuient sur divers algorithmes. La plupart de ces méthodes se basent sur le calcul des dérivées directionnelles. D'autres méthodes abolissant le calcul des dérivées se basent plutôt sur des connaissances préalables de la nature et des caractéristiques de l'image, mais ces exigences limitent le champ d'application de la méthode. On tente ici, d'élaborer une méthode alternative pour la segmentation d'images, qui s'inspire des

principes des systèmes complexes et de l'émergence. Elle utilise les automates cellulaires (ACs) comme modèle mathématique, qui est une représentation triviale de l'image. La puissance de ces ACs est décuplée par l'utilisation de la métaheuristique, qui leur fournit des aptitudes d'évolution et d'adaptation. Cette hybridation de procédés donne un outil puissant, qui est les Automates Cellulaires Evolutionnaires (EvCA), qu'on se propose d'appliquer au problème de détection de contours sur des images de différents types (binaires, à niveaux de gris,...etc) [DB12],[DM12].

On se propose d'utiliser les aptitudes calculatoires de l'AC, en lui procurant comme donnée de départ (configuration initiale) l'image à segmenter en entrée ; et on s'attend à récupérer en sortie l'image segmentée, par décodage d'une configuration finale, fournie par l'AC après plusieurs itérations. Les configurations intermédiaires sont considérées comme des étapes de calcul.

La méthode peut être résumée comme suit : La métaheuristique PSO est utilisée pour la recherche de la règle d'AC adéquate à la réalisation de la tâche de détection de contour. Pour ce faire, on utilise un ensemble d'images de référence, ainsi que leurs répliques segmentées. Ce sera le pool d'entraînement de PSO. La fonction fitness mesure la qualité du processus de détection de contours, qui résulte de l'application de l'AC sur le pool d'images de référence. Le processus évolutionnaire commence avec un essaim initial aléatoire de règles d'ACs. Le résultat sera un ensemble de règles de transition locales, qui définissent un AC détecteur de contours.

### 5.2.1 Format des règles

Dans ce travail, la classe d'automates cellulaires utilisés est appelée "totalistique". L'état d'une cellule dans un AC totalistique est représenté par un nombre (usuellement une valeur entière dans un ensemble fini). La valeur d'une cellule à un instant t dépend uniquement de la somme des valeurs des cellules avoisinantes (la cellule concernée peut être incluse)à l'instant t-1. Un AC totalistique ne prend pas en considération la position des cellules voisines, mais seulement leurs valeurs. Ce type de règles permet d'optimiser l'espace de recherche et le temps d'exécution.

L'automate cellulaire et sa fonction de transition sont définis comme suit :
Chaque cellule a 2 états : 0 ou 1. Une cellule est dite morte si son état est égal à 0, et vivante sinon. Le type de voisinage utilisé est celui de Moore (8 cellules). Le nombre de cellules vivantes varie de 0 à 9 et donne alors dix valeurs possibles. Cette variable est notée NCV. Le nombre de règles est assez grand à tester manuellement, et il est évident qu'il n y'a qu'un petit sous-ensemble intéressant parmi ces règles, qui aboutit au résultat souhaité.

La règle de transition est définie comme suit : l'état futur de la cellule centrale (noté FS) est mis à jour avec la valeur dans la ligne (binary representation)qui correspond à l'index de tableau (NAC), qui est le nombre de cellules vivantes.

Le tableau 5.1 illustre l'interprétation de la règle 120 :

TABLE 5.1 – Interprétation de la règle 120

| Rule number | 120 | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Binary representation | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | FS |
| NAC | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | |

Par exemple, on prend un pixel et ses 8 voisins dans la configuration initiale, on compte le nombre de cellules vivantes (NAC), s'il est égal à 4, 5, ou 6 l'état futur de la cellule centrale devient 1, sinon il devient 0.

### 5.2.2 Evolution de l'automate Cellulaire par la métaheuristique PSO

L'algorithme CA-PSO tente de trouver les règles qui produisent le meilleur contour sur une image, en appliquant une règle de l'AC sur l'image en entrée. La fonction fitness est calculée entre l'image vérité terrain, et l'image produite par la règle de l'AC. Les paramètres de PSO sont ajustés, une autre particule (règle) est choisie dans l'essaim, le même processus est répété jusqu'à convergence, qui est atteinte lorsque la meilleure fitness est obtenue. On note la règle qui a permis d'atteindre la meilleure fitness.

L'essaim (la population représentant l'espace de recherche) est constitué par l'ensemble des règles de l'automate cellulaire. Une particule est une règle de transition. A chaque étape de l'algorithme PSO, la taille de l'essaim est fixée à 30 pour la convergence rapide du processus.

Chaque particule de l'essaim est définie par :

•Sa position : elle représente le numéro de la règle de transition de l'AC. C'est une valeur codée sur 10 bits.

•Sa vitesse : c'est un nombre réel, initialisé à 0, il guide le processus vers la convergence.

•Sa fitness : c'est la fonction objectif qui mesure la qualité du contour obtenu, après application de la règle correspondante. Elle est initialisée à zéro.

### 5.2.3 La fonction Fitness

La fonction Fitness utilisée pour guider le processus de sélection des règles a un grand effet sur les résultats finaux. Pour cela, on considère trois fonctions de fitness : fonction SSIM (Structural SIMilarity index)[WBSS04], Root Mean Square Error (RMSE) et la distance de Hamming. Le rôle de ces fonctions est de mesurer la différence de qualité entre l'image résultat et l'image de référence.

La fonction SSIM a la particularité de mesurer la similarité entre deux images en tenant compte de trois facteurs indépendants : la luminance, le contraste et la structure.

La métrique SSIM entre deux images x et y est définie comme suit :

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{5.1}$$

où $\mu_x, \mu_y, \sigma_x^2, \sigma_y^2, \sigma_{xy}$, sont respectivement la moyenne de $x$, la moyenne de $y$, la variance de $x$, la variance de $y$, et la covariance de $x$ et $y$. Selon [WBSS04], $C_1$ est initialisé à $(0.01 * 255)^2$ et $C_2 = (0.03 * 255)^2$.

La distance de Hamming calcule le nombre de pixels différents entre deux images.

La mesure RMSE est calculée selon cette équation

$$RMSE = \sqrt{\frac{1}{MN}\sum_{r=0}^{M-1}\sum_{c=0}^{N-1}[E(r,c) - O(r,c)]^2} \tag{5.2}$$

où $O(r,c)$ est l'image originale (dans notre cas la vérité terrain), $E(r,c)$ est l'image reconstituée (dans notre cas le résultat de notre algorithme).

### 5.2.4 L'algorithme CA-PSO pour la détection de contours

L'algorithme 9 montre la manière dont CA-PSO opère et donne ses différentes étapes. Le processus PSO est initialisé, chaque particule (règle) de l'essaim est convertie en binaire et appliquée sur l'image pixel par pixel, selon la règle de transition définie dans le paragraphe (5.2.1). A chaque étape, le nombre NAC est évalué, la transition correspondante est appliquée sur l'image. On obtient une image résultat (contour). Sa fitness est calculée, en comparaison avec la vérité terrain. Une autre particule est sélectionnée dans l'essaim, les paramètres de

PSO sont mis à jour, ce qui permet d'obtenir un nouvel essaim à tester (un autre ensemble de règles). Les mêmes étapes sont répétées jusqu'à atteindre le critère de convergence. A la fin de l'algorithme on obtient en sortie le meilleur contour et les règles qui ont permis de l'obtenir.

**Data**: *imageoriginale*, *veriteterrain*
**Output**: contour, règle

1. Initialisation de PSO : lire image entrée, initialiser taille essaim
2. **for** *i=1 à taille-essaim* **do**
   2.1. particle [i].position = rand(1023)
   2.2. particle[i].velocity = 0
   2.3. particle[i].fitness = 0
**end**
3. **for** *i=1 à taille-essaim* **do**
   p-best[i]= particle[i]
**end**
4.G-best = particle[1]
5. **while** *critère d'arrêt non satisfait* **do**
   **for** *p = 1 to swarm-size* **do**
      application de la regle d'AC pour chaque particule
      **for** *chaque pixel de l'image* **do**
         calculer NAC
         appliquer la règle de transition,
         sauvegarder le résultat dans une nouvelle image contour
      **end**
   **end**
   **for** *i = 1 to swarm-size* **do**
      **if** *particle[i].fitness > p- best[i]* **then**
         p-best[i] = particle[i]
      **end**
      **if** *g-best[i].fitness > particle[i].fitness* **then**
         g-best[i] = particle [i]
      **end**
   **end**
   **for** *i = 1 à taille-essaim* **do**
      mettre à jour les paramètres de PSO : position et vitesse
      obtention d'un nouvel essaim à tester
   **end**
**end**
retour à l'étape 3.

**Algorithm 9:** Principe de l'algorithme CA-PSO pour la détection de contours

### 5.2.5 Résultats expérimentaux

Cette section présente les résultats de l'algorithme CA-PSO pour la détection de contours.

Durant ces expérimentations, la taille de l'essaim est égale à 30.

#### 5.2.5.1 Les meilleures règles obtenues

Les expériences menées sur des dizaines d'images de types variés (synthétique, binaire, niveaux de gris) montrent que trois meilleures règles sont extraites, qui donnent d'excellents contours, après seulement une application de la règle totalistique sur l'image en entrée. Ces règles sont : la règle 56, la règle 120, la règle 112.

Le taux de fitness moyenne obtenue est de l'ordre de 99% , ce qui dénote la grande robustesse du paquet de règles optimal. Le processus de recherche de règles prend un temps aléatoire, qui peut être court ou long. L'explication est : puisque la taille de l'essaim est de 30 particules, sélectionnées aléatoirement, il est possible que la bonne règle soit dans le premier essaim, comme il est possible que la bonne règle soit extraite après plusieurs changements dans l'essaim. Ce qui est intéressant à noter, c'est que, une fois les bonnes règles identifiées, il suffit de les appliquer sur l'image à traiter *une seule fois*, et directement, on obtient en sortie l'image contour correspondante.

#### 5.2.5.2 Résultats sur images de synthèse

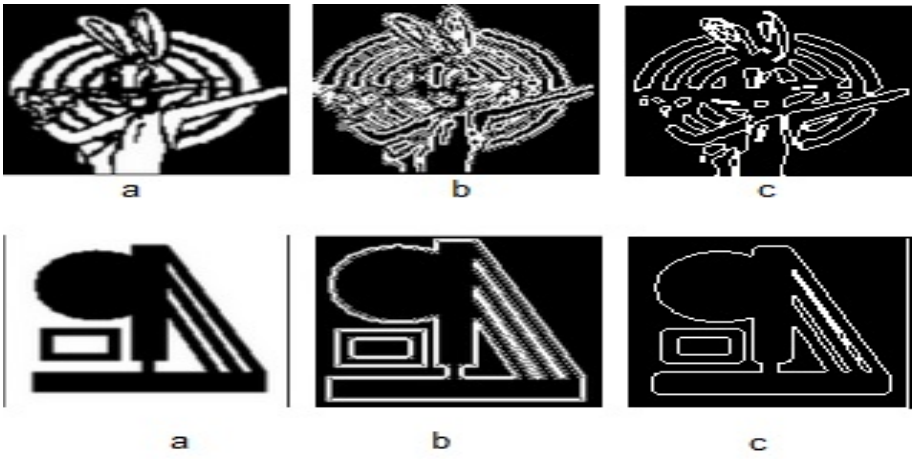


FIGURE 5.1 – Résultats de l'extraction de contours sur des images de synthèse (a) image originale (b) résultat [SBM07] (c) résultat CA-PSO

On constate que les contours obtenus par [SBM07] sont épais et entourés de bruit, alors que ceux de notre méthode sont nets, fins et sans bruit.

#### 5.2.5.3 Résultats sur image à niveaux de gris

Dans ce paragraphe, nous allons montrer les résultats comparatifs de la détection de contours sur une image réelle à niveaux de gris, réalisée par le détecteur standard de Canny, et par notre algorithme CA-PSO.

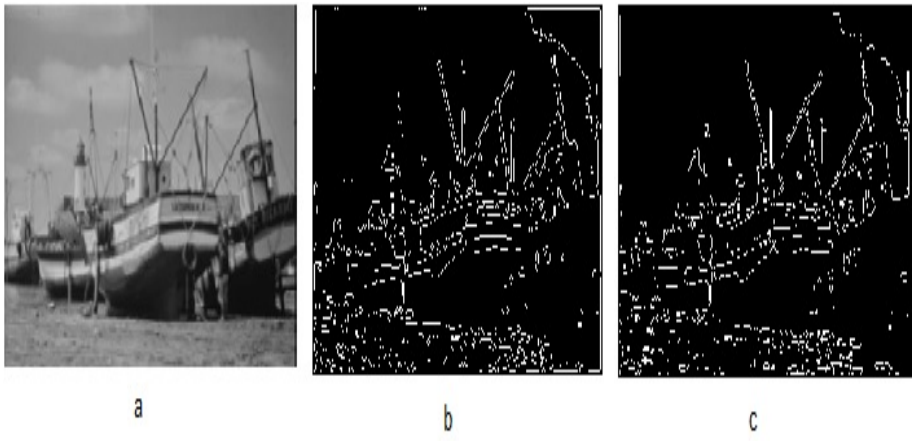


FIGURE 5.2 – Résultats de l'extraction de contours sur l'image bateau (a) image originale (b) résultat Canny (c) résultat CA-PSO

#### 5.2.5.4 Résultats sur image réelle

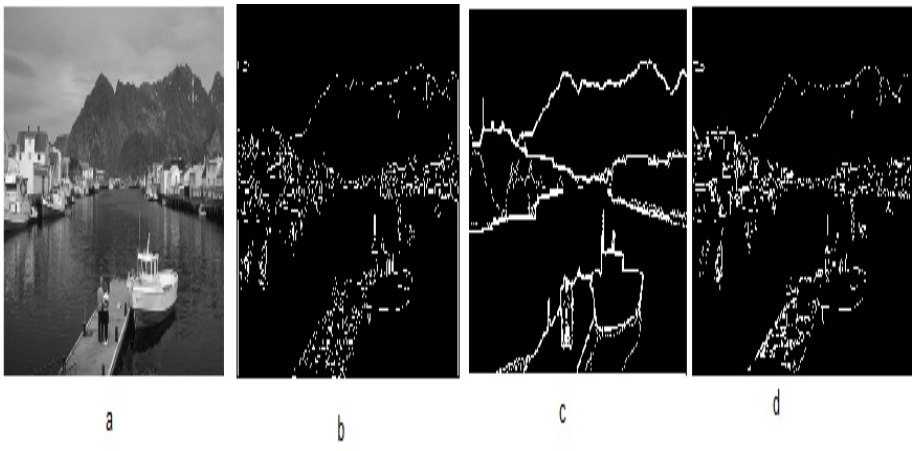


FIGURE 5.3 – Résultats de l'extraction de contours sur une image réelle (Benchmark Berkeley) (a) image originale (b) résultat Canny (c) contour étalon (d) résultat CA-PSO

#### 5.2.5.5 Résultats sur image médicale

On a testé notre algorithme sur des images médicales (IRM cérébrale). Le résultat est illustré en figure 5.4 La même constatation que pour les images précédentes s'impose : malgré

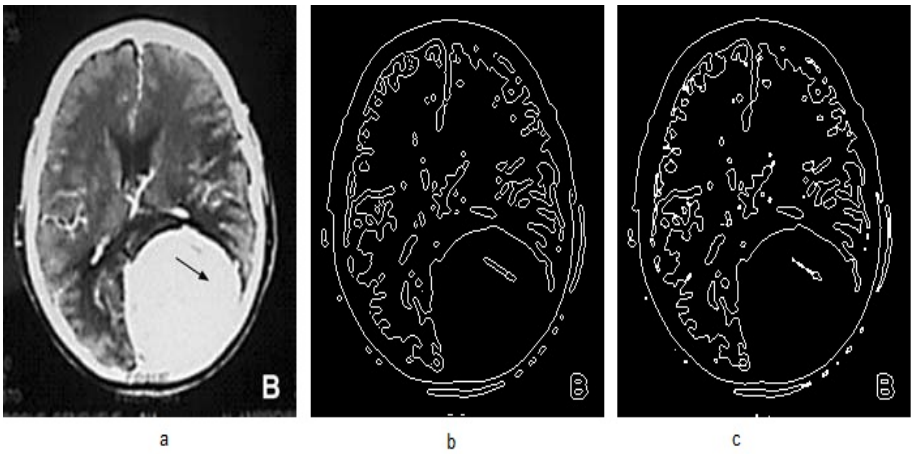


FIGURE 5.4 – Résultats de l'extraction de contours sur l'image d'une IRM cérebrale (a) image originale (b) résultat Canny (c) résultat CA-PSO

la complexité des images IRM, et les textures différentes qui les caractérisent, notre méthode a extrait les contours de manière correcte, en éliminant quelques bruits apparus dans le résultat de Canny. Les contours sont fins et continus.

#### 5.2.5.6 Temps d'exécution

Dans le tableau 5.2, on évalue les performances de notre méthode du point de vue temps d'exécution. On constate que notre méthode donne un meilleur temps d'exécution par rapport à celui de Canny, et cela pour différents types d'images.

TABLE 5.2 – Comparaison du temps d'exécution sur différentes images

| Image | Taille image | Temps CA-PSO | Temps Canny |
|---|---|---|---|
| Bateau | 427 * 302 | 0.422 | 0.672 |
| IRM | 225 * 260 | 0.187 | 0.578 |
| lapin | 256 *256 | 0.203 | 0.656 |

TABLE 5.3 – Valeurs de la fitness SSIM pour les algorithmes de Canny, EV-CA et CA-PSO

| Algorithme | Canny | EV-CA | CA-PSO |
|---|---|---|---|
| Caractères | 0.98775422 | 0.9726516 | 0.99547786 |
| Formes | 0.9811263 | 0.9715647 | 0.9987343 |
| Lapin | 0.9858974 | 0.9726549 | 0.9974553 |
| Oiseau | 0.98622844 | / | 0.99262532 |
| Ile | 0.98392303 | / | 0.99944197 |

### 5.2.6 Conclusion

Dans cette contribution, une nouvelle méthode pour résoudre le problème d'émergence inversée a été proposée et testée sur le cas de détection de contours sur images. Cette méthode est basée sur l'hybridation de deux paradigmes puissants dans les domaines de systèmes complexes et la vie artificielle : l'automate cellulaire et l'optimisation par essaim de particules. Un processus évolutionnaire détermine les règles locales d'un AC capable d'extraire les contours sur plusieurs types d'images. Dans ce but, on a utilisé la métaheuristique PSO pour faire évoluer l'ensemble des règles pour les AC candidats à l'exécution de cette tâche. Le processus a abouti à 3 règles, qui donnent la meilleure fitness et un contour très acceptable.
Après avoir testé l'algorithme développé sur une multitude d'images de types différents, en comparant les résultats obtenus avec d'autres détecteurs de contours soit standard et connus, tels que Canny, Sobel...etc, ou bien issus de travaux précédents sur les AC, on conclut que la solution fournie par notre algorithme est efficace pour traiter divers types d'images, et donne des résultats satisfaisants.
Les résultats expérimentaux sont encourageants et la comparaison avec un algorithme basé sur l'évolution génétique démontre la faisabilité, la convergence et la robustesse de notre algorithme CA-PSO.
La suite envisagée à ce travail, qui fera l'objet de la partie suivante, est l'introduction des principes de la mécanique quantique, pour aboutir à la métaheuristique Quantum PSO, puis sa validation sur la détection de contours et le filtrage de bruits.

## 5.3 Contribution 2 : Utilisation du PSO Quantique pour la résolution de problème inverse

Récemment, en s'inspirant de la mécanique quantique, et de l'analyse de trajectoire de PSO [CK02], Sun et al. ont utilisé une nouvelle stratégie pour échantillonnage autour des meilleures positions précédentes [SFX04]. Plus tard, il y'a eu introduction dans l'algorithme de la moyenne des meilleures positions, une nouvelle version de QPSO a vu le jour : Quantum

behaved particle swarm optimization [SXF04].
L'équation itérative de QPSO est très différente de celle de PSO. De plus, contrairement à PSO, les particules dans la version quantique de PSO n'ont pas de vecteurs vitesse, il y'a moins de paramètres à ajuster, ce qui rend QPSO plus facile à implémenter.
L'algorithme QPSO a montré son efficacité dans la résolution d'un large éventail de problèmes d'optimisation. QPSO surpasse PSO en plusieurs aspects, comme la simplicité de ses équations, moins de paramètres de contrôle, rapidité de convergence, mode opératoire simple...etc [SXF04],[SFP+11],[SFW+12].

### 5.3.1 L'Algorithme Quantum-behaved PSO

Dans le modèle quantique de PSO, chaque particule a un comportement quantique, on peut seulement lire la probabilité que la particule apparaisse à une position x, à partir de la fonction de densité de probabilité $|\psi(x,t)|^2$, dont la forme dépend du champ de potentiel auquel appartient la particule. L'état d'une particule est décrit par l'équation onde $\psi(x,t)$ au lieu de position et vitesse. Le comportement dynamique de la particule est largement différent de celui dans le PSO classique. Les valeurs exactes de la position et de la vitesse ne peuvent être déterminées simultanément.
En résolvant l'équation de Schrödinger, on obtient la fonction de densité de probabilité normalisée $Q$ :

$$Q(y) = |\varphi(y)|^2 = \frac{1}{L} e^{-2|y|/L} \tag{5.3}$$

où $L$ est la variable la plus importante, qui détermine le périmètre de recherche de chaque particule. Employant la méthode de Monte Carlo, pour $y = x - p$, on obtient la position de la particule comme suit :

$$x = p \pm (L/2) * ln(1/u), u = rand(0,1) \tag{5.4}$$

où $u$ est un nombre aléatoire tiré de $[0,1]$, $p$ est un point stochastique entre *pbest* (la position donnant la meilleure fitness jusqu'à maintenant) et *gbest* (la meilleure de tous les pbest de l'essaim). $p$ est considéré comme l'attracteur local de la particule. La valeur de $L$ est évaluée comme suit :

$$L(t) = 2 * \beta * |mbest - x(t)| \tag{5.5}$$

$$mbest = \frac{1}{M}\sum_{i=1}^{M} P_i = (\frac{1}{M}\sum_{i=1}^{M} P_{i1}, \frac{1}{M}\sum_{i=1}^{M} P_{i2}, ..., \frac{1}{M}\sum_{i=1}^{M} P_{iD}) \tag{5.6}$$

où *mbest* (mean best position) est définie comme la valeur moyenne des meilleures positions de toutes les particules, $M$ est la taille de la population. $L$ peut être considéré comme "la force de créativité ou imagination", car elle caractérise le champ d'application des connaissances de

la particule, par conséquent, plus la valeur de $L$ est grande, plus la particule a des chances de trouver de nouvelles connaissances.
Le paramètre $\beta$, appelé *contraction-expansion coefficient*, est le seul paramètre de l'algorithme QPSO. A partir de résultats de simulations, il a été constaté que QPSO a de meilleures performances lorsque $\beta$ varie de 1.0 au début de la recherche, à 0.5 à la fin de la recherche, dans le but d'équilibrer entre exploration et exploitation [SWB05],[SXL05]. L'équation (5.4) peut être écrite comme suit :

$$x(t+1) = p \pm \beta * |mbest - x(t)| * ln(1/u) \tag{5.7}$$

La procédure d'implémentation de QPSO est donnée par l'algorithme 10.

```
Data:
Output:
1. Initialiser la population ;
2. Initialiser p_i avec la position initiale correspondante x_i ;
3. Initialiser p_g = argmin(f(p_i)) ;
4. for t=1 à maximum iteration T do
    4.1. Calculer la moyenne des meilleures positions mbest(t) avec l'équation (5.6)
    4.2. Beta décrémenté linéairement de 1.0 à 0.5
    for i= 1 à taille population M do
        if f(pi) < f(xi) then
            pi = xi
        end
        p_g = argmin(f(p_i)) ;
        for j = 1 to D do
            f = Rand(0,1) ;
            Pd=f*pi+(1-f)*pg(d) ; u = Rand(0,1) ;
            if u>0.5 then
                xi = int16(pd - beta * abs(mbest(d) - xi)* ln(1/u)) ;
                else
                    xi = int16(pd + beta * abs(mbest(d) - xi)* ln(1/u)) ;
                end
            end
        end
    end
end
```

**Algorithm 10:** Pseudo-code de l'algorithme QPSO

### 5.3.2 L'approche proposée

Dans cette partie, nous proposons l'utilisation de l'algorithme Quantum-behaved PSO pour faire évoluer un AC qui réalise plusieurs tâches de traitement d'images, tels que la détection de contours et le filtrage de bruit [DBOS18]. L'AC est utilisé comme outil de modélisation pour l'image, l'algorithme QPSO est la stratégie de recherche qui extrait les règles appropriées par un processus évolutif.

### 5.3.3 Les règles de transition

Dans cette partie, nous utilisons les mêmes principes de règles de transition vus dans la contribution 1 (voir paragraphe 5.2.1)

### 5.3.4 Réglage de paramètres

L'approche QPSO est inspirée de PSO, avec un avantage majeur : il y'a un seul paramètre à ajuster : le coefficient $\beta$, qui est automatiquement réglé dans l'algorithme. Le coefficient $\beta$ diminue linéairement de 1.0 à 0.5 durant l'exécution. Le choix de ces valeurs est justifié par de multiples recommandations dans la majorité de la littérature traitant l'algorithme QPSO. Cette recommandation engendre une bonne performance de l'algorithme QPSO.
La position de la particule $x_i$ représente la règle de transition. C'est une valeur codée sur 10 bits, variant de 0 à 1023. Au début de l'algorithme, elle est initialisée de façon aléatoire. Dans les étapes suivantes, les valeurs de la position $x_i$ sont calculées par l'équation (5.7).
Au début de l'exécution, l'utilisateur doit fixer le nombre d'itérations et la taille de la population. Plusieurs expériences sont conduites, en changeant la taille de la population et le nombre d'itérations. La taille de la population a été testée dans l'intervalle [150,300], et le nombre d'itérations dans l'intervalle [200,500]. Au bout de ces expériences, on a relevé les meilleures règles et les meilleures fitness.
Dans une seconde étape, on diminue les valeurs, la taille de la population dans l'intervalle [10,30], et le nombre d'itérations dans l'intervalle [30,50]. En jugeant d'après les résultats obtenus, on conclut que l'algorithme QPSO a de très bonnes performances, même avec une petite taille de population, et un nombre réduit d'itérations.
Les meilleurs valeurs de fitness sont atteintes après 40 générations. Au delà de ce seuil, aucune amélioration des résultats n'est notée. On conclut qu'il est inutile d'augmenter le nombre d'itérations et la taille de la population, ce qui induit une augmentation du temps d'exécution, alors qu'aucune amélioration des valeurs de fitness n'est relevée.

### 5.3.5 Les fonctions fitness

Dans cette partie, nous utilisons les mêmes fonctions de transition vues dans la contribution 1 (voir paragraphe 5.2.3)

### 5.3.6 L'algorithme QPSO pour la détection de contours

L'algorithme 11 montre les différentes étapes de QPSO pour la détection de contours.

**Data**: *imageoriginale*, *vritterrain*
**Output**: contour, règle

1. Lire l'image en entrée, et l'image vérité terrain ;
2. Initialiser la taille de l'essaim et le nombre d'itérations ;
3. Beta =1 ;
4. Initialiser aléatoirement les particules de l'espace de recherche ;
5. Initialiser toutes les fitness à zéro ;
6. **for** *p=1 à taille-essaim* **do**
   - 6.1. convertir la particule (numéro de règle) en représentation binaire
   - 6.2. appliquer la règle à l'image en entrée pixel par pixel, selon la règle de transition définie dans (5.3.3)
   - 6.3. calculer les fitness
   - 6.4. identifier la meilleure position $P_{gt}$
   - 6.5. calculer la moyenne des meilleures positions $mbest(t)$ avec l'équation (5.6)

**end**
/*fin initialisation de l'essaim*/
/*application de l'algorithme QPSO*/
7. **while** *nombre max d'itérations non atteint* **do**
   - 7.1 diminuer beta linéairement de 1.0 à 0.5
   - 7.2 calculer mbest
   - **for** *chaque particule* **do**
     - f = Rand(0,1) ;
     - Pd=f*pi+(1-f)*pg(d) ; u = Rand ;
     - **if** *u>0.5* **then**
       - xi = int16(pd - beta * abs(mbest(d) - xi)* log(1/u)) ;
       - **else**
         - xi = int16(pd + beta * abs(mbest(d) - xi)* log(1/u)) ;
       - **end**
     - **end**
     - Particule(p).x(d)= xi ;
     - application de la nouvelle règle $x_i$ à l'image
     - mise à jour de la fitness des particules
   - **end**

**end**
Afficher meilleure règle, meilleur contour, meilleure fitness.

**Algorithm 11:** L'algorithme QPSO pour la détection de contours

### 5.3.7 Résultats expérimentaux

Cette section présente quelques résultats issus de l'application de l'algorithme QPSO sur plusieurs images. QPSO a prouvé son efficacité lors de l'extraction efficace des meilleures règles qui aboutissent à un bon contour. L'évaluation de performances des résultats obtenus est faite en comparaison avec d'autres détecteurs de contours connus (Canny, Sobel, Prewitt) et les contours de référence (vérité terrain), obtenus de la base de Berkeley.

#### 5.3.7.1 Meilleur paquet de règles

Les expériences conduites sur des dizaines d'images différentes (synthétiques, binaires, niveaux de gris,...etc) montrent que quatre meilleures règles émergent. Elles donnent de bons contours, après application d'une seule itération sur l'image en entrée. Ces règles sont : règle 56, règle 120, règle 112, règle 116.
Il est important de souligner que, une fois les meilleures règles connues, elles peuvent être appliquées directement sur une image, et aboutir rapidement au résultat voulu.

#### 5.3.7.2 Résultats visuels

Les résultats de la figure 5.5 montrent clairement que la règle 112, extraite par l'algorithme QPSO, produit des résultats satisfaisants, comparés à Canny, Sobel et Prewitt. Les contours sont continus, nets et fins.

Dans la figure 5.6, la détection de contours par la règle 112 a donné des résultats acceptables, en comparaison avec Canny, Sobel et Prewitt. Le contour externe des pièces de monnaie est précis, continu, et sans bruit. Toutefois, les détails internes des pièces apparaissent dans une seule pièce.

Dans les figures 5.7, 5.8, 5.9 où l'étalon de comparaison est un contour fait à la main, obtenu à partir de la base de Berkeley, on voit clairement que la règle 112 donne de bons contours, avec un fin niveau de détails, particulièrement dans le corps de l'oiseau, l'ombre et le cygne.
Dans l'image 5.9, la règle 112 donne plus de précision sur le nez, les mains, le pull.

Le tableau 5.4 montre les meilleures valeurs de fitness obtenues pour les images : monnaie, oiseau, cygne, et chinoise. Pour chaque image, les détecteurs de Canny et Sobel sont testés, avec les trois fonctions de fitness préalablement définies : la distance de Hamming, SSIM et RMSE. Pour Canny et Sobel, une seule itération suffit pour relever les valeurs de fitness.
QPSO est testé sur 25 exécutions, avec 40 générations. Les meilleures valeurs de fitness sont recueillies. Les résultats numériques montrent clairement que QPSO fournit des performances égales ou supérieures à Canny et Sobel.

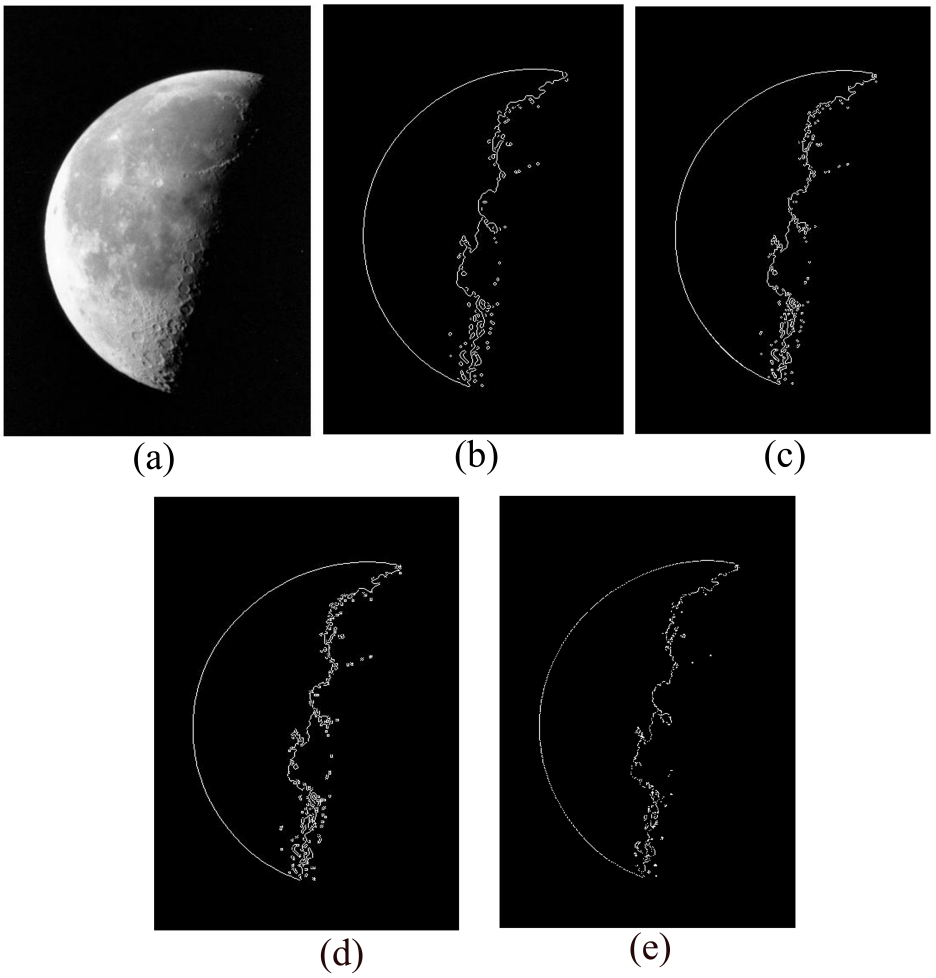


FIGURE 5.5 – Résultats visuels de QPSO et comparaison. a) Image Originale (b) Contour de Canny (c) Contour de Sobel d) Contour de Prewitt e) Contour QPSO règle 112.

#### 5.3.7.3 Comparaison avec des travaux similaires

Dans le papier [NRS15], les auteurs ont utilisé 2 approches pour trouver les contours sur l'image de Lena : Neural Network with general back propagation algorithm (BP-NN) et Neural Network with PSO (PSO-NN). Les résultats sont montrés à la figure 5.10.
Les résultats montrent clairement que l'algorithme QPSO produit un bon contour, et des détails sur le chapeau, le visage et la bouche.

Dans table 5.5, l'entropie est la mesure utilisée pour différencier l'efficacité des trois algorithmes. L'entropie est une mesure statistique utilisée pour caractériser la texture de l'image originale. Elle est calculée selon la formule suivante :

$$Entropy = -\Sigma p_i log(p_i)$$

où $p_i$, la probabilité de différence entre deux pixels adjacents, est égale à i. La valeur de l'entropie obtenue par QPSO est très inférieure par rapport à celles obtenues par les algorithmes

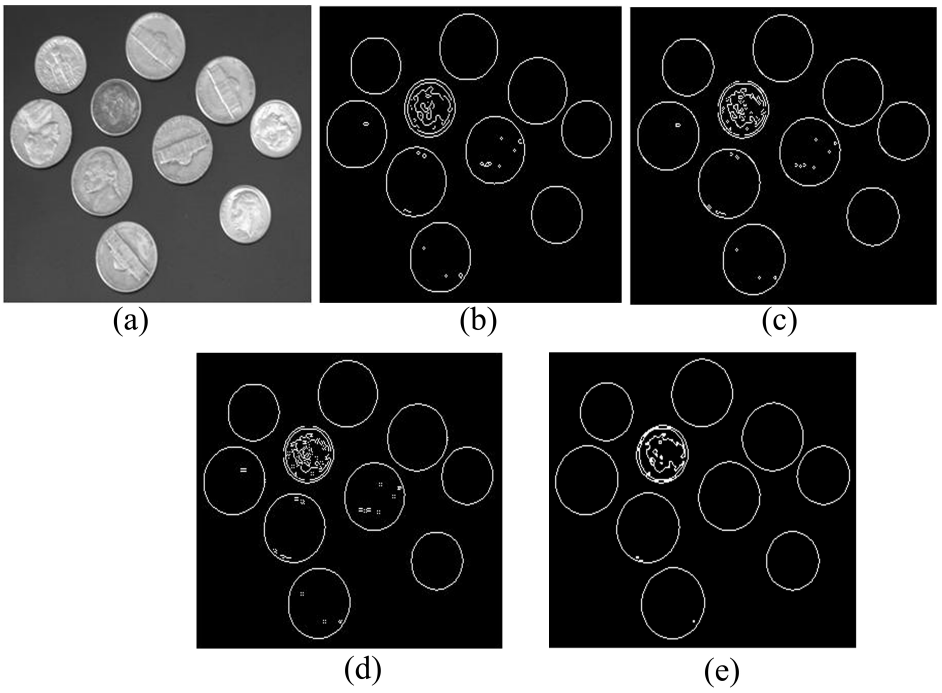


FIGURE 5.6 – Résultats visuels de QPSO et comparaison. (a)Image originale (b) Contour de Canny (c) Contour de Sobel (d) Contour de Prewitt (e) Contour QPSO règle 112.

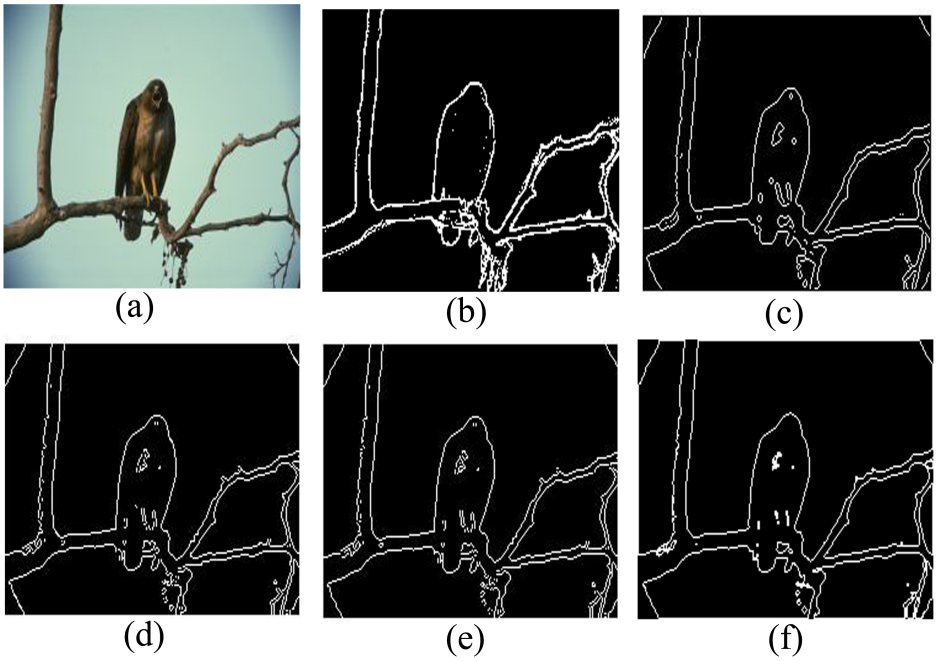


FIGURE 5.7 – Résultats visuels de QPSO et comparison. (a) Image originale (b) Contour de Berkeley (c) Contour de Canny (d) Contour de Sobel (e) Contour de Prewitt (f) Contour QPSO règle 112.

BP-NN et PSO-NN (voir la table 5.5 ), ce qui indique que le bruit est filtré, les contours sont bien définis, nets et précis. La règle 68 produit un nombre inférieur de contours discontinus, par rapport à BP-NN et PSO-NN.

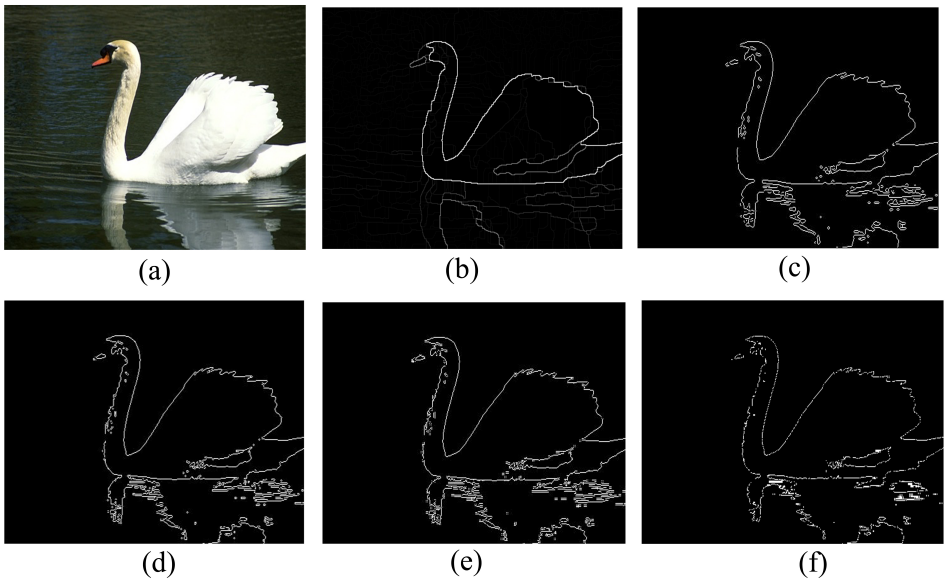


FIGURE 5.8 – Résultats visuels de QPSO comparaison. (a) Image originale (b) Contour de Berkeley (c) Contour de Canny (d) Contour de Sobel (e) Contour de Prewitt (f) Contour QPSO règle 112.

TABLE 5.4 – Meilleurs résultats de Fitness pour 4 images

| | QPSO | | | CANNY | | | SOBEL | | |
|---|---|---|---|---|---|---|---|---|---|
| Image | HD | SSIM | RMSE | HD | SSIM | RMSE | HD | SSIM | RMSE |
| Oiseau | **1482** | **0.99933** | **0.194428** | 1968 | 0.99934 | 0.2240 | 1677 | 0.99944 | 0.2068 |
| Chinoise | **1322** | **0.9985** | **0.2482** | 3586 | 0.9716 | 0.2751 | 2136 | 0.9992 | 0.2413 |
| Cygne | **3894** | **0.9994** | **0.1580** | 5476 | 0.9993 | 0.1874 | 4043 | 0.9994 | 0.1610 |
| Monnaie | **1575** | **0.9997** | **0.1450** | 2037 | 0.9995 | 0.1649 | 2030 | 0.9995 | 0.1646 |

TABLE 5.5 – Comparaison de BP-NN, PSO-NN et QPSO pour la détection de contours

| Algorithme | entropie | nombre de contours |
|---|---|---|
| BP-NN | 1.2335 | 50625 |
| PSO-NN | 0.2892 | 50625 |
| QPSO | **0.2411** | **10502** |

Considérons le papier [VS15], les auteurs ont utilisé l'algorithme ACO (Ant Colony Optimization) pour extraire les caractéristiques faciales sur les images de visages.

On peut voir sur la figure 5.11 que l'algorithme QPSO peut extraire de façon optimale, les caractéristiques "invariant illumination", à partir d'images de visage. Il est clair que les yeux, le nez et la bouche apparaissent de façon plus précise dans le résultat de QPSO.

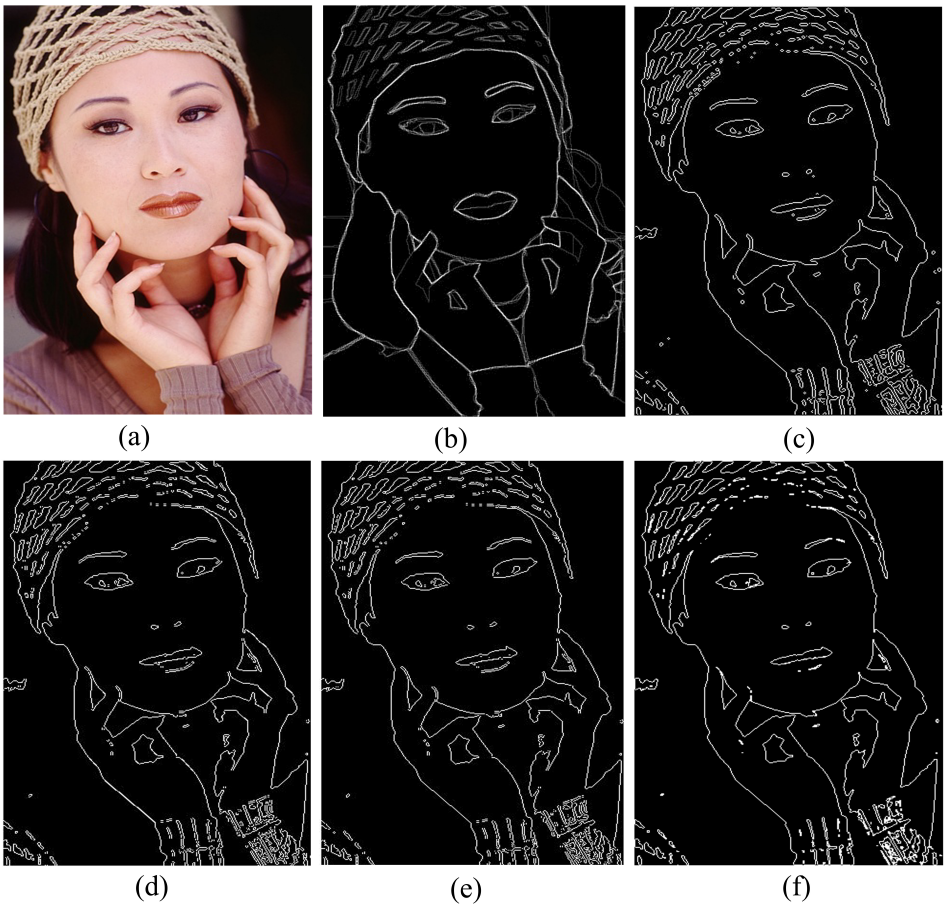


FIGURE 5.9 – Résultats visuels de QPSO et comparaison. (a) Image originale (b) Contour de Berkeley (c) Contour de Canny (d) Contour de Sobel (e) Contour de Prewitt (f) Contour QPSO règle 112.

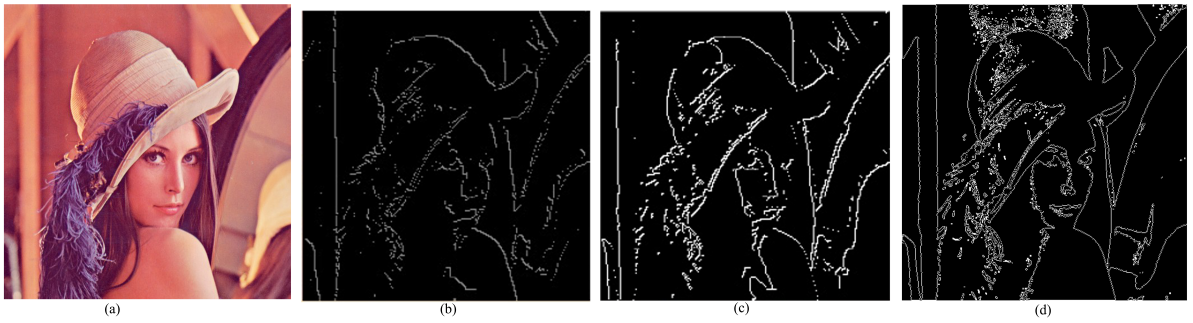


FIGURE 5.10 – Résultats Visuels de QPSO et comparaison. (a) Image originale (b) Contours de BP-NN (c) Contours de PSO-NN (d) Contours de QPSO règle 68.

### 5.3.8 L'algorithme QPSO pour le filtrage d'images

L'algorithme QPSO pour la détection de contours a été modifié de sorte à traiter le filtrage d'images. Dans ce paragraphe, des exemples sont présentés pour illustrer l'efficacité de la méthode. Nous utilisons des images binaires pour faire évoluer QPSO pour le filtrage de bruit.

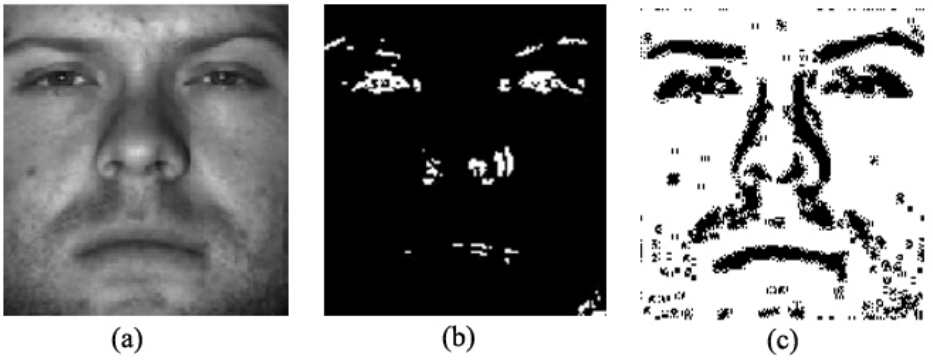


FIGURE 5.11 – Résultats visuels de QPSO et comparaison. (a) Image originale (b) Résultat de ACO ; (c) Résultat de QPSO règle 21.

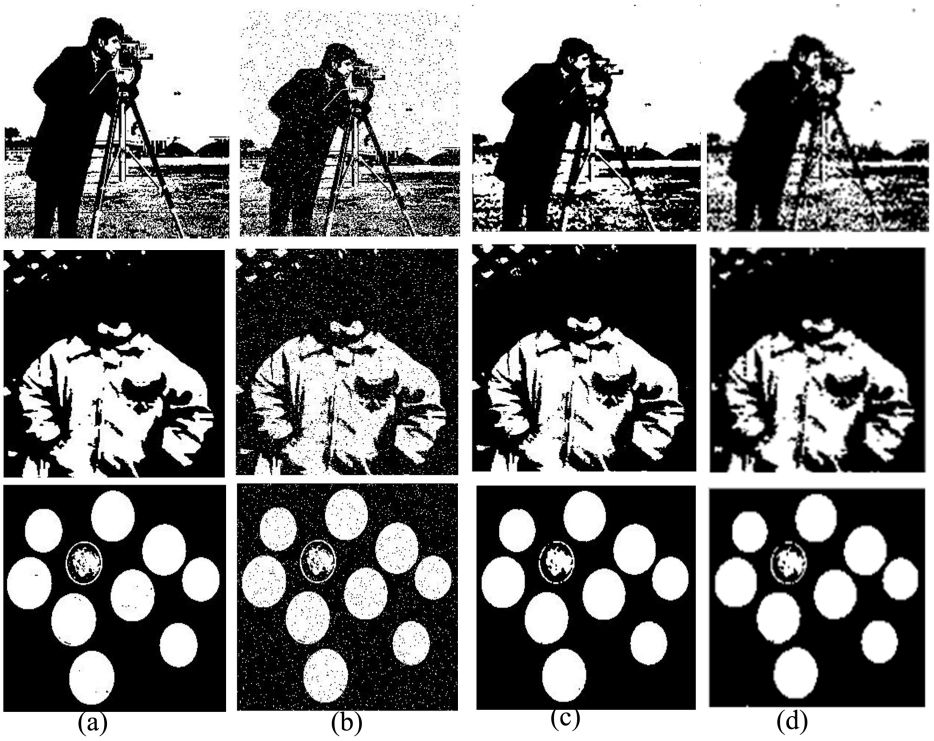


FIGURE 5.12 – Résultats visuels de QPSO et comparaison. (a) Image originale (b) Image bruitée (c) Résultat du filtre Median d) Résultat QPSO (règle 63 et règle 31).

L'image originale est prise comme image de référence. L'image bruitée est obtenue en ajoutant un bruit "sel et poivre 0.05" à l'image originale. Plusieurs expériences sont conduites, en variant le nombre d'itérations et la taille de la population. La figure 5.12 illustre les résultats de l'application de QPSO sur trois images : Cameraman, Pout et monnaie. La comparaison est faite avec le filtre médian. A partir de ces résultats, on peut voir clairement que QPSO a permis d'extraire des règles qui opèrent un filtrage aussi efficace que le filtre médian. La règle 31 donne de bons résultats sur l'image "cameraman", alors que la règle 63 est plus efficace sur les images "Pout" et "monnaie". Le bruit est filtré efficacement avec la simple utilisation des

TABLE 5.6 – Meilleures Fitness pour les 3 images

| | QPSO | | Filtre Median | |
|---|---|---|---|---|
| Image | HD | SSIM | HD | SSIM |
| Cameraman | **367** | **0.9986** | 370 | **0.9986** |
| Pout | **300** | **0.9997** | **300** | **0.9997** |
| Monnaie | **252** | 0.9997 | 278 | **0.9999** |

règles émergentes 31 et 63.
Le tableau 5.6 montre les meilleures fitness pour les 3 images testées. QPSO est exécuté 10 fois, le nombre d'itérations est 50, la taille de la population est 30. La comparaison est faite avec le filtre médian, connu pour être l'un des plus performants de la littérature. Les fonctions de fitness utilisées sont SSIM et la distance de Hamming.

Pour l'image "Pout", on a obtenu les mêmes valeurs de SSIM et distance de Hamming, ce qui signifie que les résultats du filtre médian et de la règle 63 sont très proches. Pour l'image cameraman, les valeurs de SSIM sont les mêmes pour le filtre médian et la règle 31, alors que la distance de Hamming est meilleure pour la règle 31. Pour l'image "Monnaie", la valeur de HD est légèrement meilleure pour la règle 31, alors que le résultat de SSIM est légèrement meilleur pour le filtre médian, ce qui signifie que les deux résultats sont très proches.

## 5.4 Conclusion

Dans ce chapitre, nous avons présenté deux algorithmes de résolution de l'émergence inversée. Le premier algorithme se base sur la métaheuristique PSO. L'image est modélisée par un système complexe : l'automate cellulaire, qui évolue avec des règles de transition d'une configuration initiale à une configuration finale, qui est l'image segmentée en contours. Le processus évolutionnaire guidé par PSO, extrait les bonnes règles qui réalisent la tâche finale désirée, et fournit ainsi une solution satisfaisante au problème d'émergence inversée. Les règles extraites ont donné de bons résultats, en comparaison avec les détecteurs de contour standards, et avec les résultats de l'algorithme génétique.
Le deuxième algorithme se base sur la métaheuristique PSO combinée avec les principes de l'informatique quantique : le Quantum PSO. Cet algorithme présente plusieurs avantages par rapport à la version standard de PSO : parallélisme lors de la recherche de solutions dans un large espace, le modèle utilise une seule équation, il y'a un seul paramètre à régler. Les expériences menées sur plusieurs types d'images, pour deux tâches de traitement d'images : détection de contours et filtrage, ont démontré que l'algorithme QPSO est très approprié et adéquat pour extraire les bonnes règles lors du processus évolutif de l'AC. Les résultats fournis ont été comparés avec les filtres classiques, et d'autres travaux connexes et se sont avérés très satisfaisants et prometteurs.

Chapitre 6

# L'Algorithme Génétique Quantique pour la Résolution de l'Emergence Inversée

## 6.1 Introduction

Dans ce chapitre, nous présentons une nouvelle approche pour la résolution de l'émergence inversée basée sur le "Quantum Genetic Computing" et les systèmes complexes. L'application se situe dans le domaine de traitement d'images, plus particulièrement la détection de contours. L'idée principale est d'utiliser l'automate cellulaire (AC) comme système complexe pour modéliser l'image, et l'algorithme génétique quantique comme stratégie de recherche pour l'optimisation. L'automate cellulaire est une grille de cellules qui opèrent en parallèle et interagissent localement avec leurs voisins, en utilisant des règles de transition simples et produisant une structure globale exhibant de nouvelles fonctionnalités. Le problème avec les automates cellulaires est de trouver, parmi un large ensemble de règles de transition, le sous-ensemble qui mène à l'effet attendu, dans notre cas, la détection de contours. Pour aborder ce problème difficile, nous proposons l'utilisation d'un algorithme génétique quantique (QGA)pour faire évoluer l'automate cellulaire. La configuration finale de l'AC est une image segmentée en contours. L'efficacité et la faisabilité de QGA sont démontrées par des résultats visuels et quantitatifs. La comparaison avec l'algorithme génétique conventionnel (CGA) est faite. Les expériences prouvent que l'efficacité de QGA est significativement meilleure que son homologue conventionnel (CGA). Les résultats obtenus sont encourageants et prometteurs.

## 6.2 Etat de l'art

Le Quantum Computing (QC) est une science interdisciplinaire émergente qui a induit une recherche intensifiée dans la dernière décennie. QC est basé sur les principes de la mécanique quantique comme la représentation en *qubit* et la *superposition d'états*. QC est capable de traiter un grand nombre d'états en parallèle, il apporte une nouvelle philosophie à l'optimisation, en regard des concepts qui lui sont inhérents [WC98]. Le premier algorithme quantique a été proposé par [Sho94] pour la factorisation de nombres. [Gro96] a proposé un algorithme quantique pour la recherche aléatoire dans les bases de données. Des chercheurs ont essayé d'adapter quelques propriétés du Quantum Computing dans les algorithmes classiques. Depuis la fin des années 1990, l'hybridation du calcul quantique et des algorithmes évolutionnaires s'est avérée être une solution efficace pour traiter des problèmes complexes. Dans ce contexte, l'algorithme génétique quantique (QGA) a prouvé son efficacité pour la résolution de problèmes d'optimisation [HK02] , [Han03] , [DTB05]. QGA peut assurer l'équilibre entre exploration et exploitation plus facilement que l'algorithme génétique classique. D'autre part, il peut explorer l'espace de recherche avec un nombre d'individus réduit pour obtenir une solution globale en un temps minimal [HK04]. QGA est aussi caractérisé par la représentation des individus, la taille de la population, la fonction de fitness et la dynamique de la population.
En plus de la mutation et du croisement, un nouvel opérateur incluant le concept d'interférence introduit par [Nar99] est aussi utilisé. QGA présente plusieurs avantages : une petite taille de population, la vitesse de convergence, une grande aptitude à l'optimisation globale, et une bonne robustesse.
L'intérêt de ce travail est double :
D'une part, nous proposons une nouvelle approche combinant le puissant principe de Quantum Computing avec un système complexe : l'automate cellulaire, pour traiter le problème de la détection de contours. A notre connaissance, il n'existe pas dans la littérature des travaux combinant automate cellulaire et QGA, pour résoudre le problème de la détection de contours. QGA a été efficacement introduit pour résoudre des problèmes d'optimisation combinatoire [HK00], optimisation de fonctions [WLZF13]. En Bio-informatique, QGA est exploité pour prédire les nouvelles séquences de protéines [LMB06]. Dans notre domaine d'intérêt, le traitement d'images, il existe très peu de travaux traitant les tâches de traitement d'images par QGA. [ZZJS11] propose un algorithme génétique quantique pour la segmentation d'images basé sur l'entropie maximale. Dans [TBD07] les auteurs exposent un algorithme génétique quantique pour la segmentation multi-objective avec la stratégie Split/Merge. [CH13] présentent des algorithmes de clustering pour la segmentation d'images.
D'autre part, l'usage de QGA comme stratégie de recherche dans un large espace de règles de transition a donné une solution efficace au problème difficile qui consiste à trouver le sous-ensemble de règles qui mène à la fonction désirée, et offre donc une bonne issue au problème de l'émergence inversée.
Des techniques pour automatiser la recherche des règles ont été proposées. [Sip97] a appliqué

des règles évolutionnaires pour exécuter des tâches simples, [Ros06] a introduit l'algorithme SFFS pour faire évoluer un AC qui effectue des tâches de traitement d'images. [SBM07] et [KS11] ont utilisé l'algorithme génétique classique pour faire évoluer un automate cellulaire pour la détection de contours. Dans [BMAH09], l'émergence inversée et un algorithme génétique sont combinés pour le filtrage d'images.
Même si ces algorithmes ont obtenu de bons résultats, ils demeurent coûteux et ne conviennent pas à des problèmes plus diversifiés [KS00].
Ce domaine reste ouvert à la recherche, on essaie d'y apporter une contribution en proposant l'usage de l'algorithme génétique quantique comme stratégie de recherche pour trouver les règles d'un AC dans un grand espace, afin de réaliser une détection de contours. L'algorithme proposé dans ce chapitre tire profit des capacités de QGA à explorer un large espace de recherche et sa convergence rapide, et la puissance de l'automate cellulaire pour modéliser l'image efficacement et extraire le sous-ensemble de règles qui réalise la détection de contours.

## 6.3 Généralités sur le Quantum Genetic Computing

L'origine du Quantum Computing remonte au début des années 80 quand Richard Feynman a observé que les effets mécaniques du quantique ne peuvent pas être simulés efficacement sur un ordinateur. Durant la dernière décennie, le Quantum Computing a attiré un grand intérêt et a induit de larges recherches, il s'est avéré plus puissant que la programmation classique. D'autre part, le Quantum Computing a pour principale qualité le parallélisme, qui réduit considérablement la complexité algorithmique. Cette capacité de traitement parallèle peut être exploitée pour résoudre des problèmes d'optimisation qui requièrent l'exploration de larges espaces de solutions.
Le qubit [Hey99] est la plus petite unité d'information stockage dans un ordinateur quantique à 2 états. Contrairement au bit classique qui a 2 valeurs possibles : 0 ou 1, un qubit sera la superposition de ces 2 valeurs. L'état d'un qubit peut être représenté par la notation de Dirac :

$$|\psi = \alpha|0> + \beta|1> \tag{6.1}$$

où $\alpha$ et $\beta$ sont des nombres complexes qui spécifient la probabilité d'amplitude correspondant à chaque état.
$|0>$ et $|1>$ représentent respectivement les valeurs classiques des bits 0 et 1. Lorsqu'on mesure l'état d'un qubit, on peut obtenir $'0'$ avec une probabilité $|\alpha|^2$ et on peut obtenir $'1'$ avec une probabilité $|\beta|^2$.

$$|\alpha|^2 + |\beta|^2 = 1 \tag{6.2}$$

Un système de $m$-qubits peut représenter $2^m$ états simultanément. Un ordinateur quantique peut traiter toutes ces valeurs en parallèle. Lors de l'observation d'un état quantique, il

converge vers un état unique parmi les autres [Nar99]. Un algorithme quantique consiste à appliquer successivement une série d'opérations quantiques sur le système. Les opérations quantiques sont réalisées avec des circuits et des portes quantiques. Il est important de souligner que la conception d'algorithmes quantiques n'est pas chose aisée. Jusqu'à présent il n'existe pas de machine quantique capable d'exécuter les programmes quantiques [A.10].

### 6.3.1 Principes de l'Algorithme Génétique Quantique

Un algorithme génétique quantique (QGA) est un algorithme génétique où les individus manipulés sont des chromosomes quantiques. La représentation des chromosomes est basée sur le concept de qubit, et dotée d'opérations quantiques. Cela induit que les opérations génétiques classiques (croisement, mutation) seront totalement redéfinies pour s'adapter à la nouvelle représentation des chromosomes.
La structure de QGA est décrite dans l'algorithme 12

**Data**:
**Output**:

1. Q(t=0) est une population de chromosomes quantiques à la génération t.
2. P(t) est un ensemble de solutions binaires à la génération t.
3. DEBUT
4. Génération aléatoire de la population initiale des chromosomes quantiques.

Répéter jusqu'à la convergence :
* Générer P(t) par mesure de Q(t)
* Evaluer P(t)
* Sauvegarder la meilleure solution b
* Mise à jour de Q(t) par interférence quantique
* $t \leftarrow t+1$

FIN

**Algorithm 12:** Pseudo-code de l'algorithme génétique quantique

### 6.3.2 Codage des Chromosomes Quantiques

Un chromosome quantique est une chaine de $n$ qubits, qui forment un registre quantique. La table 6.1 montre la structure du chromosome quantique.

TABLE 6.1 – Structure d'un chromosome quantique

| | | | | |
|---|---|---|---|---|
| $\alpha_0$ | $\alpha_1$ | $\alpha_2$ | ... | $\alpha_n$ |
| $\beta_0$ | $\beta_1$ | $\beta_2$ | ... | $\beta_n$ |

### 6.3.3 La Mesure des Chromosomes

Dans le but d'exploiter efficacement la superposition des états dans un qubit, on a besoin de faire une lecture pour chaque bit. Cette opération conduit à extraire un chromosome binaire à partir d'un chromosome quantique. Le but est de permettre l'évaluation de la population d'individus, en fonction des chromosomes binaires extraits (voir table 6.2.)

TABLE 6.2 – Extraction d'un chromosome binaire

**Chromosome Quantique**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| $\alpha_0$ | $\alpha_1$ | $\alpha_2$ | $\alpha_3$ | $\alpha_4$ | $\alpha_5$ | ... | $\alpha_n$ |
| $\beta_0$ | $\beta_1$ | $\beta_2$ | $\beta_3$ | $\beta_4$ | $\beta_5$ | ... | $\beta_n$ |

$\Downarrow$ *Mesure*

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 0 | 1 | ... | 1 |

**Chromosome Binaire**

La fonction de mesure est donnée par le pseudo-code de l'algorithme 13

**Data**:
**Output**:

1. Soient Q un qubit et 'mesure' sa fonction de mesure.
2. Q est défini par $|\psi = \alpha|0> + \beta|1>$ où $|\alpha|^2 + |\beta|^2 = 1$
3. DEBUT
4. r = prendre r aléatoirement dans [0,1].

**if** $r > \alpha^2$ **then**
    retourner 1 **else**
        r
    **end**
    etourner 0
**end**
FIN.

**Algorithm 13:** La fonction mesure

Chaque solution binaire est évaluée pour donner la valeur de fitness. Dans l'étape suivante, un ensemble de chromosomes quantiques est mis à jour par application de quelques portes quantiques, utilisant les solutions binaires et la meilleure solution trouvée. Il existe plusieurs types de portes quantiques : porte Not, porte Controlled Not, porte de Hadamard, porte par rotation. Le choix de la porte quantique se fait en fonction du type de problème traité.

## 6.4 L'Approche Proposée

Afin de mettre en pratique notre approche de résolution de l'émergence inversée, on se propose de traiter le problème de détection de contours sur images. Dans cette optique, nous proposons la modélisation de l'image à l'aide d'un système complexe : l'automate cellulaire (AC). Il a été utilisé efficacement en traitement d'image, mais il présente un réel problème : la génération des règles de transition est laborieuse et lente.
On va décrire comment QGA est utilisé pour faire évoluer l'AC afin de réaliser une détection de contours sur images. Les images en entrée de l'AC sont des images couleur, mais dans un souci de simplicité, elles seront binarisées, de sorte que l'état de sortie d'un chromosome binaire sera 0 ou 1. Les cellules ont 2 valeurs possibles : blanc ou noir. Chaque cellule a 8 voisins (voisinage de Moore). L'automate cellulaire prend en entrée l'image originale et donne après quelques itérations l'image finale. Travailler avec des images binaires implique que toutes les combinaisons possibles du voisinage donnent $2^8$ configurations possibles ou règles. Après élimination des symétries, le nombre de règles diminue à 51 [Ros06]. L'application de ces règles séparément sur un pixel central noir, puis sur un pixel central blanc donne 102 règles. Le problème consiste à extraire parmi cet ensemble, le sous-ensemble de règles capables de réaliser l'effet attendu, sachant que le nombre de toutes les combinaisons possibles est de l'ordre de $5 * 10^{30}$.
Dans cette optique, on applique l'algorithme QGA comme suit[DB18], [DB16] :
L'algorithme emploie un nombre réduit de chromosomes quantiques, chaque chromosome est une chaine de 102 qubits, représentant l'espace de recherche. Initialement, tous les qubits ont pour valeur $\frac{1}{\sqrt{2}}$ (table 6.3 ).

TABLE 6.3 – Mesure de chromosome

**Chromosome Quantique**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| $\frac{1}{\sqrt{2}}$ | $\frac{1}{\sqrt{2}}$ | $\frac{1}{\sqrt{2}}$ | $\frac{1}{\sqrt{2}}$ | $\frac{1}{\sqrt{2}}$ | $\frac{1}{\sqrt{2}}$ | ... | $\frac{1}{\sqrt{2}}$ |
| $\frac{1}{\sqrt{2}}$ | $\frac{1}{\sqrt{2}}$ | $\frac{1}{\sqrt{2}}$ | $\frac{1}{\sqrt{2}}$ | $\frac{1}{\sqrt{2}}$ | $\frac{1}{\sqrt{2}}$ | ... | $\frac{1}{\sqrt{2}}$ |

$\Downarrow$ *Mesure*

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 0 | 1 | ... | 1 |

**Chromosome Binaire**

La fonction de mesure (voir paragraphe 3.3) est appliquée sur les chromosomes quantiques pour extraire des chromosomes binaires classiques, qui représentent la valeur de sortie de la règle de l'AC. L'étape suivante consiste à évaluer ces solutions. Chaque chromosome binaire est exécuté sur l'image originale, pour générer l'image résultat, cette image est comparée à une image de référence pour déterminer sa fitness, en se basant sur une fonction de mesure

d'erreur. Dans cet algorithme, trois fonctions de mesure d'erreur sont utilisées : Root Mean Square Error (RMSE), Hamming distance (HD) et Structural Similarity Index (SSIM). L'opération suivante consiste à mettre à jour les chromosomes, avec l'interférence quantique. Un chromosome quantique est mis à jour en utilisant une porte de rotation $U(\theta)$, définie comme suit :

$$U(\theta) = \begin{pmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{pmatrix} \tag{6.3}$$

où $\theta$ est l'angle de rotation. Cette étape a pour effet de faire converger le chromosome vers la valeur la plus ajustée. Dans l'étape suivante, la meilleure solution de la population est sélectionnée, si cette solution est meilleure que la meilleure solution stockée précédemment, elle remplace alors cette dernière. La valeur du i-eme qubit ($\alpha_i$,$\beta_i$) est mise à jour comme suit :

$$\begin{pmatrix} \alpha_i\prime \\ \beta_i\prime \end{pmatrix} = \begin{pmatrix} cos(\theta_i) & -sin(\theta_i) \\ sin(\theta_i) & cos(\theta_i) \end{pmatrix} \begin{pmatrix} \alpha_i \\ \beta_i \end{pmatrix} \tag{6.4}$$

Les paramètres utilisés dans ce problème sont montrés dans la table 6.4. $x_i$ et $b_i$ sont les i-eme bits de $x$ et $b$ (b est la meilleure solution). La valeur de $\Delta\theta_i$ a un effet sur la vitesse de convergence, mais si elle est trop grande, les solutions peuvent diverger ou converger prématurément vers un optimum local. Le signe $s(\alpha_i,\beta_i)$ détermine la direction de convergence vers un optimum global. La table de correspondance est utilisée comme stratégie de rotation des portes quantiques.

TABLE 6.4 – Table de correspondance pour la rotation des portes quantiques

| $x_i$ $b_i$ | $f(x) >= f(b)$ | $\Delta\theta_i$ | $s(\alpha_i, \beta_i)$ | | | |
|---|---|---|---|---|---|---|
| | | | $\alpha_i\beta_i > 0$ | $\alpha_i\beta_i < 0$ | $\alpha_i = 0$ | $\beta_i = 0$ |
| 0 0 | False | $0.005\pi$ | − | + | ± | ± |
| 0 0 | True | $0.005\pi$ | − | + | ± | ± |
| 0 1 | False | $0.08\pi$ | − | + | ± | ± |
| 0 1 | True | $0.005\pi$ | − | + | ± | ± |
| 1 0 | False | $0.08\pi$ | + | − | ± | ± |
| 1 0 | True | $0.005\pi$ | + | − | ± | ± |
| 1 1 | False | $0.005\pi$ | + | − | ± | ± |
| 1 1 | True | $0.005\pi$ | + | − | ± | ± |

L'étape suivante est la mise à jour des meilleures solutions locale et globale. A la fin du processus itératif, le meilleur contour obtenu et la meilleure fitness sont montrés.

## 6.5 Résultats Expérimentaux

Les expériences on été menées sur des images du benchmark de Berkeley (BSDS) qui fournit pour chaque image sa réplique segmentée tracée par un expert humain. La taille de la population est 50 individus. La mesure de performance de l'algorithme a relevé la meilleure valeur trouvée après 100 générations. La figure 6.1 montre les résultats de l'algorithme proposé sur 3 images de BSDS : Oiseau, Chinoise et Montagne. Il est clair que l'algorithme QGA produit de bons contours, en comparaison avec l'image vérité terrain, et avec un filtre de détection de contours connu : Canny.

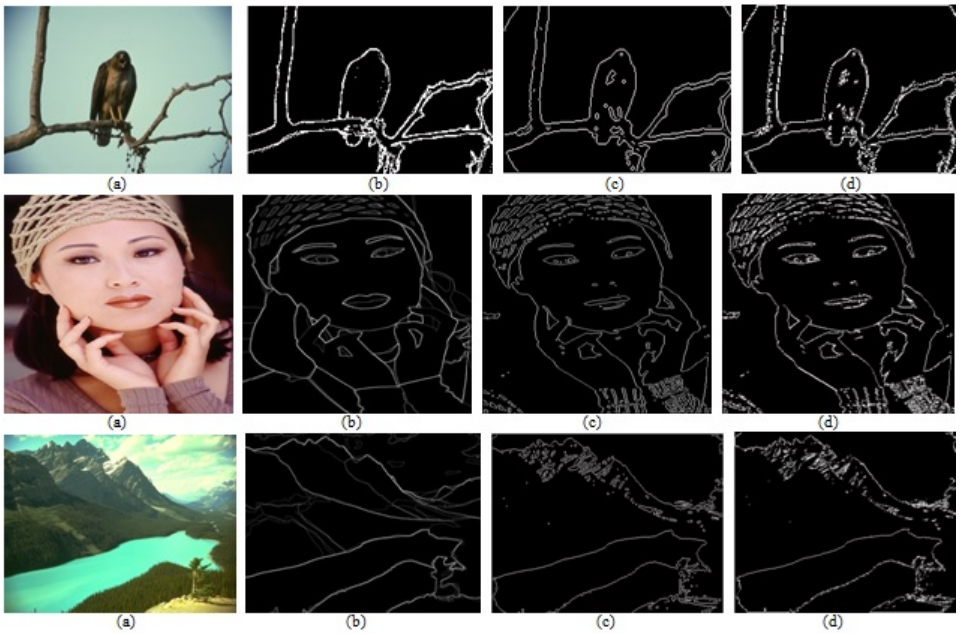


FIGURE 6.1 – Résultats visuels de QGA et comparaison (a) Image Originale (b) Vérité terrain (c) Contour de Canny (d) Contour de QGA

Les fonctions de Fitness utilisées sont : la distance de Hamming (HD), Root Mean Square Error (RMSE), et Structural Similarity Index (SSIM).
La distance de Hamming (HD) relève le nombre de pixels différents entre 2 images. La fitness F est alors calculée selon l'équation (6.5) :

$$F = \frac{1}{1 + HD} \tag{6.5}$$

L'erreur RMSE est calculée selon la formule de l'équation (6.6) :

$$RMSE = \sqrt{\frac{1}{MN} \sum_{r=0}^{M-1} \sum_{c=0}^{N-1} [E(r,c) - O(r,c)]^2} \tag{6.6}$$

où $O(r, c)$ est l'image originale (dans notre cas l'image de référence), $E(r, c)$ est l'image résultat obtenue (le résultat de QGA).
L'index SSIM entre 2 images $x$ et $y$ est défini par : [WBSS04]

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{6.7}$$

où $\mu_x, \mu_y, \sigma_{xy}, \sigma_x^2, \sigma_y^2$, sont respectivement la moyenne de $x$, la moyenne de $y$, la variance de $x$, la variance de $y$, et la covariance de $x$ et $y$. Selon [WBSS04], $C_1$ est initialisé à $(0.01 * 255)^2$ et $C_2 = (0.03 * 255)^2$.

La table 6.5 montre la meilleure valeur de fitness obtenue pour les 3 images illustrées en figure (6.1). Pour chaque image, l'algorithme QGA et le filtre de Canny sont testés, avec les 3 fonctions fitness : distance de Hamming, RMSE et SSIM.
Pour Canny, une seule itération est suffisante pour récupérer le résultat de la fitness. L'algorithme QGA est testé sur 25 exécutions, avec respectivement 20,40, et 100 itérations.

TABLE 6.5 – Meilleures valeurs de Fitness pour les 3 images

| | | QGA | | | CANNY | | |
|---|---|---|---|---|---|---|---|
| Image | iterations | F | SSIM | RMSE | F | SSIM | RMSE |
| Oiseau | 20 | $2.07e^{-4}$ | 0.9977 | 0.2767 | $4.56e^{-4}$ | 0.9765 | 0.3042 |
| | 40 | $2.12e^{-4}$ | 0.9971 | 0.2742 | - | - | - |
| | 100 | $1.98e^{-4}$ | 0.9966 | 0.2735 | - | - | - |
| Chinoise | 20 | $1.94e^{-4}$ | 0.9985 | 0.2482 | $3.87e^{-4}$ | 0.9716 | 0.2751 |
| | 40 | $1.77e^{-4}$ | 0.9984 | 0.2484 | - | - | - |
| | 100 | $1.68e^{-4}$ | 0.9976 | 0.2479 | - | - | - |
| Montagne | 20 | $1.59e^{-4}$ | 0.9992 | 0.2005 | $3.22e^{-4}$ | 0.9632 | 0.2483 |
| | 40 | $1.62e^{-4}$ | 0.9987 | 0.2132 | - | - | - |
| | 100 | $1.45e^{-4}$ | 0.9979 | 0.2014 | - | - | - |

## 6.6 Comparaison entre Algorithme Génétique Quantique et Algorithme Génétique Classique

Dans ce paragraphe, une comparaison visuelle et numérique entre les résultats de QGA et l'algorithme génétique classique (CGA) [KS11] est faite. CGA est un processus évolutionnaire pour extraire des règles de détection de contours. Il se base sur un algorithme génétique qui fait évoluer un AC sur plusieurs générations afin de réaliser la meilleure détection de contours. Dans la suite, on travaille sur les mêmes images utilisées dans [KS11]. QGA est appliqué sur ces images. L'image originale, l'image de référence, les résultats de CGA et QGA sont illustrés dans la figure 6.2, qui montre clairement les performances de QGA.

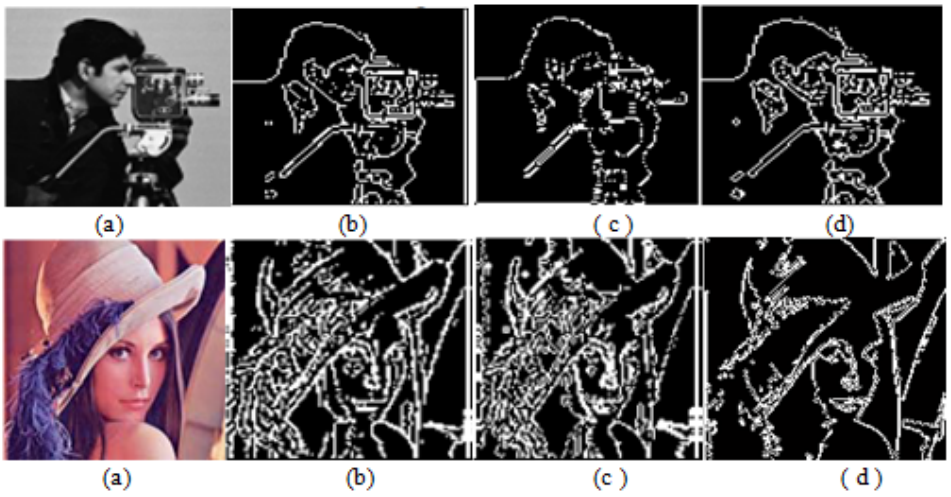


FIGURE 6.2 – Détection de contours pour les images Cameraman et Lena (a) Image Originale (b) Image de référence (c) Contour produit par CGA (d) Contour produit par QGA

Le résultat de QGA est visiblement meilleur, l'algorithme a permis d'extraire les contours de l'image originale avec une nette précision. Les résultats visuels sur les 2 images montrent que QGA donne un meilleur effet que CGA, la continuité des contours est plus forte, alors que CGA produit des résultats discontinus, et inclut de faux contours.

### 6.6.1 Résultats Numériques

Les expériences ont été réalisées sur plusieurs images, pour les deux algorithmes : CGA et QGA. La taille de la population pour l'algorithme génétique conventionnel est 100. Les probabilités de mutation et croisement sont fixées respectivement à 0.65 et 0.05. La taille de la population de QGA est 10. On a collecté les meilleures solutions sur 2000 itérations, et 25 exécutions.
Les résultats expérimentaux ont démontré que les performances de QGA sont largement supérieures à CGA. QGA donne de bons résultats même avec une petit nombre d'individus (10 chromosomes quantiques). QGA peut chercher des solutions dans le voisinage de l'optimum après un nombre d'itérations réduit, en comparaison avec CGA.

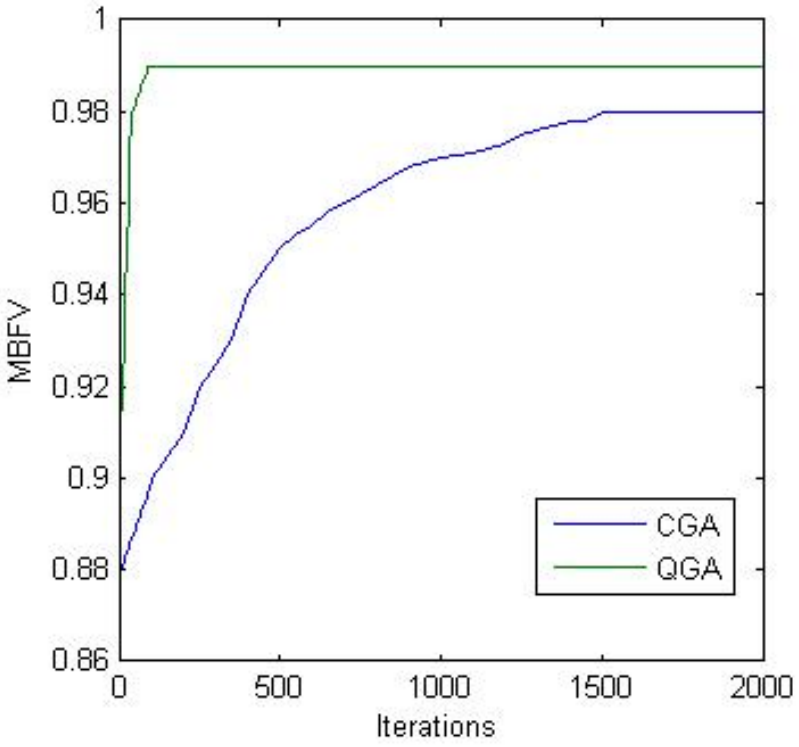


FIGURE 6.3 – Evolution de la valeur de la meilleure fitness (BFV) pour les algorithmes CGA et QGA au cours des itérations

La figure 6.3 montre la progression de la valeur moyenne de la meilleure fitness au cours de 2000 générations pour QGA et CGA. Il est clair que QGA dépasse beaucoup CGA en termes de rapidité de convergence et résultats finaux. Au début du graphique de BFV, CGA montre une lente convergence. Après 50 générations, CGA maintient un taux de convergence constant, alors que QGA affiche, dés le début du graphique, une meilleure rapidité de convergence que CGA. Ceci est dû à sa large capacité de recherche et d'exploration. Dés le départ, les deux algorithmes ont affiché un taux de convergence croissant mais CGA a maintenu une augmentation presque constante. Ceci est dû essentiellement à la convergence prématurée dont souffre l'algorithme génétique conventionnel, contrairement à QGA qui s'approche du voisinage de l'optimum global après 100 générations, la meilleure fitness est alors atteinte. Pour l'algorithme CGA, la meilleure valeur de fitness est 0.98, elle est atteinte après 1500 générations. Au delà de ce seuil, aucune amélioration n'est constatée.
Les expérimentations démontrent l'efficacité et l'applicabilité de QGA. En particulier, la figure 6.3 montre l'excellente capacité de recherche globale et la supériorité de convergence de QGA par rapport à CGA.

### 6.6.2 Complexité Algorithmique

De plus, les algorithmes quantiques ont généralement la capacité de minimiser la complexité algorithmique, sur les ordinateurs classiques. On peut faire une comparaison simple entre la complexité globale de QGA et celle de CGA pour estimer la réduction de complexité qu'on peut atteindre. Avec QGA, la complexité globale est de l'ordre de $O(N)$, N est la taille de la population (Evaluation + Interférence). Pour un CGA classique, la complexité est souvent de l'ordre de $O(N^2)$ (Evaluation + Sélection + Croisement + Mutation). En conséquence, nous concluons que ce résultat est très intéressant, vu que la complexité algorithmique a été nettement réduite, à devenir presque linéaire.

## 6.7 Conclusion

Dans ce chapitre, on a présenté un nouvel algorithme pour résoudre l'émergence inversée. Un système complexe puissant, l'automate cellulaire, modélise l'image. Un algorithme génétique quantique est utilisé pour explorer un large espace de recherche, formé par les règles de transition de l'AC, optimiser le processus, et extraire efficacement les règles qui réalisent une bonne détection de contours. Les résultats obtenus montrent que l'algorithme QGA est de loin, plus performant que son homologue classique. L'originalité de ce travail réside essentiellement dans la combinaison de ces trois concepts : le Quantum Computing, l'algorithme génétique, et l'automate cellulaire. La motivation essentielle est de tirer profit du parallélisme inhérent à QGA, durant l'exploration de l'espace de recherche, avec une efficacité maximale. Les expérimentations montrent que QGA est très prometteur pour l'optimisation de larges espaces de recherche, tout en conservant l'équilibre entre efficacité et performances. L'algorithme QGA a prouvé son efficacité et son adaptabilité dans le domaine de traitement d'images. Sa convergence vers la meilleure solution est rapide, et il a une bonne capacité de recherche globale. Un nombre réduit de chromosomes quantiques suffisent pour étudier le problème. L'usage de l'interférence quantique offre un outil puissant qui renforce la stabilité durant la recherche. Dans les perspectives futures, on pourrait envisager d'explorer les conséquences du changement des angles de rotation, du nombre de chromosomes, ...et leur impact sur les performances de l'algorithme.

# Conclusion et Perspectives

Dans cette thèse, nous avons exploré de nouvelles solutions efficaces pour résoudre le problème difficile de l'émergence inversée. Le problème a été appréhendé comme un problème d'optimisation, et les solutions proposées utilisent des métaheuristiques : PSO, QPSO et QGA. L'application des solutions apportées et leur validation s'est faite dans le domaine de traitement d'images : détection de contours et filtrage.
L'automate cellulaire est le système complexe utilisé pour modéliser l'image. Il évolue d'une configuration initiale qui est l'image originale, vers une configuration finale qui est l'image résultat, après application de règles de transition simples qui produisent l'effet désiré (détection de contours, filtrage). La résolution de l'émergence inversée revient à extraire le sous-ensemble de règles approprié, parmi un grand espace de recherche, qui aboutissent à la tâche attendue. Nous avons démontré la validité et la haute capacité des métaheuristiques utilisées (PSO, QPSO et QGA) pour la sélection de règles appropriées dans un large espace de recherche. Le processus itératif a abouti à l'extraction de règles simples et efficaces. Appliquées à différents types d'images, ces règles ont fourni des résultats très satisfaisants, en comparaison avec les meilleurs algorithmes de détection de contours, tels que Canny, Sobel et Prewitt, ou de filtrage (filtre médian). D'autres comparaisons ont été faites avec des travaux connexes, ayant utilisé les réseaux de neurones, les colonies de fourmis, l'algorithme génétique classique. Les algorithmes développés dans cette thèse se sont révélés compétitifs, robustes et performants.
Une fois les meilleures règles émergent, révélées par les algorithmes PSO, QPSO et QGA, on les applique directement sur les images originales, et on obtient ainsi les contours ou l'image filtrée en un temps minimal. Ceci démontre la flexibilité de la méthode, et représente un avantage clé lors de son application.

Nous pouvons explorer d'autres applications intéressantes à ce travail, dans le domaine de la réalité augmentée, particulièrement dans l'analyse de scènes, pour restaurer des données perdues après variation des conditions de prise de vue (météo mauvaise, éclairage faible). On peut aussi l'exploiter dans le domaine médical, pour l'angiographie et la scintigraphie, lorsque l'effet du produit de contraste est épuisé.

Ce travail représente une étape importante pour l'exploitation des métaheuristiques quantiques pour la résolution de l'émergence inversée. Dans le futur, on envisage l'amélioration des algorithmes quantiques proposés en introduisant un opérateur de mutation, ou une distribution de Cauchy, pour assurer une meilleure exploration de l'espace de recherche et augmenter la capacité de recherche globale.
Une autre perspective interessante à ce travail serait l'usage des GIM (Generalized Island Models), en combinant plusieurs métaheuristiques ensemble, comme le PSO, GA, QPSO..etc, et en les exécutant en parallèle, tout en échangeant leurs meilleures valeurs, le processus d'optimisation serait accéléré, et les résultats améliorés.
Il serait aussi intéressant d'explorer l'impact du changement de voisinage de l'AC (par exemple 5*5) sur les performances de ces algorithmes d'optimisation, et de chercher des solutions afin de les appliquer directement sur des images plus complexes comme les images couleur, cela reste un vrai défi car l'espace de recherche va croitre de façon drastique.

# Travaux de l'auteur

## Année 2009

1) « Un Système Multi-Agent Réactif pour la Segmentation d'Images », Safia Djemame, Mohamed Batouche, $2^{nd}$ International Conference on Applied Informatics ICAI 2009, 15-17 novembre 2009, Bordj Bou Arreridj, Algérie.

2)« Résolution Collective de Problème par Emergence. Application à une Segmentation d ?Images », Safia Djemame, Mohamed Batouche, First International Conference on Image and Signal Processing and Their Applications ISPA 2009, 19-21 octobre 2009, Mostaghanem, Algérie.

3)« Une Approche Bio-mimétique pour la Segmentation d'Images. Inspiration des Araignées Sociales », Safia Djemame, Mohamed Batouche. Conférence internationale sur l'informatique et ses applications, CIIA 2009, 03-04 mai 2009, Saida, Algérie.
Indexé DBLP, Lien : http ://www.informatik.unitrier.de/ley/db/conf/ciia/ciia2009.html.

4)"A Multi-Agent System for Image Segmentation. A Bio-Inspired Approach", Safia Djemame, Mabrouk Nekkache, Mohamed Batouche. IEEE-BCS Third International Symposium on Innovation in Information Communication Technology, ISIICT 2009, 15-17 December, 2009, Philadelphia University, Amman, Jordan. Publié dans le journal électronique eWic, éditeur BCS British Computer Society.
lien : http ://www.bcs.org/server.php ?show=conWebDoc.33762.

## Année 2010

1)"Cellular Automata as Complex System. Applications in Image Processing", Safia Djemame, Mohamed Batouche. International Symposium on Modelling and Implementation of Complex Systems, MISC 2010, 30-31 mai 2010, Constantine, Algérie.

2)"3D Pattern Recognition Using Ant Colony", Ait Kaci Azzou Samira, Trifi Mohamed Amine, Djemame Safia, Boukerram Abdellah, Premier Congrès International sur les modèles, Optimisation et Sécurité des Systèmes, ICMOSS 2010, 29-31 Mai 2010, Tiaret, Algérie.

3)"Développement d'une méthode d'identification d'Objets 3D en utilisant les Algorithmes Génétiques". Samira Ait Kaci Azzou, Mohamed Amine Trifi, Safia Djemame, MSISI 2010, 18-19 Octobre 2010,Skikda, Algérie.

4)"Image Segmentation Using an Emergent Complex System : Cellular Automata", Djemame Safia, Djidel Oussama, Batouche Mohamed Chawki, International Conference on Electrical Engineering, Electronics and Automatics, ICEEA 2010 ,2-3 novembre 2010, Bejaia, Algérie.

5)"Un automate cellulaire émergent pour la segmentation d'images", Djemame Safia, Djidel Oussama, Batouche Mohamed, $7^{eme}$ Séminaire national en Informatique, SNIB 2010, 2-4 novembre 2010, Biskra, Algérie.

6)"Détection de Contours sur Images à Niveaux de Gris Utilisant un Automate Cellulaire", Djemame Safia, Djidel Oussama, Batouche Mohammed, $2^{nd}$ International Conference on Image and Signal Processing and their Applications, ISPA 2010, 6-8 novembre 2010, Biskra, Algérie.

## Année 2011

1)" Un automate cellulaire continu pour l'extraction de contours sur images à niveaux de gris", Djemame Safia, Djidel Oussama, Batouche Mohamed, $1^{st}$ International Conference on Information Systems and Technologies, ICIST 2011, 24- 26 avril 2011, Tebessa, Algeria.

2)"Image Segmentation Using Continuous Cellular Automata ?, Djemame Safia, Djidel oussama, Batouche Mohamed Chawki, $10^{th}$ IEEE International Symposium on Programming and Systems, ISPS 2011, 25-27 avril 2011, Alger, Algeria.

3)"Image Segmentation Using an Emergent Complex System : Cellular Automata", Djemame Safia, Batouche Mohamed, 7th IEEE International Workshop on Systems, Signal Processing and their Applications, WOSSPA 2011, 9-11 mai 2011, Tipaza, Algeria.

## Année 2012

1)"A Hybrid Approach for Edge Detection Using Cellular Automata and Particle Swarm Optimization" , Djemame Safia, Mohamed Batouche, $2^{nd}$ World Conference on Innovation and

Computer Sciences, INSODE 2012, 10-14 May 2012, Izmir, Turkey, published in Procedia-Computer Science Journal (ISSN : 1877-0509) by Elsevier.

2)"Une Nouvelle Approche de Détection de Contours Basée sur les Automates Cellulaires et l'OEP", Djemame Safia, Mohamed Batouche, Biomedical Engineering international conference, BIOMEIC 2012, 10-11 octobre 2012, Tlemcen, Algérie.

3)"Une Approche de Détection de Contours Basée sur la Métaheuristique OEP", Djemame Safia, Mohamed Batouche, $11^{eme}$ Colloque Africain sur la recherche en Informatique CARI 2012, 13-16 octobre 2012, Alger, Algérie.

4)"Combining Cellular Automata and Particle Swarm Optimization for Edge Detection", Safia Djemame, Mohamed Batouche, International Journal of Computer Applications IJCA (ISSN 0975-8887), volume 57, N° 14, pp 16-22, novembre 2012.
Available online at : www.ijcaonline.org/archives/volume 57/number14/9182-3602

## Année 2014

1)"Optimisation de règles de transition d'un automate cellulaire pour la détection de contours", Zid Wissem, Brahma Bochra, Djemame Safia, la journée de l'étudiant, JEESI 2014, le 19 mai 2014 à l'ESI, Alger, Algérie.

2)"Exploration de l'émergence inversée pour la détection de contours en traitement d'images", Safia Djemame, Mohamed Batouche. Colloque International sur l'Optimisation et les systèmes d'Information COSI 2014, 08-10 juin 2014, Bejaia, Algérie.

3)"Segmentation d'images par émergence inversée", Djemame Safia, Mohamed Batouche, $2^{eme}$ Journée Informatique de l'université de BBA, JIBBA 2014, le 03 décembre 2014, Bordj Bou Arreridj, Algérie.

## Année 2016

1)"Hybridation d'un Algorithme Génétique Quantique et d'un Système Complexe pour le Traitement d'Images", Safia Djemame, Mohamed Batouche, $13^{eme}$ Colloque sur l'Optimisation et Systèmes d'Information, COSI 2016, 30 Mai - 01 Juin 2016, Sétif, Algérie.

2)"Quantum Genetic Computing and Cellular Automata for Solving Edge Detection", Safia Djemame, Mohamed Batouche, The first International Conference on Computer Science's

Complex Systems and their Applications, ICCSA 2016, 12-14 April 2016, Oum El Bouaghi, Algeria.

## Année 2018

1)"A Hybrid Metaheuristic Algorithm Based on Quantum Genetic Computing for Image Segmentation", Safia Djemame, Mohamed Batouche, Book Chapter, Paru dans le livre : "Hybrid Metaheuristics for Image Analysis". Springer International Publishing, juillet 2018. https ://doi.org/10.1007/978-3-319-77625-5-2

2)"Solving Reverse Emergence with Quantum PSO - Application to Image Processing", Safia Djemame, Mohamed Batouche, Hamouche Oulhadj, Patrick Siarry. Soft Computing Journal, Springer, juin 2018. https ://doi.org/10.1007/s00500-018-3331-6

# Bibliographie


[A.10] Layeb A. A quantum inspired particle swarm algorithm for solving the maximum satisfiability problem. *IJCOPI*, 1(1) :13–23, 2010.

[ABC98] G. Adorni, F. Bergenti, and S. Cagnoni. A cellular-programming approach to pattern classification. In *European Conference on Genetic Programming*, pages 142–150. Springer, 1998.

[AEH09] Mohammed R AlRashidi and Mohamed E El-Hawary. A survey of particle swarm optimization applications in electric power systems. *IEEE Transactions on Evolutionary Computation*, 13(4) :913–918, 2009.

[AP57] W Ross Ashby and JR Pierce. An introduction to cybernetics. *Physics Today*, 10(7) :34–36, 1957.

[APV05] AV Adamopoulos, NG Pavlidis, and MN Vrahatis. Genetic algorithm evolution of cellular automata rules for complex binary sequence prediction. *Lecture Series on Computer and Computational Sciences*, 1 :1–6, 2005.

[ARL03] Anas N. Al-Rabadi and George G. Lendaris. Artificial neural network implementation using many-valued quantum computing. In *Proceedings of the International Joint Conference on Neural Networks*, volume 4, pages 3112–3117, 2003.

[AZ97] Syed Mustafa Ali and Robert Zimmer. The question concerning emergence. casys ?97, abstract book. In *First International Conference on Computing Anticipatory Systems, CHAOS*, volume 48, 1997.

[BB05] Ron Breukelaar and Th Back. Using a genetic algorithm to evolve behavior in multi dimensional cellular automata : emergence of behavior. In *Proceedings of the 7th annual conference on Genetic and evolutionary computation*, pages 107–114. ACM, 2005.

[BCG82] E. R. Berlekamp, J. H. Conway, and R. Guy. *Winning ways for your mathematical theory*. New York Academic Press, 1982.

[BDT99] Eric Bonabeau, Marco Dorigo, and Guy Theraulaz. *Swarm intelligence : from natural to artificial systems*. Oxford university press, 1999.

[Bel03] M Bellac. Introduction à l'information quantique. *Cours donné à l'Ecole Supérieure de Sciences Informatiques (ESSI)*, 2003.

[BMA06] Mohamed Batouche, Souham Meshoul, and Ali Abbassene. On solving edge detection by emergence. In *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*, pages 800–808. Springer, 2006.

[BMAH09] Mohamed Batouche, Souham Meshoul, and A Al Hussaini. Image processing using quantum computing and reverse emergence. *International Journal of Nano and Biomaterials*, 2(1-5) :136–142, 2009.

[Bou03] L. Bougrain. *Réseaux de neurones artificiels.* Rapport-03, Projet CORTEX, LORIA/INRIA-Lorraine, 2003.

[Bru00] S.A. Brueckner. *Return from the Ant : Synthetic Ecosystems for Manufacturing.* phD thesis, Humboldt University, Berlin, Department of Computer Science, 2000.

[BS02] Hans-Georg Beyer and Hans-Paul Schwefel. Evolution strategies–a comprehensive introduction. *Natural computing*, 1(1) :3–52, 2002.

[BT94] E Bonabeau and G Theraulaz. *Intelligence collective.* Editions Hermès, Paris, 1994.

[BT97] E Bonabeau and G Theraulaz. *Auto-organisation et comportements collectifs : la modélisation des sociétés d'insectes.* Editions Hermès, Paris, 1997.

[Car02] T. Cardin. *Image Processing with Artificial Life.* 2002.

[Cav11] L. Cavalier. Inverse problems in statistics. *Inverse Problems and High-Dimentional Estimation*, pages 3–96, 2011.

[CD06] Arturo Chavoya and Yves Duthen. Using a genetic algorithm to evolve cellular automata for 2d/3d computational development. In *Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 231–232. ACM, 2006.

[Čer85] Vladimír Černỳ. Thermodynamical approach to the traveling salesman problem : An efficient simulation algorithm. *Journal of optimization theory and applications*, 45(1) :41–51, 1985.

[CH13] E. Casper and C. Hung. Quantum modeled clustering algorithms for image segmentation. *Progress in intelligent computing and applications*, 2(1) :1–21, 2013.

[CK02] M. Clerc and J. Kennedy. The particle swarm : explosion, stability and convergence in a multi-dimensional complex space. *IEEE Transaction on Evolutionary Computation*, 6 :58–73, 2002.

[CL99] Sheng Chen and Bing Lam Luk. Adaptive simulated annealing for optimization in signal processing applications. *Signal Processing*, 79(1) :117–128, 1999.

[CM02] A. Cornuéjols and L. Miclet. *L'apprentissage artificiel. Concepts et algorithmes.* Eyrolles, 2002.

[CNM08] Leandro Coelho, Nadia Nedjah, and Luiza Mourelle. Gaussian quantum-behaved particle swarm optimization applied to fuzzy pid controller design. In *Quantum inspired intelligent systems*, pages 1–15. Springer, 2008.

[Cob11] J. Cobb. *A Definition of Learning.* http ://www.missiontolearn.com/2009/05/definition-of-learning/, 2011.

[DAGP90] J-L Deneubourg, Serge Aron, Simon Goss, and Jacques M Pasteels. The self-organizing exploratory pattern of the argentine ant. *Journal of insect behavior*, 3(2) :159–168, 1990.

[Dar59] Charles Darwin. *The origin of species by means of natural selection : or, the preservation of favoured races in the struggle for life.* 1859.

[DB12] S. Djemame and M. Batouche. Combining cellular automata and particle swarm optimization for edge detection. *International Journal of Computer Applications*, 57(14) :16–22, 2012.

[DB16] S. Djemame and M. Batouche. Hybridation d'un algorithme génétique quantique et d'un système complexe pour le traitement d'images. In *13 eme Colloque sur l'Optimisation et Systèmes d'Information 30 Mai - 01 Juin, 2016 Sétif, ALGERIE*, 2016.

[DB18] S. Djemame and M. Batouche. A hybrid metaheuristic algorithm based on quantum genetic computing for image segmentation. In *Hybrid Metaheuristics for Image Analysis*. Springer, 2018.

[DBOS18] S. Djemame, M. Batouche, H. Oulhadj, and P. Siarry. Solving reverse emergence with quantum pso - application to image processing. *Soft Computing, Springer, https ://link.springer.com/article/10.1007%2Fs00500-018-3331-6*, 2018.

[dCPVB05] A.V.A da Cruz, M.A.C Pacheco, M. Vellasco, and C.R.H Barbosa. Cultural operators for a quantum-inspired evolutionary algorithm applied to numerical optimization problems. In *International Work-Conference on the Interplay Between Natural and Artificial Computation*, pages 1–10. Springer, 2005.

[DCT02] Leandro Nunes De Castro and Jonathan Timmis. *Artificial immune systems : a new computational intelligence approach.* Springer Science & Business Media, 2002.

[dCVP07] A.V.A da Cruz, M.M.B.R Vellasco, and M.A.C Pacheco. Quantum-inspired evolutionary algorithm for numerical optimization. In *Hybrid evolutionary algorithms*, pages 19–37. Springer, 2007.

[DD00] Rebecca Dodder and Robert Dare. Complex adaptive systems and complexity theory : inter-related knowledge domains. In *ESD. 83 : Research Seminar in Engineering Systems*, 2000.

[DE06] David Deutsch and Artur Ekert. Quantum computation. *Physics World*, 11(3) :47–52, 2006.

[DG+99] Hugo De Garis et al. Artificial embryology and cellular differentiation. *Evolutionary design by computers*, pages 281–295, 1999.

[DGF+91] Jean-Louis Deneubourg, Simon Goss, Nigel Franks, Ana Sendova-Franks, Claire Detrain, and Laeticia Chrétien. The dynamics of collective sorting robot-like ants and ant-like robots. In *Proceedings of the first international conference on simulation of adaptive behavior on From animals to animats*, pages 356–363, 1991.

[DM12] S. Djemame and M.Batouche. Une approche de détection de contours basée sur la métaheuristique oep. In *11eme Colloque Africain sur la recherche en Informatique CARI 2012, 13-16 octobre 2012, Alger, Algérie.*, 2012.

[DMC96] Marco Dorigo, Vittorio Maniezzo, and Alberto Colorni. Ant system : optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 26(1) :29–41, 1996.

[DPST03] Johann Dréo, Alain Pétrowski, Patrick Siarry, and Eric Taillard. *Métaheuristiques pour l'optimisation difficile*. Eyrolles, 2003.

[DTB05] Amer Draa, Hichem Talbi, and Mohamed Batouche. A quantum inspired genetic algorithm for solving the n-queens problem. In *Proceedings of the 7th International Symposium on Programming and Systems*, pages 145–152, 2005.

[Edm00] B. Edmonds. Syntactic measures of complexity. In *Thèse de doctorat, University of Manchester*, 2000.

[Eng06] Andries P Engelbrecht. *Fundamentals of computational swarm intelligence*. John Wiley & Sons, 2006.

[ES07] RC. Eberhart and Y. Shi. Computational intelligence. *Computational Intelligence : Concepts to Implementations*, 2007.

[Fog97] David B. Fogel. Evolutionary computation : A new transactions. *IEEE Transactions on Evolutionary Computation*, 1(1) :1–2, 1997.

[FOW66] Lawrence J Fogel, Alvin J Owens, and Michael J Walsh. Artificial intelligence through simulated evolution. 1966.

[Fro05] Jochen Fromm. Types and forms of emergence. *arXiv preprint nlin/0506028*, 2005.

[Fro06] Jochen Fromm. On engineering and emergence. *arXiv preprint nlin/0601002*, 2006.

[GADP89] Simon Goss, Serge Aron, Jean-Louis Deneubourg, and Jacques Marie Pasteels. Self-organized shortcuts in the argentine ant. *Naturwissenschaften*, 76(12) :579–581, 1989.

[GEO03] Jean-Pierre GEORGE. L'emergence, rapport technique, irit. 2003.

[GL97] Fred Glover and Manuel Laguna. Tabu search, 1997. *Kluwer Academic Publishers*, 1997.

[Gle04] Marie-Pierre Gleizes. *Vers la résolution de problèmes par émergence*. PhD thesis, Université de Toulouse, France, 2004.

[Glo86] Fred Glover. Future paths for integer programming and links to artificial intelligence. *Computers and operations research*, 13(5) :533–549, 1986.

[Gol89] DE Goldberg. Genetic algorithms in search, optimization, and machine learning, addison-wesley, reading, ma. *K. Ohno, K. Esfarjani, and Y Kawazoe : Computational Materi*, 1989.

[Gri10] A. M. Grisogono. *Complex Adaptive Systems Concepts for Complex Engineering.* JSA Action Group 14 Complex Adaptive Systems for Defence. The Technical Cooperation Program, 2010.

[Gro96] L.K Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 212–219. ACM, 1996.

[GSD+03] N. Ganguly, B.K. Sikdar, A. Deutsch, G. Canright, and P Pal. Chaudhuri. A survey on cellular automata, centre for high performance computing, dresden university of technology. 2003.

[Had32] J. Hadamard. *Le probleme de Cauchy et les équations aux dérivées partielles linéaires hyperboliques*. Hermann Paris, 1932.

[Han03] Kuk-Hyun Han. Quantum-inspired evolutionary algorithm. In *thèse de doctorat, Korea Adv. Inst. Sci. Technol. (KAIST)*, 2003.

[Has03] Salima Hassas. Systèmes complexes à base de multi-agents situés. *University Claude Bernard Lyon*, 2003.

[Hey99] Tony Hey. Quantum computing : an introduction. *Computing & Control Engineering Journal*, 10(3) :105–112, 1999.

[HK00] Kuk-Hyun Han and Jong-Hwan Kim. Genetic quantum algorithm and its application to combinatorial optimization problem. In *Proceedings of the 2000 IEEE Congress on Evolutionary Computation*, volume 2, pages 1354–1360, 2000.

[HK02] Kuk-Hyun Han and Jong-Hwan Kim. Quantum-inspired evolutionary algorithm for a class of combinatorial optimization. *IEEE transactions on evolutionary computation*, 6(6) :580–593, 2002.

[HK04] Kuk-Hyun Han and Jong-Hwan Kim. Quantum-inspired evolutionary algorithms with a new termination criterion, h &epsi ; gate, and two-phase scheme. *IEEE transactions on evolutionary computation*, 8(2) :156–169, 2004.

[Hol75] John H Holland. Adaptation in natural and artificial systems. an introductory analysis with application to biology, control, and artificial intelligence. *Ann Arbor, MI : University of Michigan Press*, 1975.

[HW03] S. Henon and G. Weisbuch. Systèmes complexes en shs. *Rapport technique*, CNRS(PARIS), 2003.

[II08] Y. Ichise and Y. Ishida. Reverse engineering of spatial patterns in cellular automata. In *13th International Symposium on Artificial Life and Robotics*, pages 172–175, 2008.

[Ila04] Andrew Ilachinski. *Artificial war : Multiagent-based simulation of combat*. World Scientific, 2004.

[Kar10] Dervis Karaboga. Artificial bee colony algorithm. *scholarpedia*, 5(3) :6915, 2010.

[KE95] J Kennedy and R.C Eberhart. Particle swarm optimization. In *IEEE int. conf. on Neural networks, Piscataway, Japan*, pages 1942–1948, 1995.

[Kel76] J.B Keller. Inverse problems. *The American Mathematical Monthly*, 83 :107–118, 1976.

[KGV+83] Scott Kirkpatrick, C Daniel Gelatt, Mario P Vecchi, et al. Optimization by simmulated annealing. *science*, 220(4598) :671–680, 1983.

[Koz94] John R Koza. Genetic programming as a means for programming computers by natural selection. *Statistics and computing*, 4(2) :87–112, 1994.

[KS00] M. Kudo and J. Sklansky. Comparison of algorithms that select features for pattern classifiers. *Pattern Recognition journal*, 33 :25–41, 2000.

[KS10] Tapas Kumar and G Sahoo. A novel method of edge detection using cellular automata. *International Journal of Computer Applications*, 9(4) :38–44, 2010.

[KS11] O. Kazar and S. Slatnia. Evolutionary cellular automata for image segmentation and noise filtering using genetic algorithms. *Journal of applied computer science and mathematics*, 10(5), 2011.

[L+92] Christopher G Langton et al. Life at the edge of chaos. *Artificial life II*, 10 :41–91, 1992.

[Lan90] Chris G Langton. Computation at the edge of chaos : phase transitions and emergent computation. *Physica D : Nonlinear Phenomena*, 42(1-3) :12–37, 1990.

[LL01] Pedro Larrañaga and Jose A Lozano. *Estimation of distribution algorithms : A new tool for evolutionary computation*, volume 2. Springer Science & Business Media, 2001.

[LMB06] Abdesslem Layeb, Soham Meshoul, and Mohamed Batouche. Multiple sequence alignment by quantum genetic algorithm. In *Proceedings 20th IEEE International Parallel & Distributed Processing Symposium*, pages 311–318. IEEE, 2006.

[LMB08] Abdesslem Layeb, Souham Meshoul, and Mohamed Batouche. Quantum genetic algorithm for multiple rna structural alignment. In *Second Asia International Conference on Modelling and Simulation*, pages 873–878. IEEE, 2008.

[LS07] Abdesslem Layeb and Djamel-Eddine Saidouni. Quantum genetic algorithm for binary decision diagram ordering problem. *International Journal of Computer Science and Network Security*, 7(9) :130–135, 2007.

[Lt11] Learn-theo. *Learning Theory*. http ://www.npac.syr.edu/projects/cpsedu/ summer98summary/examples/hpf/hpf.html., 2011.

[Luc02] Chris Lucas. A logic of complex values. In *Proceedings of the first international conference on Neutrosophy, neutrosophic logic, neutrosophic set, neutrosophic probability and statistics*, pages 121–138, 2002.

[Man01] Steven M Manson. Simplifying complexity : a review of complexity theory. *Geoforum*, 32(3) :405–414, 2001.

[MAR97] Pierre MARCENAC. Modèlisation de systèmes complexes par agent, technique et science informatique, vol. 16, n 8. *Hermès*, pages 1013–1038, 1997.

[MCD+96] Melanie Mitchell, James P Crutchfield, Rajarshi Das, et al. Evolving cellular automata with genetic algorithms : A review of recent work. In *Proceedings of the First International Conference on Evolutionary Computation and Its Applications (EvCA96)*. Moscow, 1996.

[Moi99] J. Le Moigne. *La modèlisation des systèmes complexes.* Dunod, Paris, 1999.

[Moo64] Moore. *Sequential machines.* Wesley Longman éd. Essex : Series in computer science and information processing., 1964.

[Mor08] Edgar Morin. *On complexity (Advances in systems theory, complexity, and the human sciences).* New York : Hampton Press, 2008.

[MP96] Heinz Mühlenbein and Gerhard Paass. From recombination of genes to the estimation of distributions i. binary parameters. In *International conference on parallel problem solving from nature*, pages 178–187. Springer, 1996.

[MRR+53] Nicholas Metropolis, Arianna W Rosenbluth, Marshall N Rosenbluth, Augusta H Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6) :1087–1092, 1953.

[MV80] BP McLaughlin and F Varela. *Autopoiesis and Cognition : The Realization of the Living ?. London. D.* Reidel Publishing Company Holland, 1980.

[Nar99] A. Narayanan. Quantum computing for enginners. In *Congress on Evolutionary Computation*, pages 2231–2238, 1999.

[NC10] Michael A Nielsen and Isaac L Chuang. *Quantum computation and quantum information.* Cambridge university press, 2010.

[NM96] A. Narayanan and M. Moore. Quantum-inspired genetic algorithms. In *Proceedings of the IEE International Conference on Evolutionary Computation,ICEC 96*, pages 41–46, 1996.

[NRS15] D.L. Naidu, C.S. Rao, and S. Satapathy. *A hybrid approach for image edge detection using neural network and particle swarm optimization*, volume 337 of *Advances in Intelligent Systems and Computing*, chapter A hybrid approach for image edge detection using neural network and particle swarm optimization, pages 1–9. Springer International Publishing, 2015.

[OES05] M Omran, Andries Petrus Engelbrecht, and A Salman. Particle swarm optimization method for image clustering. *International Journal of Pattern Recognition and Artificial Intelligence*, 19(03) :297–321, 2005.

[OH92] Michael J. O'Brien and Thomas D. Holland. The role of adaptation in archaeological explanation. *American Antiquity*, 57(1) :36–59, 1992.

[Par97] H Van Dyke Parunak. " go to the ant" : Engineering principles from natural multi-agent systems. *Annals of Operations Research*, 75 :69–101, 1997.

[Par02] Lael Parrott. Complexity and the limits of ecological engineering. *Transactions-American Society of Agricultural Engineers*, 45(5) :1697–1702, 2002.

[Pot04] A. Potgieter. *The Engineering of Emergence of Complex Adaptive Systems*. Thèse de doctorat, Université de Pretoria, 2004.

[PP02] Adriana Popovici and Dan Popovici. Cellular automata in image processing. In *Fifteenth International Symposium on Mathematical Theory of Networks and Systems*, volume 1, 2002.

[Pre98] John Preskill. Lecture notes for physics 229 : Quantum information and computation. *California Institute of Technology*, 12 :14, 1998.

[PS96] Ilya Prigogine and Isabelle Stengers. *La fin des certitudes : temps, chaos et les lois de la nature*, volume 77. Odile Jacob, 1996.

[PSL06] Kenneth Price, Rainer M Storn, and Jouni A Lampinen. *Differential evolution : a practical approach to global optimization*. Springer Science & Business Media, 2006.

[PWGB10] Suraj Pandey, Linlin Wu, Siddeswara Mayura Guru, and Rajkumar Buyya. A particle swarm optimization-based heuristic for scheduling workflow applications in cloud computing environments. In *2010 24th IEEE international conference on advanced information networking and applications*, pages 400–407. IEEE, 2010.

[RCL01] K.A. Richardson, P. Cilliers, and M. Lissack. *Complexity Science : A Gray Science for the Stuff in Between*. Lawrence Erlbaum Associates, 2001.

[Rec73] Ingo Rechenberg. Evolutionsstrategie–optimierung technisher systeme nach prinzipien der biologischen evolution. 1973.

[Rey87] Craig W Reynolds. Flocks, herds and schools : A distributed behavioral model. *ACM SIGGRAPH computer graphics*, 21(4) :25–34, 1987.

[Ros06] Paul L Rosin. Training cellular automata for image processing. *IEEE transactions on image processing*, 15(7) :2076–2087, 2006.

[Ros10] Paul L Rosin. Image processing using 3-state cellular automata. *Computer vision and image understanding*, 114(7) :790–802, 2010.

[RP00] Eleanor Rieffel and Wolfgang Polak. An introduction to quantum computing for non-physicists. *ACM Computing surveys*, 32(3) :300–335, 2000.

[RRD90] Pierre Roussel-Ragot and Gerard Dreyfus. A problem independent parallel implementation of simulated annealing : Models and experiments. *IEEE transactions on computer-aided design of integrated circuits and systems*, 9(8) :827–835, 1990.

[San07] Ronan Sandford. Modélisation de réseaux de neurones quantiques. un modèle pour le fonctionnement du cerveau ? *Laboratoire de sciences cognitives, Université Victor Segalen Bordeaux 2*, 2007.

[Sap03] E. Sapin. *Recherche par algorithmes évolutionnaires d'automates cellulaires universels*. PhD thesis, Dijon, 2003.

[SBDH97] Patrick Siarry, Gérard Berthiau, François Durdin, and Jacques Haussy. Enhanced simulated annealing for globally minimizing functions of many-continuous variables. *ACM Transactions on Mathematical Software (TOMS)*, 23(2) :209–228, 1997.

[SBM07] Sihem Slatnia, Mohamed Batouche, and Kamal E Melkemi. Evolutionary cellular automata based-approach for edge detection. In *International Workshop on Fuzzy Logic and Applications*, pages 404–411. Springer, 2007.

[SE99] Yuhui Shi and Russell C Eberhart. Empirical study of particle swarm optimization. In *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on*, volume 3. IEEE, 1999.

[SFP+11] J. Sun, W. Fang, V. Palade, X. Wua, and W. Xu. Quantum-behaved particle swarm optimization with Gaussian distributed local attractor point. *Applied Mathematics and Computation*, 218 :3763–3775, 2011.

[SFW+12] J. Sun, W. Fang, X. Wu, V. Palade, and W. Xu. Quantum-behaved particle swarm optimization : analysis of individual particle behavior and parameter selection. *Evolutionary Computation*, 20(3) :349–393, 2012.

[SFX04] J. Sun, B. Feng, and W. Xu. Particle swarm optimization with particles having quantum behavior. In *Proceedings of IEEE Congress on Evolutionary Computation, 19–23 june, Portland, USA*, pages 325–331, 2004.

[SH09] P Jebaraj Selvapeter and Wim Hordijk. Cellular automata for image noise filtering. In *World Congress on Nature and Biologically Inspired Computing, NaBIC 2009.*, pages 193–197. IEEE, 2009.

[Sho94] Peter W Shor. Algorithms for quantum computation : Discrete logarithms and factoring. In *Foundations of Computer Science, 1994 Proceedings., 35th Annual Symposium on*, pages 124–134. IEEE, 1994.

[Sip97] M. Sipper. The evolution of parallel cellular machines toward evolware. *Biosystems*, 42 :29–43, 1997.

[SP97] Rainer Storn and Kenneth Price. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization*, 11(4) :341–359, 1997.

[SW06] Barry Smit and Johanna Wandel. Adaptation, adaptive capacity and vulnerability. *Global environmental change*, 16(3) :282–292, 2006.

[SWB05] J. Sun, X. Wenbo, and F. Bin. Adaptive parameter control for Quantum-behaved particle swarm optimization on individual level. In *Proceedings of IEEE conference on Systems, Man and Cybernetics, 10–12 october, Hawaii, USA*, pages 3049–3054, 2005.

[SXF04] J. Sun, W. Xu, and B. Feng. A global search strategy of Quantum-behaved particle swarm optimization. In *Proceedings of IEEE Conference on Cybernetics and Intelligent Systems, 1–3 december, Singapore*, pages 111–116, 2004.

[SXL05] J. Sun, W. Xu, and J. Liu. *Parameter selection of Quantum-behaved particle swarm optimization, Advances in Natural Computation*, volume 3612, pages 543–552. Springer Berlin Heidelberg, 2005.

[Tal09] Hichem Talbi. Algorithmes évolutionnaires quantiques pour le recalage et la segmentation multiobjectif d'images. *Thèse de doctorat, Université Mentouri, Constantine*, 2009.

[TBD04] Hichem Talbi, Mohamed Batouche, and Amer Draa. A quantum-inspired genetic algorithm for multi-source affine image registration. In *International Conference Image Analysis and Recognition*, pages 147–154. Springer, 2004.

[TBD07] Hichem Talbi, Mohamed Batouche, and Amer Draa. A quantum-inspired evolutionary algorithm for multiobjective image segmentation. *International Journal of Mathematical, Physical and Engineering Sciences*, 1(2) :109–114, 2007.

[TDB04] Hichem Talbi, Amer Draa, and Mohamed Batouche. A new quantum-inspired genetic algorithm for solving the travelling salesman problem. In *IEEE International Conference on Industrial Technology*, volume 3, pages 1192–1197. IEEE, 2004.

[TDB06] Hichem Talbi, Amer Draa, and Mohamed Batouche. A novel quantum-inspired evolutionary algorithm for multi-sensor image registration. *International Arabic Journal on Information Technology*, 3(1) :9–15, 2006.

[Tur48] A. Turing. *Intelligent Machinery*. Report for National Physical Laboratory : National Physical Laboratory, 1948.

[Tur50] Alan M Turing. Computing machinery and intelligence. *Mind*, 59(236) :433–460, 1950.

[UHS05] Hiroshi Umeo, Masaya Hisaoka, and Takashi Sogabe. A survey on optimum-time firing squad synchronization algorithms for one-dimensional cellular automata. *International Journal of Unconventional Computing*, 1(4) :403, 2005.

[vdBE00] Frans van den Bergh and Andries P Engelbrecht. Cooperative learning in neural networks using particle swarm optimizers. *South African Computer Journal*, 1(26) :84–90, 2000.

[VM98] Dan Ventura and Tony Martinez. An artificial neuron with quantum mechanical properties. In *Artificial Neural Nets and Genetic Algorithms*, pages 482–485. Springer, 1998.

[VS15] S. H. Krishna Veni and L. Padma Suresh. *An analysis of various edge detection techniques on illuminant variant images*, volume 325 of *Advances in Intelligent Systems and Computing*, chapter An analysis of various edge detection techniques on illuminant variant images, pages 521–532. Springer India, 2015.

[WBSS04] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment : from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4) :600–612, 2004.

[WC98] C.P. Williams and S.H. Clearwater. *Explorations in Quantum Computing.* Springer Verlag, Berlin, Germany, 1998.

[Wei99] Gerhard Weiss. *Multiagent systems : a modern approach to distributed artificial intelligence.* MIT press, 1999.

[WLZF13] Huaixiao Wang, Jianyong Liu, Jun Zhi, and Chengqun Fu. The improvement of quantum genetic algorithm and its application on function optimization. *Mathematical Problems in Engineering*, 2013, 2013.

[Wol83] S. Wolfram. Statistical mechanics of cellular automata. *Review of Modern Physics*, 55 :601–644, 1983.

[Woo11] D. Wooten. *Overview for Discussion on Complex Systems at UAA and Around the Country.* Technical report, University of Alaska Anchorage, 2011.

[WT07] Sartra Wongthanavasu and Vorachai Tangvoraphonkchai. Cellular automata-based algorithm and its application in medical image processing. In *ICIP (3)*, pages 41–44, 2007.

[WT08] Sartra Wongthanavasu and S Tanvoraphonkchai. Cellular automata-based identification of the pectoral muscle in mammograms. In *The proceedings of the 3rd international symposium on biomedical engineering*, pages 294–298, 2008.

[Yun93] J.B. Yunés. *Synchronisation et automates cellulaires : la ligne de fusiliers.* PhD thesis, Université Paris-Diderot-Paris VII, 1993.

[ZJL03] Gexiang Zhang, Weidong Jin, and Na Li. An improved quantum genetic algorithm and its application. In *International Workshop on Rough Sets, Fuzzy Sets, Data Mining, and Granular-Soft Computing*, pages 449–452. Springer, 2003.

[ZYY07] Ke Zhang, Jin-sha Yuan, and Xue-ming Yang. Edge detection of images based on cloud model cellular automata. *WSEAS transactions on biology and biomedicine*, 4 :67–72, 2007.

[ZZJS11] J. Zhang, J. Zhou, H. Jin, and J. Song. An improved quantum genetic algorithm for image segmentation. *Journal of computational information Systems*, 11 :3979–3985, 2011.