Evolution d'Automate Cellulaire par Algorithme Génétique Quantique

Zakaria Laboudi [1] - Salim Chikhi [2]

Equipe SCAL, Laboratoire MISC
Université Mentouri de Constantine.
E - Mail : [1] laboudizak@yahoo.fr; [2] slchikhi@yahoo.com

Résumé. Evoluer des solutions plutôt que de les calculer représente certainement une approche de programmation très prometteuse. Le calcul évolutionnaire a déjà été connu dans l'informatique depuis plus de 4 décennies. Plus récemment, une autre variante d'algorithmes évolutionnaires a été inventée : les algorithmes génétiques quantiques. Dans ce papier, nous aborderons cette approche en donnant un exemple où elle sert à programmer automatiquement les règles de transitions des automates cellulaires. Nos résultats ont montré que les AGQ peuvent être un outil très puissant pour l'exploration des espaces de recherches d'AC.

Mots clés : Algorithmes Génétiques, Automates Cellulaires, Algorithmes Génétiques Quantiques, Calcul Quantique

1 Introduction

Les algorithmes génétiques (AG) sont un exemple représentatif d'un ensemble de méthodes connues sous le nom d'algorithmes évolutionnaires. Cette approche a été inventée dans les années 70 par John Holland, et connait depuis quelques décennies un intérêt croissant. Un AG est un algorithme itératif basé sur la notion de génération, mais ils sont aussi intrinsèquement parallèles dans la mesure où ils peuvent simuler l'évolution d'un ensemble de solutions.

Le calcul quantique est un domaine émergent des sciences de l'information en s'inspirant des principes de la mécanique quantique. Le premier algorithme quantique a été proposé par Shor en 1994 [3], pour la factorisation des nombres. Grover [7] a également proposé un algorithme quantique pour la recherche dans les bases de données non triées, la complexité de son algorithme a été réduite pour devenir de l'ordre de N. Plus récemment, le calcul quantique a suscité beaucoup d'attention et a connu un intérêt croissant par des chercheurs en informatique.

Les algorithmes génétiques quantiques (AGQ) sont la combinaison des AG et le calcul quantique. Il y a eu de grands efforts pour utiliser les AGQ pour explorer des espaces de recherche, nous citons par exemple [8] qui a proposé un AG qui inspire des principes de la mécanique quantique en le testant sur le problème type du voyageur de commerce, [12] où les AGQ ont été testés sur le problème type du sac à dos, [17] a proposé une version parallèle des AGQ. Dans [16], les AGQ ont

également été utilisées pour résoudre le problème d'ordonnancement des diagrammes binaires de décision.

Remarquant que jusqu'à présent les AGQ sont moins utilisés par rapport aux AG et pourtant les travaux réalisés dans ce champ montrent des résultats très encourageants. On se propose alors, dans ce papier, d'évoluer des AC par des AGQ pour effectuer certaines tâches de calcul et d'en extraire quelques capacités calculatoires afin de réaliser des traitements de manière efficace. En effet, la difficulté de concevoir un AC pour présenter un comportement ou accomplir une tâche particulière a souvent limité leur application, une solution possible est l'automatisation du processus de conception (programmation) pour améliorer la viabilité des AC du fait que les règles de transitions doivent être soigneusement produites à la main [2] [15]. Dans la littérature, la plupart des papiers qui portent sur l'apprentissage automatique des règles des AC mettent l'accent sur une seule technique : évolution des AC par des algorithmes évolutionnaires ce qui est connu sous le nom d'automates cellulaires évolutionnaires (ACEV). Cette technique repose sur la représentation des fonctions de transitions d'AC par des chromosomes puis les évoluer par un AG. Le début fut avec Packard, puis Mitchell et al qui ont utilisé un AG pour évoluer des AC afin de résoudre le problème de la classification de densité (voir section5) [2] [1]. Sipper aussi a évolué un autre type d'AC dits non-uniforme (les cellules peuvent avoir des règles différentes) pour effectuer la tâche de la classification de densité [6]. Plus tard, d'autres chercheurs comme Paredis [9], Pollack [10] et Werfel [11] ont considéré le problème de la classification de densité lors de l'étude des systèmes co-évolutifs où deux populations sont évoluées en tant que prédateurs (modélisés par des fonctions de transitions) et proies (modélisées par des configurations). Après la découverte de quelques règles déduites par des ACEV lors de l'étude du problème de la classification de densité (citons comme exemple les stratégies calculatoires de Mitchell et al [2] et de Pollack [13]), l'espace applicatif de cette technique s'est élargi en touchant à d'autres variantes de problèmes. Sapin a, par exemple, utilisé des ACEV pour découvrir des AC universels [13] alors que d'autres chercheurs ont utilisé des ACEV pour effectuer certaines tâches de traitement d'image comme le filtrage et la segmentation d'images [14] [15].

2 Calcul quantique

En informatique quantique, la plus petite unité de stockage de l'information est le bit quantique (qubit) [12]. Un qubit peut être dans l'état 1, dans l'état 0 ou dans une superposition des deux. L'état d'un qubit peut être représenté comme indiqué par la formule 1:

$$|\Psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \qquad (1)$$

Où | 0> et | 1> représentent les valeurs des bits classiques 0 et 1, respectivement, α et β sont des nombres complexes satisfaisant :

$$|\alpha|^2 + |\beta|^2 = 1 \quad (2)$$

$|\alpha|^2$ représente la probabilité qu'un qubit soit trouvé dans l'état 0 tandis que $|b|^2$ représente la probabilité qu'un qubit soit trouvé dans l'état 1. Un registre quantique de m bits peut donc représenter 2^m valeurs simultanément. Cependant, lorsque la 'mesure' est prise, il ne reste plus de superposition et une seule des valeurs devient alors disponible pour l'utilisation. C'est pourquoi on pense que les ordinateurs quantiques seront surtout utilisés dans des applications qui impliquent un choix parmi une multitude d'alternatives.

Un algorithme quantique est un algorithme qui ne fonctionne que sur un ordinateur quantique. Il est défini comme une succession d'opérations quantiques. En règle générale, un algorithme quantique a une complexité moindre que les algorithmes équivalents qui s'exécutent sur des ordinateurs classiques grâce au concept quantique de la superposition. Parmi les algorithmes quantiques les plus célèbres, on cite celui de Shor (1994) [3] pour la factorisation des nombres et celui de Grover (1996) [7] pour la recherche dans une base de données non triée. Ces deux algorithmes ont résolu des problèmes dont la complexité a été réduite pour devenir linéaire.

3 Automates cellulaires

Les automates cellulaires (AC) sont des systèmes dynamiques dont la caractéristique principale est de dépendre de règles pouvant être très simples mais dont le comportement global résultant peut être très complexe. Leur simplicité et leur modèle mathématique rigoureux ont fortement favorisé leur utilisation. Un AC est un espace en treillis de N cellules dont chacune est dans un des k états possibles au temps t. Chaque cellule suit la même règle pour mettre à jour son état, l'état s de la cellule au temps t+1 dépend de son état et les états d'un certain nombre de cellules voisines au temps t. L'AC commence avec une configuration initiale (CI) de cellules et à chaque pas de temps les états de toutes les cellules dans le treillis sont mis à jour d'une manière synchrone. Une règle de transition Φ peut être exprimée sous forme d'une table de règles (" look-up table ") qui énumère, pour chaque voisinage local, la mise à jour de l'état de la cellule centrale du voisinage [4]. Par exemple, si on considère un AC unidimensionnel à deux états avec un rayon de voisinage r = 1 la table des règles de mise à jour peut être donnée comme indiquée par la table 1.

N	000	001	010	011	100	101	110	111
s	0	1	1	1	0	0	0	1

Table 1. Table des voisinages

4 Algorithmes génétiques quantiques

Les AGQ sont une combinaison entre les AG et le calcul quantique. Ils sont principalement basés sur les qubits et la superposition d'états de la mécanique

quantique. Contrairement à la représentation classique des chromosomes (chaines binaires par exemple), ici les chromosomes sont représentés par des vecteurs de qubits (registres quantiques). Ainsi, un chromosome peut représenter la superposition de tous les états possibles. La structure d'un AGQ est donnée par la figure 1 [12] :

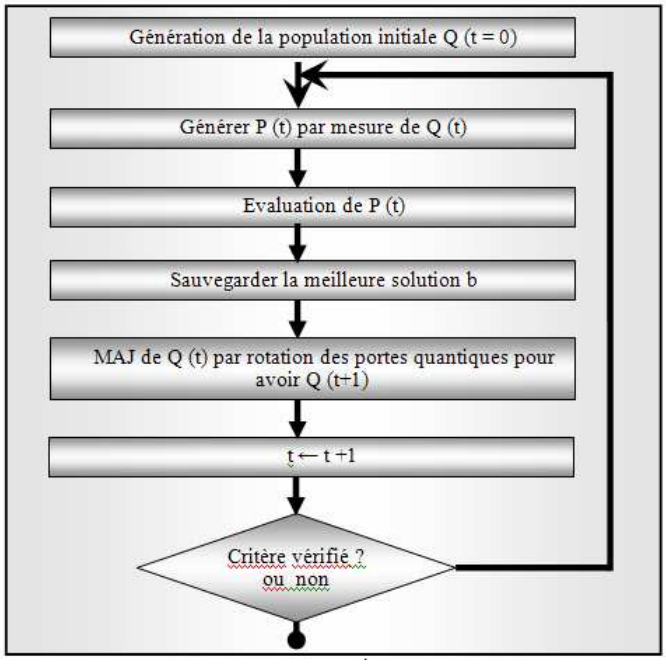

Fig.1. Structure d'un AGQ

Q (t) est la génération des chromosomes quantiques au temps t, P (t) est la génération résultante après effectuer la mesure des chromosomes de Q. Après évaluer chaque chromosome on sauvegarde la meilleure solution trouvée dans b puis on effectue une interférence par la porte quantique appropriée pour améliorer le rendement de chaque chromosome quantique.

4.1 Structure of quantum chromosomes

Comme l'élément de base ici est le qubit, un chromosome est tout simplement une chaîne de m qubits formant un registre quantique. La figure 2 montre la structure d'un chromosome quantique.

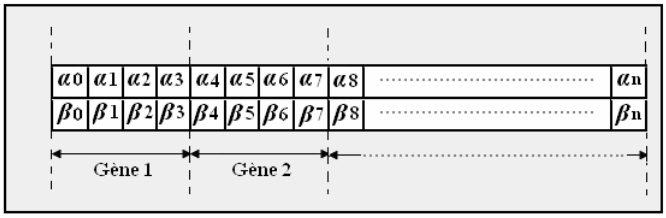

Fig.2. Structure d'un chromosome quantique.

4.2 Initialisation de la population

La génération de la population initiale a pour but de créer les chromosomes quantiques de la population initiale. La façon la plus simple est d'initialiser toutes les amplitudes des qubits par la valeur $2^{-1/2}$. Cela signifie qu'un chromosome quantique représente tous les états des superpositions avec la même probabilité.

4.3 Evaluation des individus

Le rôle de cette phase est similaire à celle d'un AG ordinaire : quantifier la qualité de chaque chromosome quantique dans la population pour effectuer une reproduction. L'évaluation se fait selon une fonction d'adaptation qui associe à chaque individu, après sa mesure, une valeur d'adaptation, et elle permet donc de noter les individus de la population.

4.4 Opérations génétiques quantiques

4.4.1 Mesure des chromosomes

Pour pouvoir exploiter efficacement les états superposés des qubits, il faut effectuer une opération de lecture pour chaque qubit. Cette opération conduit à extraire un chromosome classique à partir d'un autre quantique. Le but étant de permettre l'évaluation des individus de la population en fonction des chromosomes classiques extraits.

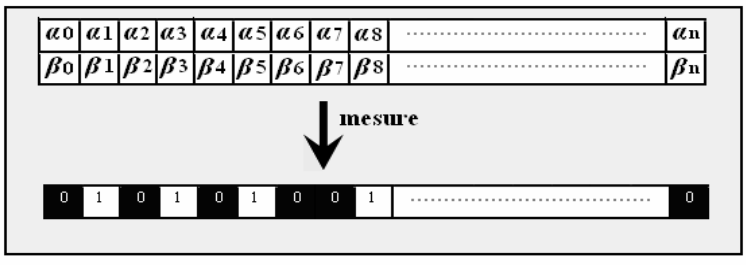

Fig.3. Mesure d'un chromosome quantique.

Une façon très simple pour implémenter cette fonction est donnée par le pseudo code qui se trouve ci dessous. Soient *q* un qubit et ' *mesure* ' sa fonction de mesure. *q* est défini par $|\Psi> = \alpha |0> + b|1>$ où $|\alpha|^2 + |b|^2 = 1$.

```
Fonction mesure ()
Début
     Pour tout qubit q d'un chromosome faire
         r := tirer aléatoirement r dans [0,1] ;
         si (r>•²)
               retourner 1 ;
         sinon
               retourner 0 ;
         fin si
     Fin pour
fin
```

4.4.2 Interférence

Cette opération permet la modification des amplitudes des individus afin d'améliorer le rendement. Elle consiste principalement en le déplacement de l'état de chaque qubit dans le sens de la valeur de la meilleure solution trouvée. Cette opération est utile pour intensifier la recherche autour de la meilleure solution. Elle peut être réalisée à l'aide d'une transformation unitaire qui permet une rotation dont l'angle est une fonction des amplitudes (ai, bi) et de la valeur du bit correspondant à la solution référence. La valeur de l'angle de rotation δθ doit être choisie de manière à éviter la convergence prématurée. Elle est souvent fixée empiriquement et sa direction est déterminée en fonction des valeurs de ai, bi et de la valeur du qubit situant à la position i de l'individu en cours de modification [16].

4.4.3 Stratégie de rotation des portes quantiques

La rotation des amplitudes des individus est faite par des portes quantiques. La porte quantique peut également être conçue en conformité avec le problème présent. La population Q (t) est mise à jour avec une rotation des portes quantiques des qubits constituant les individus. La politique de rotation adoptée est donnée par la formule :

$$\begin{bmatrix} \alpha_i^{t+1} \\ \beta_i^{t+1} \end{bmatrix} = \begin{bmatrix} \cos(\Delta\theta_i) & -\sin(\Delta\theta_i) \\ \sin(\Delta\theta_i) & \cos(\Delta\theta_i) \end{bmatrix} \begin{bmatrix} \alpha_i^{t} \\ \beta_i^{t} \end{bmatrix} \quad \textbf{(3)}$$

Où Δθi est l'angle de rotation de la porte quantique du qubit i d'un individu. Elle est souvent obtenue à partir d'une table de recherche afin d'assurer la convergence.

5 Evolution des AC par des AGQ

Nous examinons dans cette étude les AC dont : les cellules ont deux états possibles: 0 ou 1, les AC évoluent des configurations à 1-D, les règles de transitions ne prennent en compte que les trois voisins directs à gauche d'une cellule et ses trois voisins directs à droite (celle-ci sera incluse), le rayon du voisinage est donc de trois (r = 3). Le voisinage η des cellules est donc constitué de sept cellules au total. Chaque cellule peut avoir 128 configurations différentes pour η. La règle de transition doit associer à chaque configuration de η la valeur que doit prendre la cellule concernée dans la prochaine génération. Par conséquent, l'espace de tous les AC possibles devra comprendre 2 ^ 128 automates différents.

Pour démontrer la puissance des AGQ pour l'évolution des AC, nous avons mis en avant le problème type le plus étudié dans la littérature : le problème de la classification de densité ρc = ½. Pour cette tâche l'automate décide pour chaque configuration initiale de cellules noires ou blanches, s'il y a plus de cellules noires ou plus de cellules blanches. Quand il y a plus de cellules noires, l'automate doit tomber dans une configuration où toutes les cellules sont noires. On veut donc utiliser un AG pour essayer obtenir par évolution des automates capables de réaliser cette tâche. L'AG doit également s'arrêter lorsque le meilleur individu est capable de réaliser la tâche demandée ou bien l'algorithme a itéré pendant un certain nombre de générations. Ceci est une tâche non-triviale pour un AC, car l'existence d'une majorité de cellules noires est une propriété globale d'une configuration, alors que chaque cellule de l'automate peut seulement communiquer localement. Il était prouvé que cette tâche est insoluble pour un AC [5], autrement dit qu'il n'existe pas un AC qui peut effectuer cette tâche pour toutes les configurations possibles. Cette tâche peut donc être considérée comme un bon critère pour mesurer la viabilité des AGQ pour explorer des espaces de recherche d'AC en permettant de déduire des règles qui recouvrent un grand nombre de configurations.

5.1 Codage des fonctions de transitions

Nous avons codé de tels automates sur des registres quantiques de 128 qubits où chaque qubit est associé à l'une des 128 configurations possibles de η. La mesure d'un chromosome quantique conduit à créer un chromosome ordinaire dont la taille est de 128 bits. La figure 4 illustre la structure de chaque chromosome quantique appartenant à la population. Chaque règle est donc représentée sous forme d'une chaîne binaire contenant les bits de sortie de la table de voisinage. Le bit de la position 0 (c'est-à-dire, la position à gauche) dans la chaîne est l'état futur de la cellule centrale du voisinage 000**0**000, le bit à la position 1 dans la chaîne est l'état futur de la cellule centrale du voisinage 000**0**001, etc.

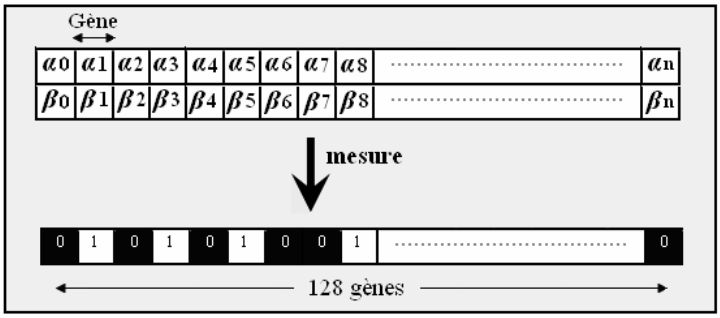

Fig.4. Structure d'un chromosome quantique [128 qubits].

5.2 Initialisation de la population

Les tests pour chaque règle de transition ont été réalisés sur des configurations de longueur N = 149 en respectant les conditions des limites périodiques. La taille de la population est de 100 individus, ce qui a été en réalité la taille de la population utilisée dans le test de Mitchell et al [2]. La population initiale est générée de manière à ce que toutes les amplitudes des chromosomes quantiques aient été initialisées par $2^{-1/2}$. Pour chaque chromosome quantique on associe un ensemble de 300 configurations initiales générées aléatoirement mais forcées d'être uniformément réparties sur $\rho \in [0.0, 1.0]$ de façon à ce que pour la moitié $\rho < \rho c$ et $\rho > \rho c$ pour l'autre moitié. A chaque génération, un nouvel ensemble de 300 configurations initiales est généré.

5.3 Evaluation des individus

L'évaluation de chaque chromosome quantique se fait, après avoir effectué une mesure, par l'évolution du chromosome extrait sur l'ensemble de ses 300 configurations initiales pendant M itérations. On attribue comme note à un individu la fraction des cellules pour lesquelles l'état final est correct.

5.4 Interférence

Une fois la population évaluée, on détermine l'AC ayant la valeur d'adaptation la plus élevée, en prenant en compte les générations précédentes. Les amplitudes des individus sont mises à jour par une rotation des portes quantiques selon la table 2.

xi bi	f (x) > f(b)	Δθi	s (ai bi)			
			ai bi > 0	ai bi < 0	ai = 0	bi = 0
0 0	0	0.004π	-	+	±	±
0 0	1	0.004π	-	+	±	±
0 1	0	0.08π	-	+	±	±
0 1	1	0.004π	-	+	±	±
1 0	0	0.08π	+	-	±	±
1 0	1	0.004π	+	-	±	±
1 1	0	0.004π	+	-	±	±
1 1	1	0.004π	+	-	±	±

Table 2. Table de recherche pour la rotation des portes quantiques

xi et bi sont les ièmes bits de x et b (la meilleure solution trouvée) respectivement. f est la fonction fitness et s(ai bi) est le signe de l'angle θi. D'après la table de recherche, on peut facilement constater que cette stratégie améliore, pour chaque individu, les amplitudes des qubits qui sont mauvaises selon un angle $\delta\theta_1 = 0.08\pi$ tandis qu'elle diminue celles qui sont bonnes selon un angle $\delta\theta_2 = 0.004\pi$. La modification des amplitudes des qubits se fait selon les signes des amplitudes, la meilleure solution trouvée et la solution extraite par l'individu la contenant. Il est normal que $\delta\theta_1$ soit >> $\delta\theta_2$ car la diminution des amplitudes sert à corriger seulement les erreurs stochastiques permettant d'une part d'éviter la dérive génétique et de garantir la diversité génétique de l'autre part.

6 Résultats expérimentaux

Nous avons exécuté notre algorithme pendant 100 générations. Chaque AC de la population a été itéré pendant M = 150 itérations. Nous avons comparé notre meilleure fitness obtenue avec celle meilleure qui existe dans la littérature. La table 3 montre le résultat obtenu.

Approche	**Fitness**
AGQ	94.18 %
AG (Meilleure fitness dans la littérature [Mitchell et al])	93% - 95 %

Table 3. Fitness Table

7 Discussion

Concernant l'aspect efficacité, la table 3 montre que la fitness obtenue par des AGQ est très similaire à celle obtenue par des AG. Le problème de la classification de

densité est généralement considéré comme un problème type très difficile à résoudre par un AC, il peut donc être un bon critère pour mesurer la puissance des pour l'exploration des espaces de recherches d'AC. Il est clair que pour cette tâche, notre algorithme était capable de déduire des règles appropriées et exactement comme un AG. Cela prouve que les AGQ et les AG peuvent être au moins équivalents et on peut donc juger que les AGQ peuvent être de puissant outil pour l'exploration des espaces de recherche d'AC.

On peut donc se demander sur l'intérêt pour les AGQ par rapport au modèle classique d'AG. De manière générale, les algorithmes quantiques peuvent minimiser la complexité des algorithmes équivalents qui s'exécutent sur des ordinateurs ordinaires. On peut faire une comparaison entre la complexité d'un AGQ et celle d'un AG pour pouvoir estimer jusqu'à quelle niveau de réduction en termes de complexité peut-on aller.

Commençons par les AGQ, comme l'interférence est réalisée en se basant seulement sur la détermination du meilleur individu dans la population, cela est très similaire à la recherche d'un optimum dans un tableau ce qui signifie que la complexité d'un AGQ augmente linéairement avec N, N étant la taille de la population.

Pour un AG ordinaire, la complexité est calculée en considérant les complexités des opérations de la sélection, le croisement et la mutation. La complexité totale de l'algorithme ici est donc au moins de l'ordre de O (N log N).

En comparant les deux complexités, il est évident que la complexité d'un AGQ est moindre que celle d'un AG, et on pense donc que ce résultat est très intéressant car la complexité ici a été réduite pour devenir linéaire. En effet, on peut imaginer que se passe-t-il lors de la manipulation des populations de chromosomes qui seront de très grandes tailles, il sera donc très utile d'utiliser des AGQ plutôt que des AG.

Une autre caractéristique très intéressante pour les AGQ est leur possibilité d'exécution parallèle. D'après la structure des AGQ indiquée dans la figure 1, il semble qu'ils peuvent supporter des traitements parallèles notamment pour la phase d'évaluation et celle d'interférence où on peut traiter plusieurs chromosomes quantiques en même temps, cela parce que la plupart des opération génétiques ont été omises (ni sélection ni croisement ni même une mutation)

On peut donc conclure que les AGQ peuvent constituer de très puissant outil pour l'exploration des espaces de recherches de manière efficace et performante.

8 Conclusion et perspectives

Dans ce travail nous avons montré la puissance des AGQ pour l'exploration des espaces de recherches. Nous avons considéré le problème de la classification de densité qui est l'un des problèmes très difficiles à résoudre par un AC. Les résultats expérimentaux ont montré que les AGQ peuvent être de très puissant outil pour l'exploration des espaces de recherches tout en préservant l'aspect efficacité / performance. Nos travaux futurs porteront sur l'exécution parallèle des AGQ afin de pouvoir extraire le potentiel du parallélisme offert par cette technique.

Références

1. N.H.Packard, Adaptation toward the edge of chaos. In J.A.S.Kelso, A.J.Mandell, and M.F. Shlesinger, editors, Dynamic Patterns in Complex Systems, pages 293–301, Singapore, World Scientific (1988).
2. Melanie Mitchell, Peter T. Hraber, and James P. Crutchfield, Revisiting the Edge of Chaos: Evolving Cellular Automata to Perform Computations, Complex Systems, 7, 89--130 (1993).
15. P.W. Shor, Algorithms for Quantum Computation:Discrete Logarithms and Factoring, Proceedings of the 35th Annual Symposium on the Foundation of Computer Sciences, pp. 20-22, 1994.
3. M. Mitchell, P. Hraber, and J. Crutchfield, "Evolving cellular automata to perform computation: Mechanisms and impedients," Phys. D, vol. 75, pp. 361–391 (1994).
4. Land, M. and Belew, R.: "NO Two-State CA for Density Classification Exists". Physical Review Letters 74, 5148, 1995.
5. M.Sipper, co-evolving Non-Uniform Cellular Automata to Perform Computations, Physica D, vol. 92, pp. 193-208 (1996).
16. L.K. Grover, A Fast Quantum Mechanical Algorithm for Database Search, Proceedings, 28th Annual ACM Symposium on the Theory of Computing, ACM Press, pp. 212-221, 1996.
17. A. Narayanan and M. Moore, "Quantum-inspired genetic algorithms," in Proceedings of IEEE International Conference on Evolutionary Computation, pp. 61-66, 1996.
12. J. Paredis, "Coevolving cellular automata: Be aware of the red queen," in Proc. 7th Int. Conf. Genetic Algorithms, T. Bäck, Ed. San Mateo, CA: Morgan Kaufmann, pp. 393–400, 1997.
13. H. Juillé and J. B. Pollack, "Coevolving the 'ideal' trainer: Application to the discovery of cellular automata rules," in Proc. 3rd Annu. Conf. Genetic Programming, J. R.Koza,W. Banzhaf, K. Chellapilla, M. Dorigo, D. B. Fogel, M. H. Garzon, D. E. Goldberg, H. Iba, and R. L. Riolo, Eds. San Mateo, CA: Morgan Kaufmann, 1998.
14. Werfel. J, Mitchell.M and Crutchfield. J. P. "Resource sharing and coevolution in evolving cellular automata". Submitted to IEEE Trans. Evol. Comp, 1999.
6. K.H. Han, Genetic Quantum Algorithm and Its Application to Combinatorial Optimization Problem, IEEE Proc. Of the 2000 Congress on Evolutionary Computation, pp. 1354-1360 (2000).
7. E.Sapin, O.Bailleux, J.J.Chabrier, and P.Collet, A New Universal Cellular Automaton Discovered by Evolutionary Algorithms, Gecco2004. Lecture Notes in Computer Science, 3102:175–187 (2004).
8. M. Batouche, S. Meshoul and A. Abbassene, On solving edge detection by emergence, the 19th International Conference on Industrial Engineering & Other Applications of Applied Intelligent Systems (IEA/AIE'06), Annecy, (France), 27-30, Lecture Notes in Artificial Intelligence (LNAI 4031), pp.800-808 (2006).
9. Paul L Rosin, Training Cellular Automata for Image Processing, IEEE Transactions on Image Processing, VOL. 15, NO. 7, Page(s): 2076 – 2087 (2006).
10. Abdesslem Layeb and Djamel-Eddine Saidouni, "Quantum Genetic Algorithm for Binary Decision Diagram Ordering Problem". IJCSNS International Journal of Computer Science and Network Security, VOL.7 No.9 (2007).
11. Shuxia Ma, Weidong Jin. A New Parallel Quantum Genetic Algorithm with Probability Gate and Its Probability A New Parallel Quantum Genetic Algorithm with Probability-Gate and Its Probability Analysis. International Conference on Intelligent Systems and Knowledge Engineering ISKE (2007).