Crossover Interaction Between the Individuals in Structured Quantum Evolutionary Algorithms

M. H. Tayarani. N
Azad University of Mashhad
tayarani@ieee.org

Abstract: *In this paper we propose a new crossover interaction operator between the individuals in structured Quantum Evolutionary Algorithms for improving the performance of Quantum Evolutionary Algorithms (QEA). The proposed structures for QEA are cellular, ring, star, grid, ladder and other structures. The proposed structures provides a better exploitation of local neighbourhoods before they move towards a global best, hence it increases population diversity. This paper compares the effects of the different structures and the proposed operator on the performance of the algorithm and diversity of the population. The proposed algorithm is tested on Knapsack problem, Trap problem and 14 numerical benchmark functions on several dimensions of 100, 250, 500 and 1000. Experimental results show that the proposed algorithm consistently exceeds the performance of QEA while keeping than QEA.*

Keywords: Quantum Evolutionary Algorithms, Cellular Genetic Algorithms, Knapsack Problem, Optimization, Structured Evolutionary algorithms.

1. Introduction

There are at least two important ingredients for a successful evolutionary design: first is maintaining a sufficiently rich genetic pool to enable exploration of new solutions, and second is developing a proper set of genetic operators to guide each generation of possible solutions toward better solutions through exploitation of existing solutions. These two ingredients have been addressed in many of the research in the evolutionary community. For example, a biologically motivated tournament-based transposition mechanism is proposed in [1] that helps preserve genetic diversity. Transposition is a context sensitive operator that promotes gene movement intra or inter chromosomes. In other work, the microbial genetic algorithm included bacterial recombination [2] as a form of recombination related to bacterial conjugation. Pseudo-bacterial genetic algorithms (PBGA) [3] incorporate a modified mutation operator called bacterial mutation, based on a natural phenomenon of microbial evolution. Bacterial evolutionary algorithms (BEA) [4] introduce a new operator, called gene transfer operator, equally inspired by microbial evolution.

The above strategies have mainly focused on genetic operators to preserve genetic diversity while a much neglected potential exists in the genetic representation. QEA is a different approach in which individuals are coded after quantum states of electrons in a probabilistic fashion. The resulting architecture is highly suitable to preserve diversity, i.e. each individual consists of m qubits that is equivalent to 2^m states. In quantum informatics, the basic carrier of information is not a bit but a quantum system with two states such as in an atom, an ion or a photon with two polarized directions, or the qubit. A qubit is in a linear superposition state and are used to specify the amplitudes of two states. In [5, 6] quantum-inspired evolutionary algorithms are investigated for a class of combinatorial optimization problems in which quantum rotation gates act as update operators. This quantum rotation gate is also used in a novel parallel quantum GA for hierarchical ring model and infinite impulse response (IIR) digital filter design [7]. Reference [8] proposes quantum evolutionary algorithm for multi-objective optimization and quantum rotation gate.

A third general approach to maintain diversity is *cellular* and *distributed* population [9, 10]. In distributed evolutionary algorithms, the population is partitioned in a set of islands which isolated EAs are executed in each island. In Cellular EAs (CEAs) the individuals are located in a grid structure and each individual interacts with its neighbours. These types of decentralized algorithms provide a better sampling of the search space and improve the performance of EAs [10]. In [11] the behaviour of Cellular Genetic Algorithms (CGA) and Dynamic CGA is analyzed. They claim that the behaviour of Dynamic CGA is more efficient and accurate than other evaluated structures.

This paper proposes a several structures for QEA with crossover interaction between the neighbours. In our proposed algorithm the quantum individuals are located in a structured environment and are interacted only with their neighbours. The interaction between the quantum individual is a crossover operator. The experimental results on the proposed algorithm show that our proposed cellular structure and interaction operator can improve the performance of the QEA.

This paper is organized as follow: Section 2 describes the representation of the QEA; in Section 3 the proposed algorithm is introduced. Section 4 evaluates the proposed algorithm on some benchmark functions and finally Section 5 concludes the proposed algorithm.

2. QEA

QEA is inspired from the principles of quantum computation, and its superposition of states is based on qubits, the smallest unit of information stored in a two-state quantum computer. A qubit could be either in state "0" or "1", or in any superposition of the two as described in (1):

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

Where α and β are complex numbers, which denote the corresponding state is appearance probability, constraint below:

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

This probabilistic representation implies that if there is a system of m qubits, the system can represent 2^m states simultaneously. At each observation, a qubits quantum state collapses to a single state as determined by its corresponding probabilities.

2.1 Representation

QEA uses a novel representation based on the above concept of qubits. Consider *j-th* individual in *t-th* generation defined as an *m*-qubit as (3):

$$q_j^t = \begin{bmatrix} \alpha_{j1}^t & \alpha_{j2}^t & \cdots & \alpha_{ji}^t & \cdots & \alpha_{jm}^t \\ \beta_{j1}^t & \beta_{j2}^t & \cdots & \beta_{ji}^t & \cdots & \beta_{jm}^t \end{bmatrix} \tag{3}$$

Where $|\alpha_{ji}^t|^2 + |\beta_{ji}^t|^2 = 1$, i=1,2,…,m , m is the number of qubits, i.e., the string length of the qubit individual, j=1,2,…,m, n is the population size and t is generation number of the evolution. Since a qubit is a probabilistic representation, any superposition of states is simultaneously represented. If there is, for instance, a three-qubits (m = 3) individual such as (4):

$$q_j^t = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{3}} & \frac{1}{2} \\ \frac{1}{\sqrt{2}} & \frac{\sqrt{2}}{\sqrt{3}} & \frac{\sqrt{3}}{2} \end{bmatrix} \tag{4}$$

Alternatively, the possible states of the individual can be represented as:

$$\begin{aligned} q_i^t = {} & \tfrac{1}{2\sqrt{6}}|000\rangle + \tfrac{1}{2\sqrt{2}}|001\rangle + \tfrac{1}{2\sqrt{3}}|010\rangle + \tfrac{1}{2}|011\rangle \\ & + \tfrac{1}{2\sqrt{6}}|100\rangle + \tfrac{1}{2\sqrt{2}}|101\rangle + \tfrac{1}{2\sqrt{3}}|100\rangle \\ & + \tfrac{1}{2}|111\rangle \end{aligned} \tag{5}$$

Note that the square of above numbers are true probabilities, i.e. the above result means that the probabilities to represent the state $|000\rangle, |001\rangle, |100\rangle, |010\rangle$ are 1/24, 1/8, 1/24 and 1/12 respectively. Consequently, the three-qubits system of (4) could carry all eight states information at the same time.

Evolutionary computing with the qubit representation has a better characteristic of diversity than classical approaches since it can represent superposition of states. Only one qubit individual such as (4) is enough to represent eight states, whereas in classical representation eight individuals are needed. Additionally, along with the convergence of the quantum individuals, the diversity will gradually fade away and the algorithm converges.

3 New Structures for QEA

In structured QEA, the q-individuals are located in a structured population and each individual interacts only with its neighbors. This paper compares 11 structures for QEA which are Ring, Star, Btree, $K_{m,m}$, Cluster, $Random_h$ and the

structure which is proposed in [7]. The proposed structures are shown in Fig 1. Random$_h$ structure is the structure in which the neighbours of each q-individual are determined in each generation randomly. In this structure each q-individual is connected to h other q-individuals. The structured evolutionary algorithms have the advantage that the connections among neighbours help the algorithm to exploit possible solutions of the algorithm, and the overlapped small neighbourhoods help algorithm to explore the search space. This is similar to the Cellular Evolutionary Algorithms handling of population of diversity [5]. So the importance of the structured QEA is that the fitness and genotype diversity in the population is preserved for a long number of generations. This section tries to find the best structure for the QEA.

The procedure of the Crossover Interaction QEA is described in the following steps:

```
Procedure CQEA
begin
     t•0
  1. initialize Q(0) based on cellular
     structure.
  2. make X(0) by observing the states of
     Q(0).
  3. evaluate X(0).
  4. for all binary solutions x^0_ij in X(0)
     store x^0_ij to Best_ij
  5. while not termination condition do
     begin
      t•t+1
  6.  make X(t) by observing the states
       of Q(t-1)
  7.  evaluate X(t)
  8.  update Q(t) based on Best and X(t)
       using Q-gates
  9.  for all binary solutions x^t_ij in X(t) do
       begin
 10.     find Neighbors_ij in X(0).
 11.     select binary solution y with
           best fitness in Neighbors_ij
 12.     perform crossover operator on y
           and x^t_ij and make two offspring
 13.     select one offspring randomly
 14.     if offspring is fitter than Best_ij
           store it to Best_ij
       end
     end
end
```

Table 1. Lookup table OF $\Delta\theta$. $f(x)$ is the fitness of possible solution x and $f(b)$ is the fitness of best solution b

x_i	b_i	$f(x) \geq f(b)$	$\Delta\theta$
0	0	*false*	0
0	0	*true*	0
0	1	*false*	0.01π
0	1	*true*	0
1	0	*false*	-0.01π
1	0	*true*	0
1	1	*false*	0
1	1	*true*	0

CQEA has a population of quantum individuals $Q(t) = \{q_{ij}^t \mid i, j = 1,2,..., S\}$, where t is generation step and S is the size of lattice-like population.

The description of this algorithm can be as below:

1. In the initialization step, the quantum-individuals q^0_{ij} are located in a lattice-like environment. Then $[\alpha^0_{ij,k} \quad \beta^0_{ij,k}]^T$ of all q^0_{ij} are initialized with $1/\sqrt{2}$, where i,j=1,2,…,S is the location of the q-individuals in the lattice, k=1,2,...,m, m is the number of qubits in the individuals. This implies that each qubit individual q^0_{ij} represents the linear superposition of all possible states with equal probability.

2. This step makes a set of binary instants $X(t) = \{x_{ij}^t \mid i, j = 1,2,..., S\}$ at generation t=0 by observing $Q(0) = \{q_{ij}^0 \mid i, j = 1,2,..., S\}$ states, where $X(t)$ at generation t is a random instant of qubit population and S is the size of lattice. Each binary instant, x^t_{ij} of length m, is formed by selecting each bit using the probability of qubit, either $|\alpha^t_{ij,k}|^2$ or $|\beta^t_{ij,k}|^2$ of q^0_{ij}. Observing x^t_{ij} from qubit $[\alpha^0_{ij,k} \quad \beta^0_{ij,k}]^T$ is performed as below:

$$x^t_{ij,k} = \begin{cases} 0 & \text{if } \mathrm{R}(0,1) < |\alpha^t_{ij,k}|^2 \\ 1 & \text{otherwise} \end{cases} \qquad (6)$$

Where R(.,.) is a uniform random number generator.

3. Each instant x^t_{ij} is evaluated to give some measure of its fitness. In this step, the fitness of all binary solutions of $X(0)$ are evaluated.

4. In this step, all binary solutions x^0_{ij} in $X(0)$ are stored in $Best_{ij}$.

5. The while loop is terminated when the condition is satisfied. The termination condition can be considered as maximum generation or convergence condition.

6. Observing the binary solutions $X(t)$ from $Q(t)$.

7. Evaluating the binary solutions $X(t)$.

8. The quantum individuals are updated using Q-gate.

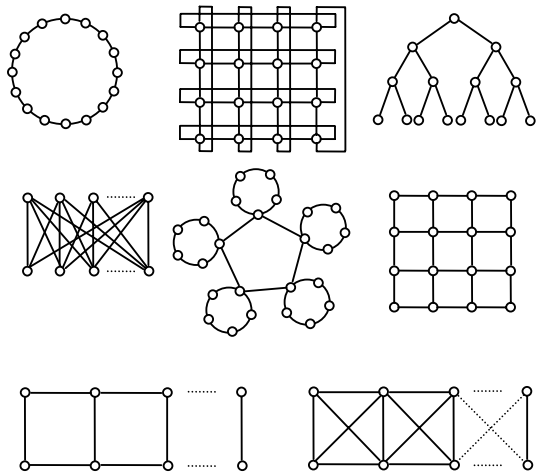

Figure 1. The compared structures. The structures from top, left to bottom right are Ring, Cellular, Btree, $K_{m,m}$, Cluster, Grid, Ladder and Crossed-ladder.

9. The "**for**" loop is running for all binary solutions x^t_{ij} (i,j=1,2,…,S) in the cellular structured population.
10. Finding the neighbours of the binary solution located on the location i,j and store it to $Neighbors_{ij}$.
11. Find the best possible solution in the $Neighbors_{ij}$, and store it to y.
12. Perform crossover operator on the possible solution y and x^t_{ij} . The crossover operator makes two new offspring.
13. Select one of two offspring randomly.
14. If the selected offspring is fitter than $Best_{ij}$, store it to $Best_{ij}$.

Table 2 shows the experimental results on the proposed structures. According to table 2 the best structure for QEA for the Knapsack problem and Trap problem is the cellular structure and the best structure for the numerical functions is the Random structure. For the numerical functions the best structure is $Random_2$ and $Random_4$ with 4 best results, after these structures, $Random_6$ places in third place with 3 best results. According to table 2, the original structure of QEA which is proposed in [5] can not reach the best results for any of the problems, so the proposed structures improve the performance of QEA consistently.

3.1 Quantum Gates Assignment

The common mutation is a random disturbance of each individual, promoting exploration while also slowing convergence. Here, the quantum bit representation can be simply interpreted as a biased mutation operator. Therefore, the current best individual can be used to steer the direction of this mutation operator, which will speed up the convergence. The evolutionary process of quantum individual is completed through the step of "update $Q(t)$." A crossover operator, quantum rotation gate, is described below. Specifically, a qubit individual q^t_{ij} in the cellular structure is updated using the rotation gate $U(\theta)$ in this algorithm. The k-th qubit of the quantum individual which is located at location $(i.j)$ in generation t $[\alpha^t_{ij,k} \quad \beta^t_{ij,k}]^T$ is updated as:

$$\begin{bmatrix} \alpha^{t+1}_{ij,k} \\ \beta^{t+1}_{ij,k} \end{bmatrix} = \begin{bmatrix} \cos(\Delta\theta) & -\sin(\Delta\theta) \\ \sin(\Delta\theta) & \cos(\Delta\theta) \end{bmatrix} \begin{bmatrix} \alpha^t_{ij,k} \\ \beta^t_{ij,k} \end{bmatrix} \qquad (7)$$

In this paper we use H_ε gate for updating the Q-bits [12]:

$$\left[\alpha^{t+1}_{ij} \quad \beta^{t+1}_{ij}\right]^T = H_\varepsilon\left(\alpha^t_{ij}, \beta^t_{ij}, \Delta\theta_j\right)$$

Where for $\left[\alpha'^{t+1}_{ij} \quad \beta'^{t+1}_{ij}\right]^T = R(\Delta\theta_j)\left[\alpha_{ij} \quad \beta_{ij}\right]^T$:

Table 2. The best Structure for QEA. The bold results are the best Ones. The results are averaged over 30 runs. The dimension of problem is set to m=100.

	Reference [7]	Cellular	Ring	Star	Ladder	Crossed Ladder	Cluster	$K_{m,m}$	Grid	Btree	$Random_2$	$Random_4$	$Random_6$
KP 1	396.32	409.65	**411.75**	405.13	411.05	408.41	407.12	405.42	411	410.53	372.74	377.88	387.92
KP 2	396.51	**397.82**	397.77	393.89	397.52	396.93	395.31	395.63	397.69	397.36	386.05	387.95	392.73
KR 1	558.41	**569.77**	569.77	569.77	569.77	569.77	569.77	569.5	569.77	569.77	510.67	537.23	551.84
KR 2	414.03	426.87	**430.1**	417.9	427.99	426.55	420.52	419.13	426.77	426.71	380.98	394.75	399.97
Trap	72.167	**80.333**	79.917	66.33	78.75	76	70.25	71.08	80.25	78.17	77.16	75.83	74.75
Schwefel 2.26	32211	40686	36000	34158	35779	35864	35178	35793	39451	37118	**42921**	39886	38626
Rastrigin	-1996.2	-1883.2	-2076.8	-2131.5	-1962.6	-2016.3	-2095.3	-1975.9	-1910.9	-2115	-1889.9	**-1739.7**	-1864.8
Ackley	-17.25	-17.18	-17.19	-17.85	-17.18	-17.19	-17.30	-17.63	**-17.16**	-17.20	-17.17	-17.20	-17.30
Griewank	-39.88	-36.18	-43.81	-42.16	-40.55	-37.77	-40.79	-37.45	-37.35	-41.17	-36.57	-34.31	**-33.59**
Penalized 1	-1.64e5	-1.58e5	-1.80e5	-1.82e5	-1.63e5	-1.64e5	-1.65e5	-1.64e5	-1.62e5	-1.80e5	-1.43e5	**-1.27e5**	-1.29e5
Penalized 2	-38381	-35606	-38797	-40565	-36923	-38030	-37729	-36021	-36454	-39727	**-34043**	-36755	-36746
Michalewicz	23.21	25.11	22.39	21.61	23.92	22.39	23.12	25.93	24.44	23.82	24.91	25.36	**27.19**
Goldberg	41.21	42.46	38.87	38.50	40.87	41.07	41.77	41.52	41.27	39.64	**45.00**	43.29	43.90
Sphere Model	-4.22e5	-4.03e5	-4.73e5	-5.47e5	-4.47e5	-4.56e5	-4.68e5	-4.49e5	-4.10e5	-4.50e5	**-3.80e5**	-4.08e5	-3.91e5
Schwefel 2.22	-4.84	-4.5152	-5.07	-5.3371	-5.00	-4.98	-4.94	-4.98	-4.89	-5.09	-4.66	-4.83	**-4.41**
Schwefel 2.21	-177.47	-175.91	-177.86	-178.26	**-174.22**	-176.95	-177.73	-176.95	-175.52	-175.39	-175.52	-176.3	-174.74
Dejong	-2.60e7	-2.08e7	-2.70e7	-3.73e7	-2.82e7	-2.68e7	-2.42e7	-2.55e7	-2.39e7	-2.84e7	-2.39e7	**-1.94e7**	-2.24e7
Rosenbrock	-1.12e5	-95848	-1.16e5	-1.31e5	-96417	-1.01e5	-1.09e5	-94559	-99086	-1.08e5	-84035	**-80515**	-82034
Kennedy	-1.08	-0.221	**-0.0003**	-12.42	-0.010	-0.210	-2.17	-6.90	-0.14	-0.125	-0.214	-1.17	-1.82

Table 3. Experimental results on the Knapsack problem, Trap problem and fourteen numerical function optimization problems. The number of runs was 20. **Mean** and **Std** represent the mean and standard deviation of best for 20 runs respectively. KP1 and KR1 means Knapsack penalty1 and Knapsack repair1

	$m=100$				$m=250$				$m=500$				$m=1000$			
	QEA		CQEA		QEA		CQEA		QEA		CQEA		QEA		CQEA	
	MEAN	STD	MEAN	STD	MEAN	STD	MEAN	STD	QEA	STD	QEA	STD	QEA	STD	QEA	STD
Schwefel [13]	4.98×10^{4}	1.87×10^{3}	5.75×10^{4}	1.89×10^{3}	8.7×10^{4}	3.8×10^{3}	9.9×10^{4}	5.9×10^{3}	1.3×10^{5}	2.2×10^{3}	1.5×10^{5}	6.7×10^{3}	2.1×10^{5}	7.5×10^{3}	2.7×10^{5}	1.1×10^{4}
Rastrigin [13]	-1.16×10^{-3}	95.25	-8.7×10^{2}	63.7	-4.9×10^{3}	1.8×10^{2}	-4.2×10^{3}	3.0×10^{2}	-1.2×10^{4}	2.9×10^{3}	-1.2×10^{4}	3.8×10^{2}	-2.7×10^{4}	4.1×10^{2}	-2.4×10^{4}	5.4×10^{2}
Ackley [13]	-17.7	0.05	-17.6	0.04	-18.10	5.7×10^{-2}	-17.99	3.4×10^{-2}	-18.37	1.3×10^{-2}	-18.27	2.4×10^{2}	-18.45	1.0×10^{-2}	-18.35	3.3×10^{-2}
Griewank [13]	-13.5	3.33	-8.97	1.19	-91.7	5.2	-69.3	4.4	-268.8	7.13	-239.7	12.3	-608.8	13.09	-548.8	18.38
Penalized 1 [13]	-6.14×10^{4}	7.2×10^{3}	-3.74×10^{4}	5.6×10^{-3}	-3.98×10^{5}	2.3×10^{4}	3.1×10^{5}	2.4×10^{4}	-1.1×10^{6}	2.3×10^{4}	-9.9×10^{5}	2.5×10^{4}	-2.3×10^{6}	2.18×10^{4}	-2.1×10^{6}	5.9×10^{4}
Penalized 2 [13]	-1.5×10^{4}	1.9×10^{3}	-1.04×10^{4}	2.9×10^{3}	-9.2×10^{4}	3.4×10^{3}	-7.6×10^{4}	6.3	-2.7×10^{5}	1.0×10^{4}	-2.4×10^{5}	4.0×10^{3}	-5.9×10^{5}	1.5×10^{4}	-5.1×10^{5}	1.4×10^{4}
Michalewicz [15]	35.6	1.97	45.2	1.94	56.5	2.78	69.5	4.16	77.05	5.52	101.9	2.12	121.7	5.89	190.1	5.65
Goldberg [14]	56.06	2.72	63.64	1.28	101.1	3.7	116.7	5.2	162.44	3.42	177.07	6.78	308.5	8.1	342.2	4.4
Sphere Model [13]	-1.7×10^{5}	2.4×10^{3}	-1.0×10^{5}	1.8×10^{4}	-9.8×10^{6}	4.5×10^{4}	-8.5×10^{-5}	6.3×10^{4}	-3.02×10^{6}	9.4×10^{4}	-2.69×10^{6}	1.2×10^{5}	-6.75×10^{6}	1.2×10^{5}	-5.87×10^{6}	2.3×10^{5}
Schwefel 2.22 [13]	-2.85	0.19	-1.93	0.14	-4.57	0.17	-3.9	0.19	-6.22	0.17	-5.86	0.11	-6.4	0.08	-6.1	0.13
Schwefel 2.21 [13]	-139.45	2.27	-145.15	9.02	-117	2.13	-177	2.32	-190.23	4.02	-188.83	1.37	-196.1	1.46	-194.8	0.75
Dejong [15]	-3.9×10^{6}	7.15×10^{5}	-2.9×10^{6}	4.3×10^{5}	-1.36×10^{8}	8.77×10^{6}	-8.91×10^{8}	1.44×10^{7}	-9.8×10^{9}	5.9×10^{7}	-7.6×10^{9}	9.3×10^{7}	-4.87×10^{9}	2.16×10^{8}	-3.65×10^{9}	2.37×10^{8}
Rosenbrock [14]	-2.12×10^{4}	2.2×10^{3}	-1.6×10^{4}	2.7×10^{3}	-2.1×10^{6}	9.6×10^{4}	-1.6×10^{5}	1.8×10^{4}	-8.3×10^{5}	3.1×10^{4}	-6.2×10^{5}	4.5×10^{4}	1.88×10^{6}	9.1×10^{4}	-1.39×10^{6}	1.17×10^{5}
Kennedy [14]	-7.9×10^{-2}	5.9×10^{-2}	-1.2×10^{-2}	6.1×10^{-3}	-9.93	3.28	-1.73	0.64	-60.9	5.6	-34.2	4.12	-147.14	9.77	-113.56	6.32
KP1	493.5	1.7×10^{-13}	493.5	1.7×10^{-13}	1371.1	9.4	1398.4	2.4×10^{-13}	2499.7	42.9	2745.5	10.1	4424.08	66.72	5101.26	44.13
KP2	425.2	1.3	426.4	0.5	978.3	7.0	1003.8	1.6	1878.6	11.8	2004.7	6.1	3588.35	46.07	3935.21	29.13
KR1	412.4	0.89	410.5	1.48	1023.3	6.5	1025.7	6.9	1865.7	14.3	1893.9	20.4	3601.9	34.7	3753.2	50.8
KR2	461.8	0.38	462.0	0.24	1044.6	1.2	1046.1	0.5	1974.4	7.3	2003.1	1.4	3899.43	33.04	4148.75	10.07
TRAP	85.2	1.43	87.5	1.92	193.5	3.27	212.8	3.16	335.5	6.9	389.2	6.9	579.73	8.27	672.4	18.92

$$\begin{bmatrix}\alpha_{ij}^{t+1} & \beta_{ij}^{t+1}\end{bmatrix}^T = \begin{cases}\begin{bmatrix}\sqrt{\varepsilon} & \sqrt{1-\varepsilon}\end{bmatrix}^T & \text{if } \left|\alpha_{ji}'^{t+1}\right|^2 \le \varepsilon \text{ and } \left|\beta_{ji}'^{t+1}\right|^2 \ge 1-\varepsilon \\ \begin{bmatrix}\sqrt{1-\varepsilon} & \sqrt{\varepsilon}\end{bmatrix}^T & \text{if } \left|\alpha_{ji}'^{t+1}\right|^2 \ge 1-\varepsilon \text{ and } \left|\beta_{ji}'^{t+1}\right|^2 \le \varepsilon \\ \begin{bmatrix}\alpha_{ij}^{t+1} & \beta_{ij}^{t+1}\end{bmatrix}^T & \text{otherwise}\end{cases}$$

Where $R(\Delta\theta_i)$ is rotation Gate and $\Delta\theta_i, i=1,2,...,n$ is the angel of rotation for each Q-bit.

4. Experimental Results

Here we used the Knapsack Problem, Trap Problem and 14 numerical benchmark functions for testing our proposed algorithm (see Appendix). The population size for all of the experiments is set at 25 (S=5), and maximum generation termination condition is used. All results were averaged over 20 runs. The value of $\Delta\theta$ is considered as Table 1 and the H_ε Gate is used with ε=0.01. In numerical function optimization problems, the binary coding is used and the number of bits for each binary number is set to 10 bits (per variable). Fig 2 shows the experimental results on Knapsack and Trap problems for m=1000. As it seen, our proposed algorithm can work better than QEA in knapsack and trap problem with dimension of 1000. We evaluated our proposed algorithm in dimension of m=100, m=250, m=500 and m=1000. Table 3 is summarized the experimental results. The number of runs for all experiments is 20. The time complexity of the proposed algorithm is similar to the conventional version of QEA but the performance of the proposed algorithm is better than QEA.

5. Conclusion

In this paper we proposed a cellular structure with a crossover interaction for neighbourhood interaction. The cellular structure of our proposed algorithm can make better population diversity and a better performance. Also the crossover interaction in our proposed algorithm can combine the advantages of the genetic algorithms with QEA. The crossover interaction can make a fine exploitation and the cellular structure and probabilistic representation of our algorithm can improve the exploration of proposed algorithm. In section 3 we showed that our proposed algorithm can work better than the QEA.

References

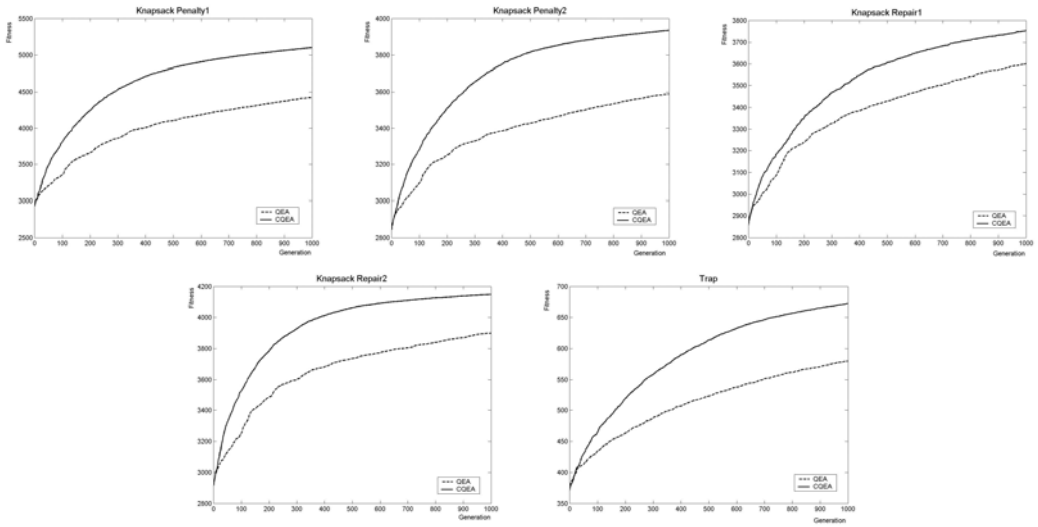

Fig 2. Experimental results on the Knapsack and Trap Problem. The number of runs was 20. The dimension of the problems in experiments is set at $m = 1000$.

[1] Anabela Borges Simões1 and Ernesto Costa, "Transposition versus crossover: an empirical study," *Genetic and Evolutionary Computation Conference* (GECCO 99) 1999.

[2] I. Harvey, "The Microbial genetic algorithm," *Submitted to Evolutionary Computation*. MIT Press 1996.

[3] N. Nawa, T. Furuhashi, T. Hashiyama and Y. Uchikawa, "A study of the discovery of relevant fuzzy rules using pseudo-bacterial genetic algorithm," *IEEE, Transactions on Industrial Electronics* 1999.

[4] N. Nawa and T. Furuhashi, "Bacterial evolutionary for fuzzy system design," *IEEE International Conference on Systems, Man and Cybernetics, IEEE Press* 1998.

[5] K. Han and J. Kim, "Quantum-inspired evolutionary algorithm for a class of combinatorial optimization," *IEEE Trans. on Evolutionary Computation,* vol., no. 6, 2002.

[6] K.H Han, K.H Park, C.H Lee and J.H Kim, "Parallel quantum-inspired genetic algorithm for combinatorial optimization problem," *Proceeding of the 2001 IEEE congress on Evolutionary Computation* Seoul, Korea, 2001.

[7] G Zhung, W Jin and L Hu, "Novel parallel quantum GAs," *Proceedings of the 4th International Conference on Parallel and Distributed Computing, Applications and Technologies, PDCAT,*2003.

[8] G Zhung, W Jin and L Hu, "Quantum evolutionary algorithm for multi-objective optimization," *Proceedings of the* 2003 *IEEE International Symposium on Intelligent Control Houston,* Texas October 5-8.2003.

[9] E. Cantú-Paz, *Efficient and Accurate Parallel Genetic Algorithms*, 2nd Ed, ser. Book Series on Genetic Algorithms and Evolutionary Computation. Norwell, MA: Kluwer, 2000, vol. I.

[10] E. Alba and M. Tomassini, "Parallelism and evolutionary algorithms," *IEEE Trans. Evol. Comput.,* vol. 6, no. 5, pp. 443–462, Oct. 2002.

[11] E. Alba and B. Dorronsoro, "The Exploration/ Exploitation Tradeoff in Dynamic Cellular Genetic Algorithms," *IEEE Trans. Evol. Comput.* vol. 9, pp. 126–142, 2005.

[12] K. H. Han and J. H. Kim. "Quantum-Inspired Evolutionary Algorithms with a New Termination Criterion, H_ε-Gate, and Two-Phase Scheme," *IEEE Trans. Evol. Comput.* vol. 8, pp. 156–169, 2004.

[13] W. Zhong, J. Liu, M. Xue and L. Jiao, "A Multi-agent Genetic Algorithm for Global Numerical Optimization," *IEEE Trans. Sys, Man and Cyber.* vol. 34, pp. 1128–1141, 2004.

[14] V. K. Koumousis and C. P. Katsaras. "A Saw-Tooth Genetic Algorithm Combining the Effects of Variable Population Size and Reinitialization to Enhance Performance," *IEEE Trans. Evol. Comput.* vol. 10, pp. 19–28, 2006.

[15] A.-R. Khorsand and M.-R. Akbarzadeh-T. "Quantum Gate Optimization in a Meta-Level Genetic Quantum Algorithm". *IEEE International Conference on Systems, Man and Cybernetics,* 2005.

Appendix

In this section two combinatorial optimization problems, Trap problem and Knapsack problem, and 14 function optimization problems are discussed to evaluate the proposed PDCQEA.

1. Trap problem

Trap problem is defined as:

$$f(x) = \sum_{i=0}^{N-1} \mathrm{Trap}(x_{5i+1}, x_{5i+2}, x_{5i+3}, x_{5i+4}, x_{5i+5}) \qquad (8)$$

Where N is the number of traps and

$$\mathrm{Trap}(x) = \begin{cases} 4 - \mathrm{ones}(x), & \text{if } \mathrm{ones}(x) \le 4 \\ 5 & \text{if } \mathrm{ones}(x) = 5 \end{cases} \quad (9)$$

Where the function "ones" returns the number of ones in the binary string x. Trap problem has a local optimum in $(0,0,0,0,0)$ and a global optimum in $(1,1,1,1,1)$.

2. Knapsack problem

Knapsack problem is a well-known combinatorial optimization problem which is in class of NP-hard problems [19]. Knapsack problem can be described as selecting various items x_i (i=1,2,…,m) with profits p_i and weights w_i for a knapsack with capacity C. Given a set of m items and a knapsack with capacity C, select a subset of the items to maximize the profit $f(x)$:

$$f(x) = \sum_{i=1}^{m} p_i x_i \ .$$

Subject to

$$\sum_{i=1}^{m} w_i x_i \le C \ .$$

In this paper we are considered:

$w_i = \mathrm{R}(1, v)$

$p_i = \mathrm{R}(1, v)$

Where $\mathrm{R}(\cdot,\cdot)$ is a uniform random number generator and v=10.

The use of QEA for solving Knapsack problem is described in [19].

3. Numerical Function Optimization

Global numerical optimization problems arise in many fields of science, engineering, and business. Since many of these problems cannot be solved analytically, GAs becomes one of the popular methods to address them [13]. There are some benchmark numerical functions for testing the optimization algorithms. Here we used 14 benchmark functions for testing the algorithms. The numerical benchmark functions are listed in Table 3.

These functions have some local minima and a global minimum. We used them for maximization process, so we used $-f(x)$ as fitness function.