Proc. of the Intl. Conf. on Advances In Computing, Electronics and Electrical Technology - CEET 2014.

ISBN: 978-1-63248-052-1 doi: 10.15224/ 978-1-63248-052-1-31

Quantum-Inspired Evolutionary Algorithm to Solve Graph Coloring Problem

Pronaya Prosun Das, Mozammel H. A. Khan

***Abstract*—Graph Coloring Problem (GCP) bears an enormous significance to the researchers in the field of soft computing. In this paper, we demonstrate a Quantum-Inspired Evolutionary Algorithm (QEA) to solve GCP. We use two dimensional arrays of Q-bits called Q-bit individual to produce binary individual. Q-gate operation is applied as a variation operator on Q-bit individuals. In traditional evolutionary algorithm (EA) for GCP, k-coloring approach is used and the EA is run several times for decreasing value of k until lowest possible k is reached. In our QEA, we start with the number of colors equal to the theoretical upper bound of the chromatic number, which is maximum out-degree + 1, and during evolution some colors are made unused to reduce the number of color in each generation. As a result, solution is found in a single run. We test 36 datasets from DIMACS benchmark and compare the result with several recent works. For five datasets, our algorithm obtains better solution than other.**

***Keywords*—quantum-inspired evolutionary algorithm (QEA), graph coloring problem (GCP), combinatorial optimization, Q-bit representation, Q-gate.**

I. Introduction

Graph coloring problem (GCP) is a well-known NP-hard problem [1]. GCP assigns different colors to the adjacent vertices of a graph using minimum number of colors. The GCP is illustrated with a simple graph in Fig. 1, where minimum number of colors needed to color eleven vertices is three. In Fig. 1, the vertex number is shown within the circle and the color number is shown outside the circle. The notable application of GCP are seen in pattern recognition [2], map coloring [3], radio frequency assignment [4], Bandwidth allocation [5], and timetable scheduling [6] etc.

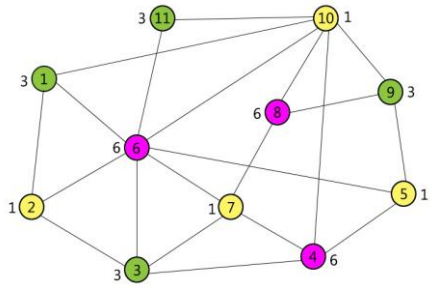


Fig.1: Example of graph coloring.

Assume, graph **G = (V, E)** is to be colored with **m** number of colors. The upper bound of the number **m** is **d+1**, where **d** is the maximum out degree of the graph.

Pronaya Prosun Das, Mozammel H A Khan
East West University
Bangladesh.

Our objective is to color all the vertices initially with **m** = **d+1** and then reducing **m** dynamically so that minimum chromatic number, denoted by **x(G)**, is found, that is **m = x(G)** is reached. In this paper, we propose a Quantum-inspired Evolutionary Algorithm (QEA) [7], which is the combination of concept of quantum computing [8] and evolutionary algorithm. It uses population of Q-bit individuals and Q-gate as a main variation operator.

II. Prior Work

One of the most recent works on GCP is Memetic Algorithm (MA) [11] that used binary encoded chromosome. Population is updated mainly using classical crossover operator. Offsprings are corrected if needed. Then a deterministic improvement technique is applied on the corrected offsprings to improve the solution quality. Another work on GCP [12] combines wisdom of artificial crowds approach with the genetic algorithm (GA). In this approach, multiple parent selection and multiple mutations based on the closeness of the solution to the global optima are used. The algorithm is run several times for several decreasing values of ***k*** and the minimum possible ***k*** value is taken as the minimum chromatic number. A guided genetic algorithm for GCP called MSPGCA is reported in [13], where the authors fine-tuned the initial chromosomes using a simple genetic algorithm and then the deterministic MSPGCA algorithm is run to dynamically reduce the chromatic number. In paper [14], the authors have proposed a hybrid algorithm to solve GCP. GA has a highly degenerate objective function. In order to compensate for this degeneracy, bitstream neuron (Boltzmann Machine) was applied to the solution obtained from GA. A hybrid immune algorithm is also applied on GCP [15]. All the above mentioned approaches used integer encoding for the chromosomes except the paper [11].

III. Methodology

Representation of the Graph is discussed first and then the proposed QEA for GCP is discussed.

A. *Representation*

A Q-bit is defined as the smallest unit of information [8] in QEA, which is defined with a pair of numbers **(α, β)**, where $|\alpha|^2 + |\beta|^2 = 1$. $|\alpha|^2$ and $|\beta|^2$ gives the probability that the Q-bit will be found in the "0" state and the "1" state, respectively. For GCP, we use two-dimensional array of Q-bits as a Q-bit individual, where each row corresponds to a color and each column corresponds to a vertex. Later binary individuals are produced from Q-bit individuals. If the ***j***th vertex be colored using the ***i***th color, then the **(*i*, *j*)**th element of the array is 1 and the other elements are 0s. Thus, in a valid chromosome, every column must have a single 1 and a row

Proc. of the Intl. Conf. on Advances In Computing, Electronics and Electrical Technology - CEET 2014.

ISBN: 978-1-63248-052-1 doi: 10.15224/ 978-1-63248-052-1-31

will have one or more than one 1s placed on non-adjacent vertices columns. The encoding scheme is illustrated in Fig. 2. A row may also have all 0s, in which case the color is not used in the solution. If a column has all 0s (the vertex is not colored) or more than one 1s (more than one color is assigned to that vertex), then the encoding is invalid. On the other hand, if a row has 1s in adjacent vertices columns (same color is assigned to adjacent vertices), then the encoding is invalid. These situations may arise during creation of the binary individual from Q-bit individual. When a binary individual becomes invalid, then it is corrected using **repair** procedure. Thus the number of rows having at least one 1 is the number of used colors and is used as the fitness function in our QEA.

	1	2	3	4	5	6	7	8	9	10	11
1	0	**1**	0	0	**1**	0	**1**	0	0	**1**	0
2	0	0	0	0	0	0	0	0	0	0	0
3	**1**	0	**1**	0	0	0	0	0	**1**	0	**1**
4	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	**1**	0	**1**	0	**1**	0	0	0

Fig. 2: Proposed binary individual for GCP of the graph of Fig. 1.

B. Proposed QEA to solve GCP

We present the detailed algorithm of QEA for the Graph Coloring. The Graph Coloring problem is considered to demonstrate the applicability of QEA to the combinatorial optimization problem.

Procedure QEA for the GCP
01: **begin**
02: t ← 0
03: initialize Q(t)
04: **make** P(t) by observing the states of Q(t)
05: **repair** P(t)
06: evaluate P(t)
07: store the best solutions among P(t) into B(t)
08: **while** (t < MAX GEN) **do**
09: **begin**
10: t ← t + 1
11: **make** P(t) by observing the states of Q(t - 1)
12: **repair** P(t)
13: evaluate P(t)
14: **update** Q(t)
15: store the best solutions among B(t - 1) and P(t) into B(t).
16: store the best solution b among B(t).
17: **if** (migration-period)
18: **then** migrate b or b_j^t to B(t) globally or locally, respectively
19: **end**
20: **end**

Here, length of row and column of Q-bit individual is **m** and **v** respectively for GCP. **m** is the number of colors and start with the value equal to 'maximum out degree+1'and **v** is the number of vertices. QEA maintains a population of Q-bit individuals, $\boldsymbol{Q(t)} = \{\boldsymbol{q_1^t}, \boldsymbol{q_2^t}, \boldsymbol{q_3^t}, \dots, \boldsymbol{q_k^t}\}$ at generation $\boldsymbol{t}$, where $\boldsymbol{k}$ is the size of population, and $\boldsymbol{q_p^t}$ is a Q-bit individual.

$$q_p^t = \begin{bmatrix} \alpha_{p_{1,1}}^t & \alpha_{p_{1,2}}^t \dots \alpha_{p_{1,v}}^t \\ \beta_{p_{1,1}}^t & \beta_{p_{1,2}}^t \dots \beta_{p_{1,v}}^t \\ \alpha_{p_{2,1}}^t & \alpha_{p_{2,2}}^t \dots \alpha_{p_{2,v}}^t \\ \beta_{p_{2,1}}^t & \beta_{p_{2,2}}^t \dots \beta_{p_{2,v}}^t \\ & \vdots \\ & \vdots \\ \alpha_{p_{m,1}}^t & \alpha_{p_{m,2}}^t \dots \alpha_{p_{m,v}}^t \\ \beta_{p_{m,1}}^t & \beta_{p_{m,2}}^t \dots \beta_{p_{m,v}}^t \end{bmatrix}$$

Where, $\boldsymbol{m \times v}$ is the number of Q-bits in an individual and $\boldsymbol{p = 1, 2, \dots, k}$. In the step of **initialize Q(t),** all $\boldsymbol{\alpha_{i,j}^0}$ and $\boldsymbol{\beta_{i,j}^0}$ are initialized with $1/\sqrt{2}$, where, $i = 1, 2, \dots, v$ and $\boldsymbol{j = 1, 2, \dots, m}$. On line 04 and 11, QEA produce population of binary individuals, $\boldsymbol{P(t)} = \{\boldsymbol{x_1^t}, \boldsymbol{x_2^t}, \dots, \boldsymbol{x_k^t}\}$ from population of Q-bit individuals, where **t = 0, 1, 2, … .** For notational simplicity, $\boldsymbol{x}$ and $\boldsymbol{q}$ are used instead of $\boldsymbol{x_p^t}$ and $\boldsymbol{q_p^t}$ respectively. The following **make** procedure is used to obtain the binary string $\boldsymbol{x}$.

Procedure make (x)
01: **begin**
02: i ← 0
03: j ← 0
04: **while** (i < m) **do**
05: **begin**
06: i ← i + 1
07: **while** (j < v) **do**
08: **begin**
09: j ← j + 1
10: **if (**random [0, 1) < $|\beta_{i,j}|^2$**)**
11: **then** $x_{i,j} \leftarrow 1$
12: **else** $x_{i,j} \leftarrow 0$
13: **end**
14: **end**
15: **end**

Binary individuals are repaired if needed. Two possible problems may occur – (i) a column may have multiple 1s or (ii) a column may have all 0s. We also have to ensure that adjacent vertices have colored with different colors.

	1	2	3	4	5	6	7	8	9	10	11
1	0	**1**	0	0	0	**1**	0	0	**1**	0	**1**
2	0	0	0	0	0	**1**	0	**1**	0	0	0
3	**1**	0	0	**1**	0	0	**1**	0	**1**	0	**1**
4	0	0	**1**	0	0	0	0	0	**1**	0	0
5	0	0	0	0	0	0	0	**1**	0	0	**1**
6	**1**	0	0	**1**	0	**1**	0	0	**1**	0	0

Fig. 3: Invalid Chromosome.

The chromosome shown in Fig. 3 is invalid and has the two possible problems. Invalid chromosomes are corrected by **repair** procedure. For case (i), only one 1 is kept and for case

Proc. of the Intl. Conf. on Advances In Computing, Electronics and Electrical Technology - CEET 2014.

ISBN: 978-1-63248-052-1 doi: 10.15224/ 978-1-63248-052-1-31

(ii), a 1 is inserted. Both are done at a selected row among used color cluster which is sorted according to number of uses so that no conflict is created at that row (breaking tie randomly). If such a used color row is not available, then a 1 is inserted at a randomly selected row among unused color clusters. A possible corrected version of invalid chromosome of Fig. 3 is shown in Fig. 4.

	1	2	3	4	5	6	7	8	9	10	11
1	0	**1**	0	0	**1**	0	**1**	0	0	**1**	0
2	0	0	0	0	0	0	0	0	0	0	0
3	**1**	0	**1**	0	0	0	0	0	**1**	0	**1**
4	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	**1**	0	0	0
6	0	0	0	**1**	0	**1**	0	0	0	0	0

Fig. 4: Corrected Chromosome of Fig. 3.

Binary chromosomes are corrected using **repair** procedure as follows:

Procedure Repair (x)
01: **begin**
02: **for** (all vertex **v** in random order) **do**
03: **begin**
04: iscolored←false
05: **for** (all color **i** which is assigned to **v** in a ordering most to least used) **do**
06: **begin**
07: **if (**iscolored**)**
08: **then** remove color **i** from **v**
09: **else if (v** is colorable with **i)**
10: **then** iscolored←true
11: **end**
12: **end**
14: **for (**all vertex **v** which is not colored**) do**
15: **begin**
16: iscolored←false
17: **for** (all used color **i** in a ordering most to least used) **do**
18: **begin**
19: **if (not** iscolored **and v** is colorable with **i)**
20: **then** {
21: color the vertex, **v** with **i**
22: iscolored←true
23: }
24: **end**
25: **if** (**not** iscolored)
26: **then** color the vertex **v** with an unused color
27: **end**
28: **end**

The **update** procedure of Q-bits is presented below:

Procedure Update (q)
01: **begin**
02: i ← 0
03: j ← 0
04: **while** (i < m) **do**
05: **begin**
06: i ← i + 1
07: **while** (j < v) **do**
08: **begin**
09: j ← j + 1
10: Determine $\Delta\theta_{i,j}$ with the lookup table
11: Obtain $(\alpha_{i,j}, \beta_{i,j})$ from the following:
12: **if** (**q** is located in the first/third quadrant)
13: **then** $[\alpha'_{i,j}\ \beta'_{i,j}]^T = H_\epsilon(\alpha_{i,j},\ \beta_{i,j},\ \Delta\theta_{i,j})$
14: **else** $[\alpha'_{i,j}\ \beta'_{i,j}]^T = H_\epsilon(\alpha_{i,j},\ \beta_{i,j}, -\Delta\theta_{i,j})$
15: **end**
16: **end**
17: $\boldsymbol{q} \leftarrow \boldsymbol{q}'$
18: **end**

Here, $\boldsymbol{H_\epsilon}$ gate is used as a Q-gate to update a Q-bit individual $\boldsymbol{q}$ as a variation operator. Q-bit of ***i***th row and ***j***th column $(\boldsymbol{\alpha_{i,j}}, \boldsymbol{\beta_{i,j}})$ is updated as follows:

$$[\alpha''_{i,j}\ \beta''_{i,j}]^T = U(\Delta\theta_{i,j})[\alpha'_{i,j}\ \beta'_{i,j}]^T :$$

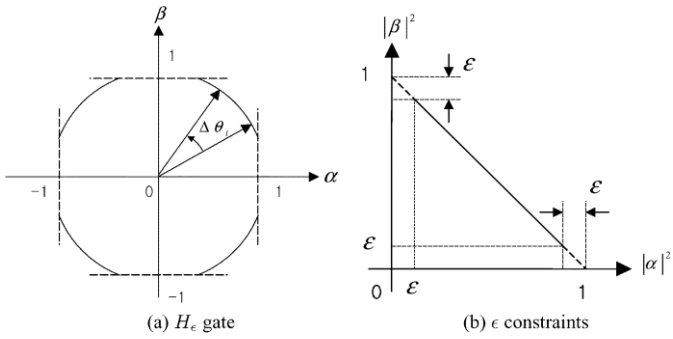
Where, $U(\Delta\theta_{i,j})$ is a simple rotation gate,

$$U(\Delta\theta_{i,j}) = \begin{bmatrix} cos(\Delta\theta_{i,j}) & -sin(\Delta\theta_{i,j}) \\ sin(\Delta\theta_{i,j}) & cos(\Delta\theta_{i,j}) \end{bmatrix}$$

i) if $|\alpha''_{i,j}|^2 \le \epsilon\ and\ |\beta''_{i,j}|^2 \ge 1-\epsilon$
$[\alpha'_{i,j}\ \beta'_{i,j}] = [\sqrt{\epsilon}\ \sqrt{1-\epsilon}]^T$;

ii) if $|\alpha''_{i,j}|^2 \ge 1-\epsilon\ and\ |\beta''_{i,j}|^2 \le \epsilon$
$[\alpha'_{i,j}\ \beta'_{i,j}] = [\sqrt{1-\epsilon}\ \sqrt{\epsilon}]^T$;

iii) otherwise
$[\alpha'_{i,j}\ \beta'_{i,j}] = [\alpha''_{i,j}\ \beta''_{i,j}]$;



(a) H_ϵ gate (b) ϵ constraints

Fig. 5: $\boldsymbol{H_\epsilon}$ gate based on rotation gate.

In this QEA for GCP, the angle parameters used for the rotation gate are shown in Table 1. Let us define an angle vector $\boldsymbol{\Theta} = [\boldsymbol{\theta_1}\boldsymbol{\theta_2} \ldots \boldsymbol{\theta_8}]^{\boldsymbol{T}}$, where $\boldsymbol{\theta_1}, \boldsymbol{\theta_2}, \ldots, \boldsymbol{\theta_8}$ can be found from table 1. We have used, $\boldsymbol{\theta_3} = \boldsymbol{0.01\pi}, \boldsymbol{\theta_5} = \boldsymbol{-0.01\pi}$, and 0 for the rest. The values from 0.001π to 0.05π are recommended for the magnitude of $\boldsymbol{\Delta\theta_{i,j}}$. Otherwise, it may converse prematurely. The sign of $\Delta\theta_{i,j}$ determines the

Proc. of the Intl. Conf. on Advances In Computing, Electronics and Electrical Technology - CEET 2014.

ISBN: 978-1-63248-052-1 doi: 10.15224/ 978-1-63248-052-1-31

direction of convergence. We have chosen $\boldsymbol{\epsilon = 0.01}$. In table 1, $\boldsymbol{f}(.)$ is the fitness, and $\boldsymbol{x_{i,j}}$ and $\boldsymbol{b_{i,j}}$ are the **(i, j)**th bits of the best solution $\boldsymbol{x}$ and the binary solution $\boldsymbol{b}$, respectively. In the QEA for GCP, $\boldsymbol{\theta_1 = 0}$, $\boldsymbol{\theta_2 = 0}$, $\boldsymbol{\theta_3 = 0.01\pi}$, $\boldsymbol{\theta_4 = 0}$, $\boldsymbol{\theta_5 = -0.01\pi}$, $\boldsymbol{\theta_6 = 0}$, $\boldsymbol{\theta_7 = 0}$, $\boldsymbol{\theta_8 = 0}$ are used.

Table 1: lookup table of $\Delta\boldsymbol{\theta_{i,j}}$

$x_{i,j}$	$b_{i,j}$	$f(x) \geq f(b)$	$\Delta\theta_{i,j}$
0	0	$false$	θ_1
0	0	$true$	θ_2
0	1	$false$	θ_3
0	1	$true$	θ_4
1	0	$false$	θ_5
1	0	$true$	θ_6
1	1	$false$	θ_7
1	1	$true$	θ_8

In line 18 of **Procedure QEA for the GCP,** migration is defined as the process of copying current best solution in binary population in place of previous solutions. A local migration is implemented by replacing some of the population by the best individual, while global migration is implemented by replacing all the solution by the best individual.

IV. Results

We have implemented our algorithm in C++ programming language and compiled using 32 bit TDM-GCC compiler, version 4.8.1. Tests were run on a PC having following configuration:

CPU: Intel Core i3-2350M 2.30 GHz

Memory: 4 GB DDR3 1333MHz

Operating System: Windows 7 64-bit

Datasets used to test our QEA for GCP are taken from Center for Discrete Mathematics and Theoretical Computer Science (DIMACS) benchmarking graph collection [9] and [10]. Instances ending in .col are in DIMACS standard format. Instances in .col.b are in compressed format. We have used datasets ending with .col extension. The top of the dataset heading resembling "p edge 11 20" means that graph has 11 vertices and 20 edges, where p denotes vertices. After that number of lines like "e 1 2" represent connection between two edges. We experiment with 36 datasets from [9] and [10]. The tested datasets are heterogeneous consisting of big graph like 5-FullIns_4.col having 1085 vertices, highly dense graph like miles1500.col, highly complex graph like queen10_10.col, and even simple graphs. Our results are tabulated in Table 2 and compared with other results. For 29 datasets, we found expected chromatic number as stated in [9] and [10]. For dataset queen10_10.col, 1-FullIns_3.col, 1-FullIns_5.col, 2-FullIns_5.col, 3-FullIns_3.col, 3-FullIns_4.col, 5-FullIns_4.col, no expected chromatic number is stated there. We get better result for 2-FullIns_5.col, 5-FullIns_4.col compared to the result of paper [14]. For queen8_8.col our result is 9, which is better than the result of papers [13] and [14]. Our QEA produced more optimal result for queen8_12.col than result of [13]. For queen10_10.col, we get chromatic number 12, whereas this number found in paper [11] and paper [13] are 13 and 14, respectively.

Table 2: Comparison of Obtained Results with Other Works.

Dataset	|V|	|E|	x(G)	[12]	[13]	[14]	[11]	QEA
myciel4.col	23	71	**5**	5	-	-	5	**5**
myciel5.col	47	236	**6**	6	-	-	6	**6**
myciel6.col	95	755	**5**	-	-	-	7	**5**
myciel7.col	191	2360	**8**	-	-	-	8	**8**
games120.col	120	638	**9**	9	9	9	9	**9**
huck.col	74	301	**11**	11	11	11	11	**11**
jean.col	80	254	**10**	10	10	10	10	**10**
david.col	87	406	**11**	11	11	11	11	**11**
queen5_5.col	25	160	**5**	5	5	-	5	**5**
queen6_6.col	36	290	**7**	7	8	-	7	**7**
queen7_7.col	49	476	**7**	7	7	7	7	**7**
queen8_8.col	**64**	**728**	**9**	**-**	**11**	**11**	**-**	**9**
queen8_12.col	**96**	**1368**	**12**	**-**	**14**	**-**	**-**	**12**
queen10_10.col	**100**	**2940**	**?**	**-**	**14**	**-**	**13**	**12**
anna.col	138	493	**11**	11	11	11	11	**11**
homer.col	561	1629	**13**	13	13	13	13	**13**
miles250.col	128	387	**8**	8	-	-	8	**8**
miles500.col	128	1170	**20**	-	-	-	20	**20**
miles750.col	128	4226	**31**	-	-	-	31	**31**
miles1000.col	128	3216	**42**	42	42	42	42	**42**
miles1500.col	128	5198	**73**	-	73	-	73	**73**
zeroin.i.1.col	211	4100	**49**	-	-	-	49	**49**
zeroin.i.2.col	211	3541	**30**	-	-	-	-	**30**
zeroin.i.3.col	206	3540	**30**	-	-	-	-	**30**
2-Insertions_3.col	37	72	**4**	-	-	4	4	**4**
inithx.i.1.col	864	18707	**54**	-	-	-	-	**54**
inithx.i.2.col	645	13979	**31**	-	-	-	-	**31**
mulsol.i.1.col	197	3925	**49**	-	49	-	49	**49**
fpsol2.i.1.col	496	11654	**65**	65	-	-	-	**65**
mulsol.i.5.col	186	3973	**31**	-	-	-	-	**31**
1-FullIns_3.col	30	100	**?**	-	-	-	-	**4**
1-FullIns_5.col	282	3247	**?**	-	6	6	-	**6**
2-FullIns_5.col	**852**	**12201**	**?**	**-**	**-**	**11**	**-**	**7**
3-FullIns_3.col	80	346	**?**	-	-	-	-	**6**
3-FullIns_4.col	405	3524	**?**	-	7	7	-	**7**
5-FullIns_4.col	**1085**	**11395**	**?**	**-**	**-**	**11**	**-**	**10**

Fig. 6 shows the average fitness (number of used color) and minimum fitness over successive generation for queen7_7 dataset indicating the dynamicity of our algorithm. In our experiments, we found that rotation probability of 0.7 performs better for all datasets. The termination condition also depends on the graph complexity. If both the average fitness and the best fitness do not change for a specified number of generations or the optimal known solution is found, then the algorithm is terminated. The number of generations varies with the graph complexity. Our algorithm is very fast because it can work with small population and also it needs less

Proc. of the Intl. Conf. on Advances In Computing, Electronics and Electrical Technology - CEET 2014.

ISBN: 978-1-63248-052-1 doi: 10.15224/ 978-1-63248-052-1-31

generation than other evolutionary algorithms to get the optimal solution. In our experiment, we have used population size within 5 to 50 for different datasets.

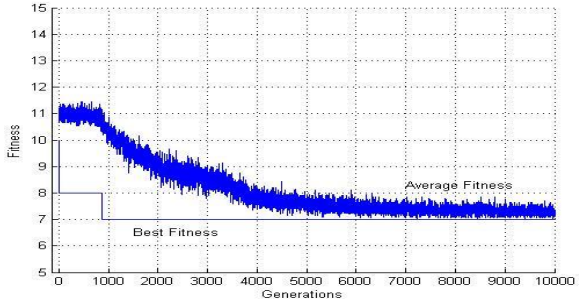


Fig. 6: Average and best (minimum) fitness for queen7_7 dataset.

V. Conclusion and Future Work

In this paper, we proposed a Quantum-inspired Evolutionary Algorithm (QEA) for graph coloring problem (GCP). The main variation operator of our QEA is the rotation gate. The population of the solution is updated using only rotation gate. We compare best binary individual with all binary individuals in the population and update the Q-bit individual. Because of the nature of the encoding, in each generation, binary individual may become invalid and in that case binary individuals are corrected to obtain the valid solution. After certain generations, all or partial population are replaced with best binary individual to avoid local optimization. We start with **m=d+1** colors, where **d** is the maximum out degree of the graph. The number **m** is the upper bound of the chromatic number. That means, in a single run, the QEA dynamically reduces the chromatic number starting with upper bound to the possible minimum number. Unlike the previous techniques, our QEA finds the minimum chromatic number in a single run reducing the total time significantly. We experiment with 36 datasets from [9] and [10]. For 29 datasets, we found expected chromatic number as stated in [9] and [10]. For five datasets queen8_8.col, queen8_12.col, queen10_10.col, 2-FullIns_5.col and 5-FullIns_4.col, our QEA produce better result than the previous works. So we can say these are the major achievement of our algorithm over the previous works. In the future, we will try to improve the execution time of the algorithm. We also have plans to compare more results of our proposed approach with the results produced by other algorithm.

References

[1] M. Kubale, “Graph colorings,” American Mathematical Society, 2004.

[2] [2] C. W. K. Chen and D. Y. Y. Yun, “Unifying graph-matching problem with a practical solution,” in Proc. International Conf. on Systems, Signals, Control, Computers, September 1998.

[3] [3] B. H. Gwee, M. H. Lim, and J. S. Ho, “Solving four-colouring map problem using genetic algorithm,” in Proc. Artificial Neural Networks and Expert Systems, 1993.

[4] [4] W. K. Hale, “Frequency assignment: theory and applications,” in Proc. IEEE, 1980, vol. 12, pp. 1497-1514.

[5] A. Gamst, “Some lower bounds for class of frequency assignment problems,” IEEE Trans. on Vehicular Technology, vol. 35, no. 1, pp. 8-14, 1986.

[6] N. K. Cauvery, “Timetable scheduling using graph coloring,” International Journal of P2P Network Trends and Technology, vol. 1, issue 2, pp. 57-62, 2011.

[7] Kuk-Hyun Han and Jong-Hwan Kim, "Quantum-inspired Evolutionary Algorithm for a Class of Combinatorial Optimization," *IEEE Transactions on Evolutionary Computation*, IEEE Press, Vol. 6, No. 6, pp. 580-593, December 2002.

[8] T. Hey, “Quantum computing: An introduction,” in Computing & Control Engineering Journal. Piscataway, NJ: IEEE Press, June 1999, vol. 10, no. 3, pp. 105–112

[9] M. Trick. (2013, March 1). Graph coloring instances. Michael Trick's Operations Research Page. [Online]. Available: http://mat.gsia.cmu.edu/COLOR/instances.html

[10] “DIMACS benchmarks”, [online] Available: http://www.cs.hbg.psu.edu/benchmarks/graphcoloring.html

[11] Hasin Al Rabat Chowdhury, Tasneem Farhat, and Mozammel H. A. Khan, "Memetic Algorithm to Solve Graph Coloring Problem," International Journal of Computer Theory and Engineering vol. 5, no. 6, pp. 890-894, 2013.

[12] M. M. Hindi and R. V. Yampolskiy, “Genetic algorithm applied to the graph coloring problem,” in Proc. 23rd Midwest Artificial Intelligence and Cognitive Science Conf., April 2012, pp. 61-66.

[13] B. Ray, A. J. Pal, D. Bhattacharyya, and T. Kim, “An efficient GA with multipoint guided mutation for graph coloring problems,” International Journal of Signal Processing, Image Processing and Pattern Recognition, vol. 3, no. 2, pp. 51-58, 2010.

[14] Timir Maitra, Anindya J. Pal, Tai-hoon Kim, Debnath Bhattacharyya: Hybridization of Genetic Algorithm with Bitstream Neurons for Graph Coloring, International Journal of U- & E-Service, Science & Technology . Sep2010, Vol. 3 Issue 3, p37-53. 17p.1 Chart.

[15] V. Cutello, G. Nicosia, and M. Pavone, “A hybrid immune algorithmwith information gain for the graph coloring problem,” in *Proc.GECCO 2003*, pp. 171-182, 2003.

About Author (s):

Pronaya Prosun Das was born in Manikganj, Bangladesh. He is a final year student of Computer Science and Engineering at East West University, Aftabnagar, Dhaka 1212, Bangladesh. His research interests include Evolutionary Algorithms, Machine Learning and Image Processing.

Mozammel H A Khan was born in Kushtia, Bangladesh. He obtained B. Sc. Engg. (EEE), M. Sc. Engg. (Comp. Engg.), and PhD (Comp. Sc. &Engg.) degrees from Bangladesh University of Engineering and Technology (BUET), Dhaka, Bangladesh. He served as faculty member of EEE Department at Bangladesh Institute of Technology, Rajshahi, Bangladesh and Computer Science and Engineering Department at Khulna University, Khulna, Bangladesh. Currently, he is a full Professor of Computer Science and Engineering Department at East West University, Dhaka, Bangladesh. His research interest includes broad areas of Logic Synthesis, Reversible Logic, Multiple-valued Logic, and Soft Computing. He has published over 85 research papers. He is a Senior Member of IEEE.