2nd Int'l Conf. on Electrical Engineering and Information & Communication Technology (ICEEICT) 2015
Jahangirnagar University, Dhaka-1342, Bangladesh, 21-23 May 2015

Solving Maximum Clique Problem using a Novel Quantum-inspired Evolutionary Algorithm

Pronaya Prosun Das
Department of Computer Science and Engineering
Jahangirnagar University
Dhaka, Bangladesh.
pronaya.prosun@gmail.com

Mozammel H. A. Khan
Department of Computer Science and Engineering
East West University
Dhaka, Bangladesh.
mhakhan@ewubd.edu

***Abstract*— Maximum Clique Problem (MCP) is one of the most important NP-hard problems in the area of soft computing and it has many real world applications in numerous fields ranging from coding theory to the determination of the structure of a protein molecule. Different heuristic, Meta heuristic and hybrid solution approaches have been applied to obtain the solution. In this paper, we demonstrate a Quantum-inspired Evolutionary Algorithm (QEA) to solve MCP. We have used one dimensional arrays of Q-bits called Q-bit individuals to produce binary individuals. After production of binary individuals, we have repaired and improved them. Here, Q-gate is the main variation operator applied on Q-bit individuals. Our algorithm was tested on DIMACS benchmark graphs and 40 of them were tested. The results obtained here are extremely encouraging. For almost all of the datasets, we get the optimal results reported on DIMACS benchmark and also compared our results with other related works. For some cases we get better results than other works.**

Keywords—Maximum clique problem (MCP), quantum-inspired evolutionary algorithm (QEA), combinatorial optimization, NP-hard problems, Q-bit representation, Q-gate, DIMACS.

I. INTRODUCTION

The maximum clique problem (MCP) is one of the most classically studied combinatorial optimization problem. It comes from graph theory and has many real world applications. The maximum clique problem can be briefly defined as follows.

Let G = (V, E) be an undirected graph of n vertices, where V represents vertex set and E represents edge set. A clique in G is a sub graph of G in which there is an edge between any two vertices. The size of a clique is the number of vertices in the clique. The maximum clique problem (MCP) is to find the largest clique in a given graph G. Finding out a clique, in a graph, is one of the original NP-hard [1] problems. That means there is no polynomial time algorithm to solve MCP. Therefore, using polynomial time heuristic algorithms such as evolutionary algorithms, greedy heuristics, simulated annealing, tabu search [1, 2] etc. are used to solve maximum clique problem.

The problem is very important as we have seen notable application of MCP in the area of coding theory, fault diagnosis, printed circuit board testing and computer vision [1] etc. It has recently been established that MCP can be used in matching protein structures [3]. Our objectives of the research is to find the largest complete sub graph in a given undirected graph G = (V, E). Let assume n is the clique size of that graph and we have to increase n dynamically so that maximum clique, denoted by ω(G), is found, that means n = ω(G) is reached.

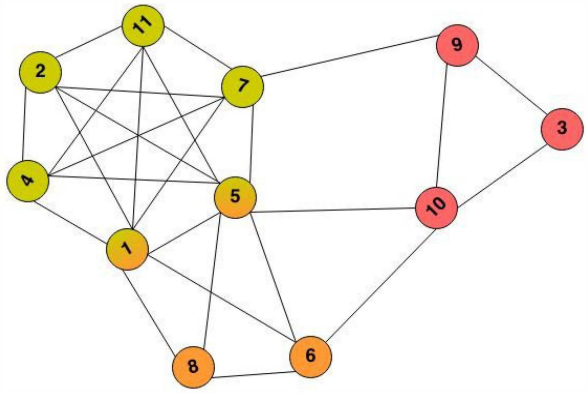

Figure 1: Example of maximum clique problem.

In this paper, we propose a Quantum-inspired Evolutionary Algorithm (QEA) [4, 13, 14], which is the combination of concept of quantum computing [5] and evolutionary algorithm. It uses population of Q-bit individuals and Q-gate as a main variation operator. As Q-bit individuals may show linear superposition of states, it has a better characteristic of population diversity than any other representation.

This paper is organized as follows: Section II explains about related works. The methodology of the research is described in section III. In section IV, the experimental results are discussed. Finally the conclusion is outlined in Section V.

II. RELATED WORKS

We have proposed a novel quantum-inspired evolutionary algorithm to solve maximum clique problem. So as to get an idea of the previous works, many research works have been studied. In the paper [6], a branch and bound algorithm for maximum clique problem was introduced. It uses existing clique finding and vertex coloring heuristics to determine lower and upper bounds for the size of a maximum clique. In another paper [7] an efficient partition-based algorithm is proposed for MCP. They solved large graph with limited memory and further reduce the high cost of CPU computation by a careful nested partition based on a cost model. Another work on MCP introduced variable depth search based algorithm [8], called k-opt local search (KLS) which uses a sequence of add and drop moves to

978-1-4673-6676-2/15/$31.00 ©2015 IEEE

solve maximum clique problem. Genetic algorithm (GA) is used in a paper [9]. Here, traditional crossover based operator and mutation are used to produce new population. Evolutionary algorithm (EA) have been reported in paper [10]. They introduces a new operator, called guided mutation which generates offspring through combination of global statistical information and the location information of solutions found so far. Besides they adopts a strategy for searching different search areas in different search phases. Then Marchiori's heuristic is applied to each new solution to produce a maximum clique.

III. METHODOLOGY OF THE RESEARCH

Here, we are going to start our discussion with the proposed representation and the algorithm will be described later.

A. Representation

We have used Q-bit based on the concept of qubits [5] of quantum computer in our QEA. A Q-bit is also 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 MCP, we have used one-dimensional array of Q-bits as a Q-bit individual, where each index corresponds to a vertex. Later binary individuals are produced from Q-bit individuals. If the ***i***th vertex is a part of a maximum clique, then the ***i***th element of the array is 1 otherwise it is 0. Thus, in a valid chromosome, 1s are placed in such a way so that it can represent a clique. The encoding scheme is illustrated in figure 2. If any cell has 1 that is not part of the maximum clique, then the encoding is invalid. When a binary individual becomes invalid, then it is corrected using **repair** procedure. So here, number of 1 in a binary individual (array) is used as the fitness function in our QEA.

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

Figure 2: Proposed binary individual for MCP of the graph of figure 1.

B. Proposed QEA for MCP

We present the detailed algorithm of QEA for the Maximum Clique Problem.

Procedure QEA for the MCP

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

Here, length of Q-bit individual is m for MCP and it is the number of vertices. QEA maintains a population of Q-bit individuals, $Q(t) = \{q_1^t, q_2^t, q_3^t, \dots, q_k^t\}$ at generation t, where k is the size of population, and q_p^t is a Q-bit individual.

$$q_p^t = \begin{bmatrix} \alpha_{p_1}^t & \alpha_{p_2}^t & \alpha_{p_3}^t \dots \alpha_{p_m}^t \\ \beta_{p_1}^t & \beta_{p_2}^t & \beta_{p_3}^t \dots \beta_{p_m}^t \end{bmatrix} \quad (1)$$

Where, m is the number of Q-bits in an individual and $p = 1, 2, \dots, k$. In the step of initialize Q(t), all α_i^0 and β_i^0 are initialized with $1/\sqrt{2}$, where, $i = 1, 2, \dots, m$. On line 04 and 12, QEA produce population of binary individuals, $P(t) = \{x_1^t, x_2^t, \dots, x_k^t\}$ from population of Q-bit individuals, where t = 0, 1, 2, …. For notational simplicity, x and q are used instead of x_p^t and q_p^t respectively. The following **Make** procedure is used to obtain the binary string x.

Procedure Make (x)

01: **begin**
02: i ← 0
03: **while** (i < m) **do**
04: **begin**
05: i ← i + 1
06: **if** (random[0, 1) <$|\beta_i|^2$**)**
07: **then** $x_i \leftarrow 1$
08: **else** $x_i \leftarrow 0$
09: **end**
10: **end**

Binary individuals are repaired if needed. A possible problem is, a cell may have 1 which is not part of maximum clique. We are considering the graph of figure 1.

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

Figure 3: Invalid chromosome.

The chromosome shown in figure 3 is invalid. Invalid chromosomes are corrected by **Repair** procedure. We have deleted redundant 1s that are not part of maximum clique. A possible corrected version of invalid chromosome of figure 3 is shown in figure 4.

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

Figure 4: Corrected chromosome of figure 3.

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

Procedure Repair (x)

01: **begin**
02: **for (***i*=0 to *m***) do**
03: **begin**
04: **if** (x_i is 1) **then**
05: **begin**
06: **for (***j*=0 to *m***) do**
07: **begin**
08: **if (**x_j is 1**) then**
09: **begin**
10: **if (***i* is not equal *j* **and** x_i *is not adjacent to* x_j**) then**
11: **begin**
12: randomly choose *i* or *j*.
13: **if (***i* is choosen**) then**
14: $x_i \leftarrow 0$
15: **else**
16: $x_j \leftarrow 0$
17: **end**
18: **end**
19: **end**
20: **end**
21: **end**
22: **end**

After repairing the binary individual x, we have improved it by adding vertex. The improved binary individual may look like figure 5 which really represents the maximum clique of the graph of figure 1:

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

Figure 5: Improved chromosome of figure 4.

Here is the **Improve** procedure given bellow,

Procedure Improve (x)

01: **begin**
02: position ← ramdomly choose a position between 0 to *m*.
03: **for (***i*←position to *m***) do**
04: **begin**
05: **if (**x_i is 0**) then**
06: **if (***i* is connected to all vertices in the clique represented by x**) then**
07: $x_i \leftarrow 1$
08: **end**
09: **for (***i*←0 to position**) do**
10: **begin**
11: **if (**x_i is 0**) then**
12: **if (***i* is connected to all vertices in the clique represented by x**) then**
13: $x_i \leftarrow 1$
14: **end**
15: **end**

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

Procedure Update (q)

01: **begin**
02: $i \leftarrow 0$
03: **while** ($i < m$) **do**
04: **begin**
05: Determine $\Delta\theta_i$ with the lookup table
06: Obtain (α_i, β_i) from the following:
07: **if (q** is located in the first/third quadrant)
08: **then** $[\alpha_i' \ \beta_i']^T = H_\epsilon(\alpha_i, \beta_i, \Delta\theta_i)$
09: **else** $[\alpha_i' \ \beta_i']^T = H_\epsilon(\alpha_i, \beta_i, -\Delta\theta_i)$
10: **end**
11: $q \leftarrow q'$
12: **end**

Here, H_ϵ gate is used as a Q-gate to update a Q-bit individual ***q*** as a main variation operator. Q-bit of ***i***th position (α_i, β_i) is updated as follows:

$$[\alpha_i'' \ \beta_i'']^T = U(\Delta\theta_i)[\alpha_i \ \beta_i]^T :$$

Where, $U(\Delta\theta_i)$ is a simple rotation gate,

$$U(\Delta\theta_i) = \begin{bmatrix} cos(\Delta\theta_i) & -sin(\Delta\theta_i) \\ sin(\Delta\theta_i) & cos(\Delta\theta_i) \end{bmatrix}$$

if $|\alpha_i''|^2 \leq \epsilon \ and \ |\beta_i''|^2 \geq 1 - \epsilon$

then $[\alpha_i' \ \beta_i'] = \left[\sqrt{\epsilon} \ \sqrt{1-\epsilon}\right]^T$;

if $|\alpha_i''|^2 \geq 1 - \epsilon \ and \ |\beta_i''|^2 \leq \epsilon$

then $[\alpha_i' \ \beta_i'] = \left[\sqrt{1-\epsilon} \ \sqrt{\epsilon}\right]^T$;

Otherwise, $[\alpha_i' \ \beta_i'] = [\alpha_i'' \ \beta_i'']$;

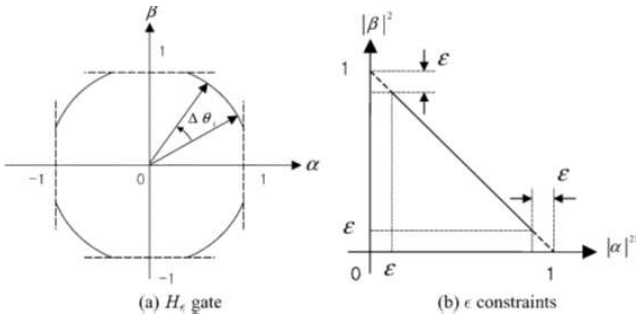

Figure 6: H_ϵ gate based on rotation gate.

In this QEA for MCP, the angle parameters used for the rotation gate are shown in table 1. Let us define an angle vector $\Theta = [\theta_1 \ \theta_2 \ ... \ \theta_8]^T$, where $\theta_1, \theta_2, ..., \theta_8$ can be found from table 1. We have used, $\theta_3 = 0.01\pi, \theta_5 = -0.01\pi$, and 0 for the rest. The values from 0.001π to 0.05π are recommended for the magnitude of $\Delta\theta_i$. Otherwise, it may converse prematurely. The sign of $\Delta\theta_i$ determines the direction of convergence. We have chosen $\epsilon = 0.01$. In table 1, $f(.)$ is the fitness, and x_i and b_i are the ith bits of the best solution x and the binary solution b, respectively. Here, $\theta_1 = 0, \theta_2 = 0, \theta_3 = 0.01\pi, \theta_4 = 0, \theta_5 = -0.01\pi, \theta_6 = 0, \theta_7 = 0, \theta_8 = 0$ are used.

Table 1: lookup table of $\Delta\theta_i$

x_i	b_i	$f(x) \geq f(b)$	$\Delta\theta_i$
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 20 of **Procedure QEA for MCP,** 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.

Here, is the overall structure [14] of Quantum-inspired evolutionary algorithm:

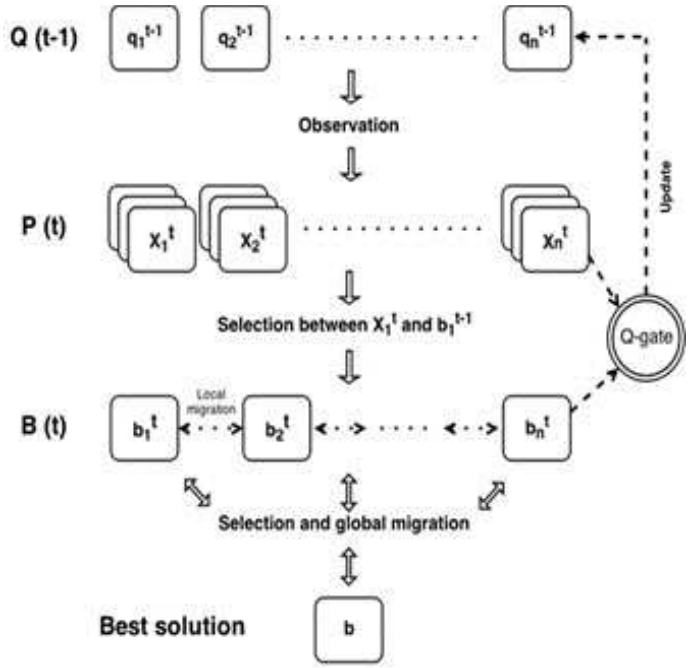

Figure 7: Structure of QEA [14].

IV. Experimental 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: 8 GB DDR3 1333MHz,

Operating System: Windows 7 64-bit

Datasets used to test our QEA for MCP are taken from Center for Discrete Mathematics and Theoretical Computer Science (DIMACS) benchmarking graph collection [11]. We have tested 40 datasets from [11]. The tested datasets are heterogeneous consisting of big graph like keller6 having 3361 vertices, highly dense graph like MANN_a81 with edges 5506380 and even simple graphs. Our results are tabulated and compared with other results in table 2.

Here, column QEA_MCP indicates our results. Green rows indicate the datasets for which QEA produces improved result. For keller6, we have got better result than all of the other works [3, 10, 12] mentioned here, eight other datasets have satisfactory results and for the remaining datasets our algorithm also produced optimal results.

In our experiments, we found that rotation probability of 0.65 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.

Table 2: Comparison of Obtained Results with Other Works.

Dataset	|V|	|E|	x(C)	QEA MCP	[12]	[3]	[10]
MANN_a27	378	70551	126	126	126	125	126
MANN_a81	3321	5506380	1100	1996	1996	1084	1098
p_hat500-1	500	31569	9	9	-	9	-
p_hat500-2	500	62946	36	36	-	36	-
p_hat700-1	700	60999	11	11	-	11	11
p_hat700-2	700	121728	44*	44	-	44	44
p_hat700-3	700	183010	62*	62	-	61	62
p_hat1000-1	1000	122253	10*	10	-	10	-
p_hat1000-2	1000	244799	46*	46	-	46	-
p_hat1000-3	1000	371746	68*	68	-	65	-
p_hat1500-1	1500	284923	12*	12	-	12	12
p_hat1500-2	1500	568960	65*	65		65	65
johnson8-2-4	28	210	4	4	-	4	-
johnson8-4-4	70	1855	14	14	-	14	-
johnson16-2-4	120	5460	8	8	-	8	-
c-fat200-1	200	1534	12	12	-	12	-
c-fat200-2	200	3235	24	24	-	24	-
c-fat200-5	200	8473	58	58	-	56	-
c-fat500-1	500	4459	14	14	-	14	-
c-fat500-2	500	9139	26	26	-	26	-
c-fat500-5	500	23191	64	64	-	54	-
c-fat500-10	500	46627	126	126	-	126	-
brock200_1	200	14834	21	21	-	21	-
brock200_2	200	9876	12	12	-	12	12
brock200_3	200	12048	15	15	-	15	-
brock200_4	200	13089	17	17	-	17	17
Brock400_1	400	59723	27	27	-	27	-
brock800_1	800	207505	23	21	-	23	-
brock800_2	800	208166	24	24	-	24	21
Hamming6-4	64	704	4	4	-	4	-
hamming8-2	256	31616	128	128	-	128	-
hamming8-4	256	20864	16	16	16	16	16
hamming10-4	1024	434176	40	40	38	40	40
gen400_p0.9_55	400	71820	55	55	-	55	55
gen400_p0.9_65	400	71820	65	65	-	65	65
gen400_p0.9_75	400	71820	75	75	-	75	75
keller4	171	9435	11	11	11	11	11
keller5	776	225990	27	27	27	27	27
keller6	3361	4619898	59*	57	49	54	56
DSJC500.5	500	125248	13*	13	-	13	13

Our algorithm is very fast because it can work with small population and also it needs less generation than other evolutionary algorithms to get the optimal solution. In our further experiment we have seen that population and generation are inversely proportional to each other. To find out this nature, we have conducted our experiment on dataset **gen400_p0.9_75** by varying population.

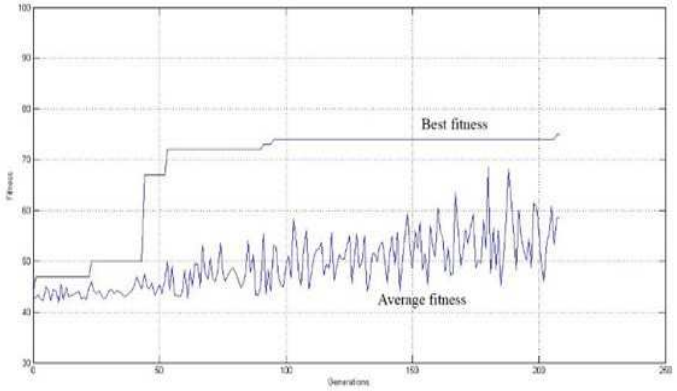

Figure 8: Average and best fitness for gen400_p0.9_75 dataset with population 5.

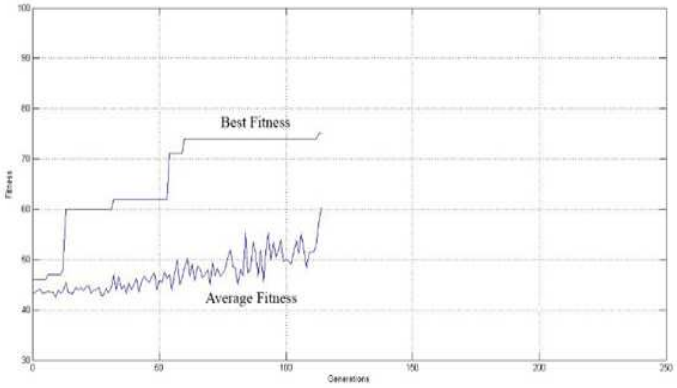

Figure 9: Average and best fitness for gen400_p0.9_75 dataset with population 10.

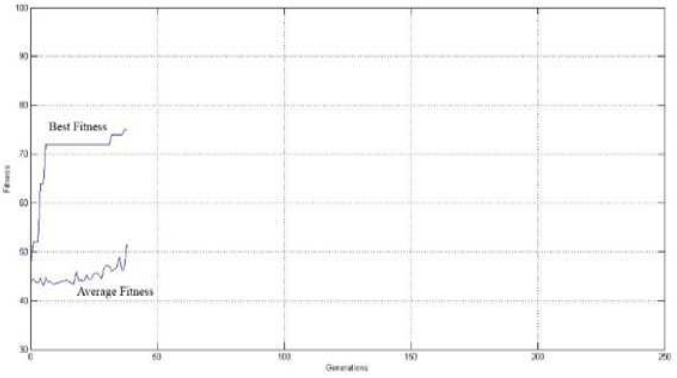

Figure 10: Average and best fitness for gen400_p0.9_75 dataset with population 30.

From figure 8, 9 and 10, we see that with increasing population, generation decreases. All of them produces same optimal result 75 for gen400_p0.9_75 dataset. But every time with increasing population size, we are getting this optimal result with less generations. So we can say, population and generation have inverse relationship to each other and it is also indicating the dynamicity of our algorithm.

V. CONCLUSION

In this paper, we proposed a Quantum-inspired Evolutionary Algorithm (QEA) for Maximum Clique Problem (MCP). The main variation operator of our QEA is the 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 they are corrected to obtain the valid solution. After certain generations, all or partial population are replaced with best binary individual to avoid local optimization. Unlike the previous techniques, our QEA finds the optimal result in a single run reducing the total time signi ficantly. We experiment with 40 datasets from [11]. For 37 datasets, we found expected results as stated in [11] and for 8 datasets our QEA produces better result than reported DIMACS results. So we can say these are the major achievements of our algorithm. 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 algorithms.

REFERENCES

[1] P. M. Pardalos and J. Xue. "The Maximum Clique Problem," Journal of Global Optimization, 4, 1994, pp. 301-328.

[2] Bo Huang, "Finding Maximum Clique with a Genetic Algorithm," The Pennsylvania State University The Graduate School Capital College.

[3] S. Balaji, V. Swaminathan, K. Kannan. "A Simple Algorithm for Maximum Clique and Matching Protein Structures", © International Journal of Combinatorial Optimization Problems and Informatics, Vol. 1, No. 2, pp. 2-11, Sep-Dec 2010. ISSN: 2007-1558.

[4] 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.

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

[6] David R. Wood, "An Algorithm for Finding a Maximum Clique in a Graph", Operations Research Letters, Vol. 21, pp. 211-217.

[7] James Cheng, Yiping Ke, "Fast Algorithms for Maximal Clique Enumeration with Limited Memory," KDD '12 Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 1240-1248. ACM 978-1-4503-1462-6, 2012.

[8] Kengo Katayama, Akihiro Hamamoto, Hiroyuki Narihisa, "An Effective Local Search for the Maximum Clique Problem," Information Processing Letters, Vol. 95, Issue 5, pp. 503-511, 15 September 2005.

[9] Harsh Bhasin, Rohan Mahajan, (2012), "Genetic Algorithms Based Solution To Maximum Clique Problem", International Journal on Computer Science and Engineering (IJCSE), Vol. 4, No.18, pp. 1443-1448.

[10] Qingfu Zhang; Jianyong Sun; Tsang, E., "An evolutionary algorithm with guided mutation for the maximum clique problem," Evolutionary Computation, IEEE Transactions on, vol.9, no.2, pp.192,200, April 2005, doi: 10.1109/TEVC.2004.840835.

[11] "DIMACS benchmarks", [online] Available in Internet: cs.hbg.psu.edu/benchmarks/clique.html, Apr. 28, 2015.

[12] Roger Ouch, Kristopher W.Reese, Roman V.Yampolskiy, "Hybrid genetic algorithm for the maximum clique problem combining sharing and migration", Association of the Advancement of Artificial Intelligence.

[13] Pronaya Prosun Das and Mozammel H. A. Khan, "Quantum-Inspired Evolutionary Algorithm to Solve Graph Coloring Problem" in Proceeding of the Intl. Conf. on Advances In Computing, Electronics and Electrical Technology - CEET 2014, pp.21-25.

[14] Pronaya Prosun Das and Mozammel H. A. Khan, "An Effective Quantum-inspired Evolutionary Algorithm for Finding Degree-constrained Minimum Spanning Tree," International Journal of Computer Science and Electronics Engineering (IJCSEE), Volume 2, Issue 4 (2014), pp.202-207, ISSN 2320–4028.