An Effective Quantum-inspired Evolutionary Algorithm for Finding Degree-constrained Minimum Spanning Tree

Pronaya Prosun Das, and Mozammel H. A. Khan

***Abstract*—**The Degree-constrained Minimum Spanning Tree (d-MST) is an extended adaptation of general Minimum Spanning Tree (MST) problem. This problem is pertinent in the design of communication networks. It consists of finding a spanning tree whose nodes do not exceed a given maximum degree and whose total edge length is minimum. In this formulation the problem turns into NP-hard, therefore meta heuristic approaches like Ant Colony Optimization, Simulated Annealing, Evolutionary Algorithms etc. are suitable for solving d-MST problem. In this paper, we demonstrate a Quantum-inspired Evolutionary Algorithm (QEA) that solves instances of the problem to optimality. We have used one dimensional array of Q-bits called Q-bit individuals to produce binary individuals. Then binary individuals are repaired according to problem specification. Here, Q-gate is the main variation operator applied on Q-bits. For testing our algorithm, three types of benchmark sets are used. Experimental results show that the algorithm has performed very well and it has also outperformed current best results.

***Keywords*—**Degree-constrained Minimum Spanning Tree, Quantum-inspired Evolutionary Algorithm, NP Hard Problems, Q-bit representation.

I. Introduction

Degree constrained minimum spanning tree is a real world variants of well-known MST problem. This concept is very important in designing networks for everything from computer and telephone communications to transportation, sewage and plumbing [1], [2]. As we know general MST is solvable in polynomial time. In real networks, however, the vertices (or nodes) are normally subject to a degree-constraint. In that case degree-constrained minimum spanning tree (d-MST) of a graph cannot be solvable with a polynomial time algorithm. So it becomes an NP-hard problem [3], [4], [5]. Therefore, Meta heuristics are often used to find good solutions in a reasonable amount of time.

In previous research, several different approaches to the d-MST problem have been proposed such as approximation algorithms [6], branch and bound algorithms, simulated annealing [7], evolutionary algorithms, iterative search techniques, multi-start hill-climbing etc.

In this paper, we propose a Quantum-inspired Evolutionary Algorithm (QEA) [8], [9], [10], which is the combination of concept of quantum computing [11] 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 prior works. The Methodology of the research is described in section III. In section IV, the experimental results are described. Finally the conclusion and discussion is outlined in section V.

Pronaya Prosun Das is with the Department of Computer Science and Engineering, East West University, Dhaka, Bangladesh (phone: 8801720581238, e-mail: pronaya.prosun@gmail.com).

Mozammel H A Khan is with the Department of Computer Science and Engineering, East West University, Dhaka, Bangladesh (e-mail: mhakhan@ewubd.edu).

II. Prior Works

One of the recent works on d-MST is an Ant-based algorithm for finding degree-constrained minimum spanning tree [12]. Artificial ants are used to identify a set of candidate edges from which a degree-constrained spanning tree has been constructed. Another work on d-MST using Genetic Algorithm by Zhou and Gen [13] where they used prufer codes to encode spanning tree. As well as paper [14] also used the same technique. Knowles and Cornes [15] did a different approach, storing a spanning tree in an $n\times(d\text{-}1)$ array where n is the number of vertex and d is the degree constrain. G. R. Raidl introduced a weighted coding in his paper [16] where he approached the problem using Genetic Algorithm. In 1994 Piggott & Suraweera [17] used bit string of size of edges $|E|$ to represent the solution tree. But according to S. B. Kamal [14] there are high chances that this encoding contains invalid spanning tree. In all the above mentioned papers no one used a binary encoding of edge to represent the chromosomes except the paper [17].

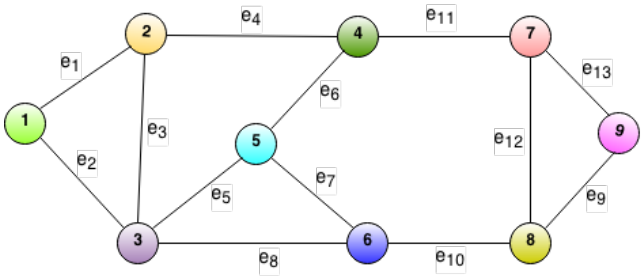
III. Methodology of the Research

A. Representation

We have used Q-bit based on the concept of qubits of quantum computer in our QEA [11]. 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.

Let a given graph be $G = (V, E)$. Here, $V = \{v_1, v_2, v_3, ..., v_n\}$ is a finite set of vertices and $E = \{e_1, e_2, e_3, ..., e_m\}$ is finite set of edges where n and m are the number of vertices and edges respectively. For d-MST problem, we have used one-dimensional array of Q-bits as a Q-bit individual, where each index corresponds to an edge. Later binary individuals are produced from Q-bit individuals. If the ith edge is a part of a d-MST, then the ith element of the array is 1 otherwise it is 0.

Thus, in a valid individual, 1s are placed in such a way so that it can represent a d-MST. The encoding scheme is illustrated in Figure 1. If any cell has 1 that is not part of the d-MST, then



e_1	e_2	e_3	e_4	e_5	e_6	e_7	e_8	e_9	e_{10}	e_{11}	e_{12}	e_{13}
1	0	0	**1**	**1**	0	0	**1**	**1**	**1**	0	**1**	**1**

Fig. 1 A Graph and its binary edge encoding for spanning tree

the encoding is invalid. Invalid binary individuals are corrected using **repair** procedure. The Summations of the costs of the selected edges are used as the fitness function in our QEA.

B. Proposed QEA for d-MST

Here we present the detailed algorithm of QEA for Degree-constrain Minimum Spanning Tree Problem.

Procedure QEA for the d-MST
01: **begin**
02: $t \leftarrow 0$
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: $t \leftarrow t + 1$
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 while**
20: **end**

Here length of Q-bit individual is m for d-MST problem, where m is the number of edges. 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,

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

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 11, 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.

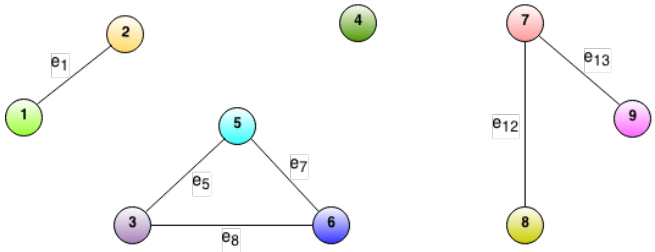
Procedure make (x)
01: **begin**
02: $i \leftarrow 0$
03: **while** (i < m) **do**
04: **begin**
05: $i \leftarrow i + 1$
06: **if (**random [0, 1) <$|\beta_i|^2$**)**
07: **then** $x_i \leftarrow 1$
08: **else** $x_i \leftarrow 0$
09: **end while**
10: **end**

At each generation it may be needed to repair the binary individuals. A cell may have 1 which is not part of d-MST. For a valid d-MST three conditions have to be maintained.

1. It must contain all the vertices of given graph *G*.
2. It must be connected and graph contains no cycle.
3. Degree of each vertex must be less than or equal to the given constrain *d*.

If any of the condition is violated, then it is repaired using **repair procedure**.

The chromosome shown in Figure 2 is invalid. We have deleted redundant edges (1s) that are not part of d-MST according to the conditions stated earlier. A possible corrected version of invalid individual of Figure 2 is shown in Figure 3.



e_1	e_2	e_3	e_4	e_5	e_6	e_7	e_8	e_9	e_{10}	e_{11}	e_{12}	e_{13}
1	0	0	0	**1**	0	**1**	**1**	0	0	0	**1**	**1**

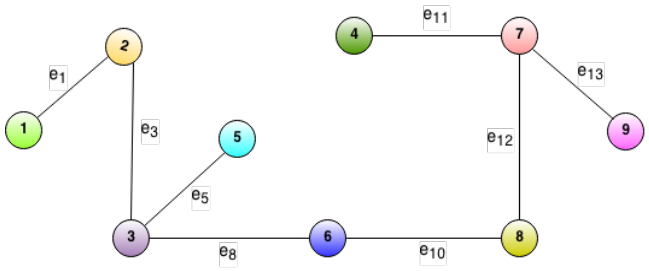
Fig. 2 Invalid binary individual before correction

Procedure repair(x)
01: **begin**
02: **for** each selected edge e in binary individual x according to edge cost in increasing order
03: **begin**
04: **if (**e makes a cycle**)**
05: **then** exclude e from chromosome
06: **else**
07: **if (**degree of u and v of edge e within constrain d**)**
08: **then** keep it
09: **else** exclude e from chromosome
10: **end if**
11: **end if**

12: **end for**
13: **if (**chromosome **x** has less than *n*-1 edge**)**
14: **then** construct a valid tree
15: **end if**
16: **end**

In every generation Q-bit individuals are updated by comparing binary individuals with best individual. It will be discussed later. The **update** procedure of Q-bits is presented below:

Procedure update (q)



e1	e2	e3	e4	e5	e6	e7	e8	e9	e10	e11	e12	e13
1	0	1	0	1	0	0	1	0	1	1	1	1

Fig. 3 Possible valid individual after correction

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 while**
11: $q \leftarrow q'$
12: **end**

Here H_ϵ gate [9] is used as a Q-gate to update a Q-bit individual q as a main variation operator. Q-bit of *i*th position $(\alpha_i, \ \beta_i)$ 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,

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

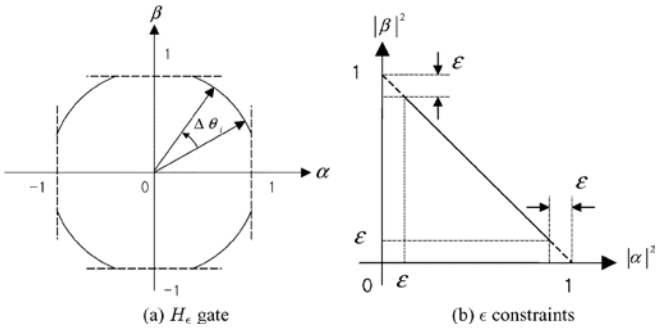


Fig. 4 $\boldsymbol{H_\epsilon}$ gate based on rotation gate

In this QEA for d-MST, the angle parameters used for the rotation gate are shown in Table I. Let us define an angle vector, $\Theta = [\theta_1 \theta_2 \ ... \ \theta_8]^T$, where $\theta_1, \theta_2, \ldots, \theta_8$ can be selected easily by intuitive reasoning.

If $x_{i,j}$ and $b_{i,j}$ are 0 and 1, respectively, and if the condition $f(x) \geq f(b)$ is false:

1. if the Q-bit is located in the first or the third quadrant (see Figure 4), θ_3, the value of $\Delta\theta_{i,j}$ is set to a positive value to increase the probability of the state $|1\rangle$;
2. if the Q-bit is located in the second or the fourth quadrant, $-\theta_3$ should be used to increase the probability of the state $|1\rangle$;

If $x_{i,j}$ and $b_{i,j}$ are 1 and 0, respectively, and if the condition $f(\boldsymbol{x}) \geq f(\boldsymbol{b})$ is false:

1. if the Q-bit is located in the first or the third quadrant, θ_5, is set to a negative value to increase the probability of the state $|0\rangle$;
2. if the Q-bit is located in the second or the fourth quadrant, $-\theta_5$ should be used to increase the probability of the state $|0\rangle$;

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 I, $f(.)$ is the fitness, and x_i and b_i are the *i*th 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.

In line 18 of **Procedure QEA for d-MST,** 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

TABLE I
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

by the best individual, while global migration is implemented by replacing all the solution by the best individual.

Here, overall structure of Quantum-inspired Evolutionary Algorithm has been shown in figure 5.

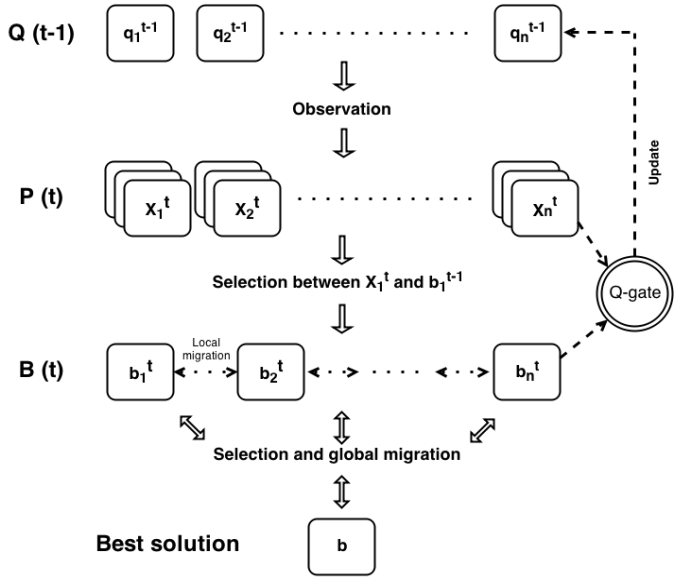


Fig. 5 Structure of QEA

IV. Experimental Results

A. Data set

We have tested our algorithm on SHRD, STR and SYM datasets [18] consists of 100 instances ranging from 15 to 100 vertices with the degree constraints of 2 through 5. The SYM data set was obtained from A. Volgenant. STR and SHRD datasets were created by A. Ernst and M. Krishnamoorthy. For SYM, STR, and SHRD data sets there is a list of the best known solutions for them from previous algorithms. Some of them have proven results. For each instance, we compared our result with the best previously published results. It should be noted that the data sets are divided into two categories, Euclidean and non-Euclidean graphs. Two types of Euclidean graphs were tested: SYM and STR graphs. The points in the SYM graphs come from higher dimensional space. As for non-Euclidean graphs, SHRD dataset is tested. It is created by assigning non-Euclidean costs such that numbers of optimal solutions are limited.

B. 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

All sets of instances have their previously published results. Most instances have their own optimal proven solution found by exact algorithms. For the case of proven solutions, our algorithm also found exactly the same results and for other solutions, QEA produced better results than the current best results.

Table II, III, IV show the comparison between our algorithm's results and current best results.

TABLE II
COMPARISON DATA FOR STRUCTURED HARD (SHRD) GRAPHS

No.	Dataset	*n*	*d*	Current Best	Our results	Gain (%)
1	shrd150	15	2	895	895	0.00
2	shrd150	15	3	582*	582	0.00
3	shrd150	15	4	430	430	0.00
4	shrd150	15	5	339	339	0.00
5	shrd159	15	2	906	904	0.22
6	shrd159	15	3	597	597	0.00
7	shrd159	15	4	430	430	0.00
8	shrd159	15	5	332	332	0.00
9	shrd200	20	2	1873	1679	11.55
10	shrd200	20	3	1100	1088	1.10
11	shrd200	20	4	829	802	3.37
12	shrd200	20	5	638	627	1.75
13	shrd209	20	2	1808	1699	6.42
14	shrd209	20	3	1106	1092	1.28
15	shrd209	20	4	807	799	1.00
16	shrd209	20	5	634	629	0.79
17	shrd258	25	2	2953	2704	9.21
18	shrd258	25	3	1838	1745	5.33
19	shrd258	25	4	1302	1277	1.96
20	shrd258	25	5	1007	999	0.80
21	shrd259	25	2	2984	2714	9.95
22	shrd259	25	3	1870	1756	6.49
23	shrd259	25	4	1312	1292	1.55
24	shrd259	25	5	1019	1016	0.30
25	shrd300	30	2	4560	3992	14.23
26	shrd300	30	3	2738	2592	5.63
27	shrd300	30	4	1965	1905	3.15
28	shrd300	30	5	1526	1504	1.46
29	shrd309	30	2	4268	3991	6.94
30	shrd309	30	3	2765	2585	6.96
31	shrd309	30	4	1947	1898	2.58
32	shrd309	30	5	1487	1474	0.88

For all of the tables, *n* is the number of vertices and *d* is the degree constraint. "Current Best" is the best result denoted on the benchmark [18] for each graph type and "Our Results" shows the result produced by our algorithm. Gain (%) shows the difference between the Current Best results and our results as a percentage and "*****" indicates the value next to it is an optimal proven result.

In our experiments, we found that rotation probability of 0.6 performs better for almost all datasets and average gain for SHRD, STR and SYM datasets are 3.28%, 0.38% and 5.10%

TABLE III
COMPARISON DATA FOR STRUCTURED (STR) GRAPHS

No.	Dataset	*n*	*d*	Current Best	Our results	Gain (%)
1	str1000	100	2	5211	5000	4.22
2	str1000	100	3	4702*	4702	0.00
3	str1000	100	4	4546*	4546	0.00
4	str1000	100	5	4403*	4403	0.00
5	str1002	100	2	7276	7109	2.35
6	str1002	100	3	6713*	6713	0.00
7	str1002	100	4	6511*	6511	0.00
8	str1002	100	5	6362*	6362	0.00
9	str1004	100	2	8935	8702	2.68
10	str1004	100	3	8313*	8313	0.00
11	str1004	100	4	8117*	8117	0.00
12	str1004	100	5	7930*	7930	0.00
13	str1006	100	2	10684	10551	1.26
14	str1006	100	3	10155*	10155	0.00
15	str1006	100	4	9951*	9951	0.00
16	str1006	100	5	9756*	9756	0.00
17	str1008	100	2	12625	12436	1.52
18	str1008	100	3	11952*	11952	0.00
19	str1008	100	4	11726	11726	0.00
20	str1008	100	5	11530	11530	0.00
21	str709	70	2	11347	11195	1.36
22	str709	70	3	10765*	10765	0.00
23	str709	70	4	10561	10561	0.00
24	str709	70	5	10367	10367	0.00
25	str705	70	2	8134	8087	0.58
26	str705	70	3	7688*	7688	0.00
27	str705	70	4	7496*	7496	0.00
28	str705	70	5	7311*	7311	0.00
29	str700	70	2	4727	4708	0.40
30	str700	70	3	4397*	4397	0.00
31	str700	70	4	4245*	4245	0.00
32	str700	70	5	4100*	4100	0.00
33	str508	50	2	10859	10789	0.65
34	str508	50	3	10358*	10358	0.00
35	str508	50	4	10138	10138	0.00
36	str508	50	5	9941*	9941	0.00
37	str504	50	2	7682	7632	0.66
38	str504	50	3	7300*	7300	0.00
39	str504	50	4	7105*	7105	0.00
40	str504	50	5	6919*	6919	0.00
41	str500	50	2	4471	4420	1.15
42	str500	50	3	4128*	4128	0.00
43	str500	50	4	3962*	3962	0.00
44	str500	50	5	3807*	3807	0.00

respectively. 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

TABLE IV
COMPARISON DATA FOR SYMMETRIC (SYM) GRAPHS

No.	Dataset	*n*	*d*	Current Best	Our results	Gain (%)
1	sym302	30	2	2048	1983	3.28
2	sym302	30	3	1311 *	1311	0.00
3	sym302	30	4	1256 *	1256	0.00
4	sym302	30	5	1252 *	1252	0.00
5	sym308	30	2	2273	2241	1.43
6	sym308	30	3	1546 *	1546	0.00
7	sym308	30	4	1506 *	1506	0.00
8	sym308	30	5	1506 *	1506	0.00
9	sym500	50	2	2522	1775	42.08
10	sym500	50	3	1156 *	1156	0.00
11	sym500	50	4	1105 *	1105	0.00
12	sym500	50	5	1098 *	1098	0.00
13	sym508	50	2	2161	1888	14.46
14	sym508	50	3	1281 *	1281	0.00
15	sym508	50	4	1266 *	1266	0.00
16	sym508	50	5	1266 *	1266	0.00
17	sym700	70	2	2746	2175	26.25
18	sym700	70	3	1341 *	1341	0.00
19	sym700	70	4	1234 *	1234	0.00
20	sym700	70	5	1183 *	1183	0.00
21	sym708	70	2	2472	1834	34.79
22	sym708	70	3	1318 *	1318	0.00
23	sym708	70	4	1284 *	1284	0.00
24	sym708	70	5	1271 *	1271	0.00

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 generation than other evolutionary algorithms to get the optimal solution. We have used population size within 20 to 100 for different datasets. In our further experiment we have also seen that population and generation has inverse relationship to each other.

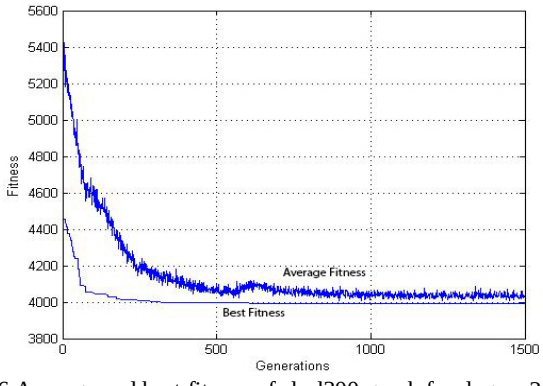


Fig. 6 Average and best fitness of shrd300 graph for degree 2 with gain 14.23%

V. Conclusion and Discussion

In this paper, we proposed a Quantum-inspired Evolutionary Algorithm (QEA) for Degree-constrain

Minimum Spanning Tree Problem (d-MST). 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. 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.

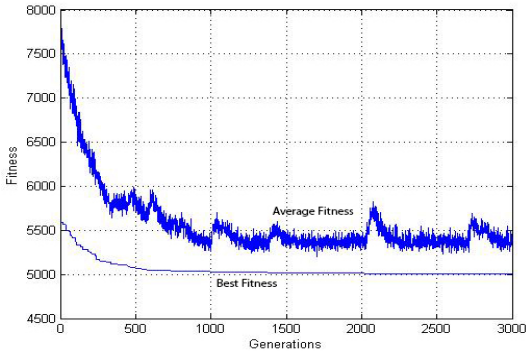


Fig. 7 Average and best fitness of str1000 graph for degree 2 with gain 4.22%

Unlike the previous techniques, our QEA finds the optimal result in a single run reducing the total time significantly. We have experimented with 3 datasets consisting 100 instances from [18]. For 41 instances, we have found better results than current best results and for other instances, they have their proven results. Our algorithm also produced the same results to those proven instances. So we can say these are the major achievements of our algorithm over the previous works. In 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.

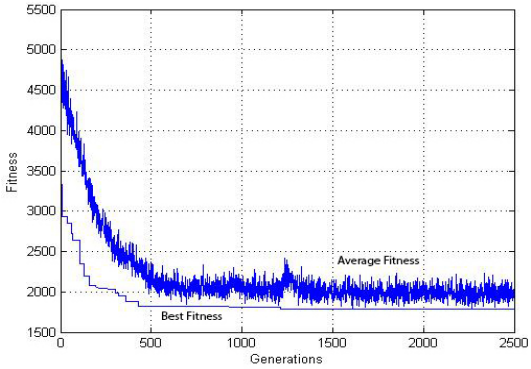


Fig. 8 Average and best fitness of sym500 graph for degree 2 with gain 42.08%

REFERENCES

[1] Narula, S. C. and C. A. Ho, "Degree Constrained Minimum Spanning Tree", Computers and Operations Research, 7, 1980, pp. 239–249.

[2] M. Savelsbergh and A. Volgenant. "Edge exchanges in the degree-constrained minimum spanning tree problem", Computers and Operations Research, 12(4), 1985, pp. 341–348.

[3] B. Boldon, N. Deo, and N. Kumar, "Minimum-Weight Degree-constrained Spanning Tree Problem: Heuristics and Implementation on an SIMD Parallel Machine", Parallel Computing (22): 369-382, 1996.

[4] B. M. E. Moret and H. D. Shapiro. "An empirical assessment of algorithms for constructing a minimum spanning tree". In N. Dean and G. E. Shannon, editors, Computational Support for Discrete Mathematics, DIMACS Workshop, March 12-14, 1992. DIMACS Series in Discrete Mathematics and Theoretical Computer Science, V. 15, pp. 99-117, 1994.

[5] D. J. Rosenkrantz, R. E. Stearns, and P. M. Lewis II. "An analysis of several heuristics for the traveling salesman problem", SIAM Journal on Computing, 6(3):563-581, 1977.

[6] R. Ravi, M. V. Marathe, S. S. Ravi, D. J. Rosenkrantz, and H. B. Hunt III, "Many birds with one stone: Multi-objective approximation algorithms," Proceedings of the 25th Annual ACM Symposium on the Theory of Computing (STOC'93), May 1993, pp. 438–447.

[7] P. J. M. van Laarhoven and E. H. L. Aarts, Simulated Annealing: Theory and Applications, Dordrecht, Netherlands: Reidel, 1987.

[8] 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.

[9] Kuk-Hyun Han; Jong-Hwan Kim, "Quantum-inspired evolutionary algorithms with a new termination criterion, Hε gate, and two-phase scheme," Evolutionary Computation, IEEE Transactions on , vol.8, no.2, pp.156,169, April 2004.

[10] 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.

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

[12] Thang N. Bui, Catherine M. Zrncic, "An ant-based algorithm for finding degree-constrained minimum spanning tree", DOI: 10.1145/1143997.1144000 Conference: Genetic and Evolutionary Computation Conference, GECCO 2006, Proceedings, Seattle, Washington, USA, July 8-12, 2006.

[13] Zhou, G.; Gen, M.; Tianzu Wu, "A new approach to the degree-constrained minimum spanning tree problem using genetic algorithm," Systems, Man, and Cybernetics, 1996., IEEE International Conference on , vol.4, no., pp.2683,2688 vol.4, 14-17 Oct 1996.

[14] Bin Kamal, S.; Kibria, M.G., "A comparison of two stochastic iterative search techniques for Degree Constrained Minimum Spanning Tree," Informatics, Electronics & Vision (ICIEV), 2012 International Conference on , vol., no., pp.847,852, 18-19 May 2012.

[15] Knowles, J. and D. Corne. "A New Evolutionary Approach to the Degree-Constrained Minimum Spanning Tree Problem," IEEE Trans. On Evolutionary Computation, 4(2). 2000, pp125-134.

[16] Günther R. Raidl and Bryant A. Julstrom, "A weighted coding in a genetic algorithm for degree-constrained minimum spanning tree problem" in Proceeding of ACM Symposium on Applied Computing, 2000, pp.440-445.

[17] Piggot, P., and Suraweera, F. (1994). Encoding Graphs for Genetic Algorithms; An investigation using the Minimum Spanning Tree Problem. Evo Workshops'94, 305-314.

[18] "DIMACS benchmarks", [online] Available: http://cs.hbg.psu.edu/txn131/spanning_tree.html

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, Bangladesh and also working as a teaching assistance there. 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.