Journal of Network and Innovative Computing.
ISSN 2160-2174 Volume 6 (2018) pp. 007–015

A Model for Multi-processor Task Scheduling Problem using Quantum Genetic Algorithm

Rashika Bangroo[1], Kushal Gupta[2] and Reya Sharma[3]

[1]DIT University,Dehradun,Uttrakhand,India,
rashi1121993@gmail.com

[2]DIT University,Dehradun,Uttrakhand,India,
kushal.loveiit@gmail.com

[3]Shri Mata Vaishno Devi University,J&K,India

Abstract: **Multiprocessor task scheduling problem is a well-known NP-hard and an important problem in the field of parallel computing. In order to solve this problem optimally, researchers have applied various heuristics and meta-heuristics. However, Genetic Algorithm (GA) is one of the widely opted meta-heuristic approaches to solve combinatorial optimization problems. In order to increase the probability of finding an optimal solution in GA, a new approach known as Quantum Genetic Algorithm (QGA) has been adopted. QGA increases the speed and efficiency of computation of a conventional GA by introducing the concept of parallelism of quantum computing in GA. In this paper, Quantum behavior inspired GA is introduced to solve multiprocessor task scheduling problem. The proposed QGA has been modified at certain points with some new operators to make it compatible for the same problem. The performance of proposed QGA is verified on two standard problems of linear algebra i.e., Gauss Jordan Elimination (GJE) and LU decomposition. The results have been compared with the state of the arts to prove its effectiveness.**

Keywords: Multi-processor DAG scheduling problem,Quantum Genetic Algorithm,Gauss Jordan Elimination,LU Decomposition

I. Introduction

In order to solve multi-processor task scheduling problem researchers have utilized the benefits of various heuristics and meta-heuristics techniques.There are many such heuristics availaible in literature.In 2008,Jin Shiyuan presents comparative study of different heuristics on multi-processor task scheduling problem [16];Min-Min heuristic by Ibarra O and kim C [2];A* search techinque by Kcafil et al [3];Simulated annealing by Eliasi R,Elperin T [4];highest level first with estimated times(HLFET)by Adam T and Chandy K [5];insertion scheduling heuristic(ISH)by Kruatrachue [6, 7] and genetic algorithm by Hou E and Ansari N [8]. Task scheduling problem refers to the scheduling of n tasks among the m processors with certain precedence and communication constraints in order to increase the system performance. Task scheduling thus plays an effective role in efficient utilisation of resources. The tasks are represented in the form of a DAG(Directed Acyclic Graph) as $G(V, E)$ where V denotes the set of all nodes and E denotes the set of all edges [9].The Genetic algorithm is based on the theory of natural selection and works on generating a set of random solutions and making them compete in an arena where only the fittest survive. Each solution in the set is equivalent to a chromosome. In order to make the solution more optimal a new version of the genetic algorithm i.e, the quantum genetic algorithm has been used which combines the features of both quantum computing as well as GA. QGA has previously been applied to several optimization problems like knapsack problem, travelling salesman problem [10], etc and has found to increase the speed and efficiency of the conventional GA. QGA provides several other advantages over the other meta-heuristic techniques such as excellent global search ability and the performance of the algorithm over several problems is not affected by small population size. Since there is no such published work on scheduling of tasks on multiprocessor systems using quantum genetic algorithm,this paper thus proposes a model for Multiprocessor task scheduling problem using Quantum inspired genetic algorithm in order to minimize the make-span time. Further, it makes a comparative study of the results obtained with the other meta-heuristic techniques.The next section briefs about the concepts related to Multiprocessor DAG scheduling problem. The section 3 describes the principles of Quantum Genetic algorithm. It also briefs about its related work.The section 4, explains the proposed methodology. The experimental set up and performance parameters are been given in section 5. Finally, the concluding remarks and acknowledgement are given in section 6 and 7 respectively.

II. Multiprocessor DAG scheduling problem

The rapid advancement in the number and size of optimisation problems is motivating to develop the parallel CPU archietecture. In order to solve such optimization problems researchers have utilized the benefits of quantum genetic algorithm. One such optimization problem is the multiprocessor task scheduling problem which involves the scheduling of N tasks among the M processors with certain precedence

and communication constraints [11, 9].Task scheduling thus plays an effective role in efficient utilisation of resources. The tasks are represented in the form of a DAG(Directed Acyclic Graph) as G(V, E) where V denotes the set of all nodes and E denotes the set of all edges. An edge(n_j,n_k)∈

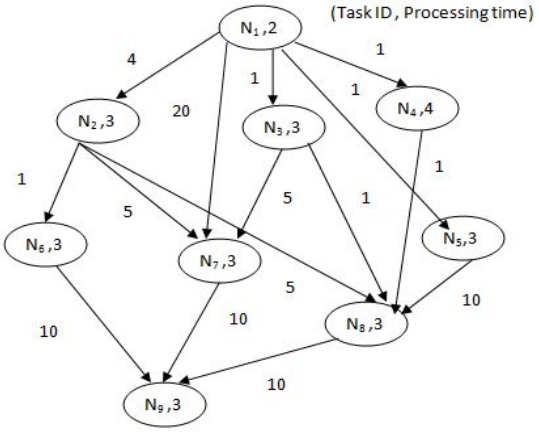

Figure. 1: Directed Acyclic Graph

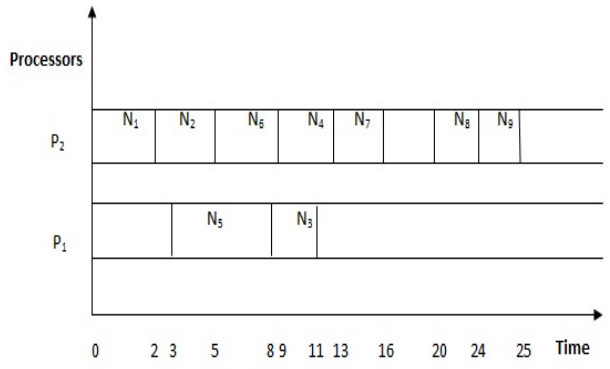

Figure. 2: A Feasible Schedule for Fig 1

E denotes communication and precedence between the tasks n_j and n_k. Each node is represented by N$(n_j, ptime)$ where n_j denotes the task id and ptime represents the processing time of the task n_j.The precedence (n_j,n_k) implies that the task n_k cannot start its execution prior to task n_j .Thus, our objective is to assign N tasks to M homogeneous processors using quantum genetic algorithm so that the makespan of DAG is minimum. Figure 1 describes a DAG with 9 nodes(tasks).Each node is associated with a node number and its processing time.The weights on the edges determine the communication delay.The communication cost becomes zero if the two tasks are assigned to the same processor. Figure 2 depicts a feasible schedule on two homogeneous processors corresponding to DAG in figure 1 which gives a total minimum makespan of 25 time units.

III. Quantum Genetic Algorithm

In the past few years there has been tremendous increase in the use of quantum computing for solving various NP-hard optimization problems. The quantum computing has its roots derived from the Laws of quantum mechanics. The quantum based algorithms use superposition of the states as compared to classical techniques which is more efficient for obtaining the solution. Much research has not been done in these domains of quantum computing but still it emerges as a field of high interest.Quantum computing is an emerging field in computer science which is being derived from the concepts of Quantum Physics. Quantum genetic algorithms are based solely on the concept of quantum superposition state and quantum bits [12]. In quantum computers, information is stored in the form of qubits. Qubit is a double quanta system which acts as a physical media in order to store the desired information. A qubit can be in any state $|0>$, $|1>$ or a superposition of these two states. The state $|0>$ and $|1>$ are known as the spin-down and spin-up states of the qubit respectively. A qubit contains information about both the spin-up and spin-down states. The state of a qubit is defined as follows:

$$|\phi>=\alpha|0>+\beta|1> \tag{1}$$

where α and β are complex numbers known as the probability amplitude of the corresponding qubit state. The α and β must satisfy the equation :

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

A. Qubit Encoding

Genetic algorithms using binary representation use bits for representing each gene of a chromosome. In case of QGA each gene of a chromosome is represented by a quantum bit [13]. In order to derive a classic gene from a quantum gene,it is adequate to select the quantum bit randomly by using the values of α and β respectively . A Qubit is usually defined in terms of its probability amplitude as $\begin{bmatrix}\alpha\\ \beta\end{bmatrix}$ and for n parameters the multi-qubit encoding is defined as follows :

$$\begin{bmatrix}\alpha_1 & \alpha_2 & \alpha_3 & . & . & . & \alpha_n\\ \beta_1 & \beta_2 & \beta_3 & . & . & . & \beta_n\end{bmatrix} \tag{3}$$

where n denotes the number of genes in chromosome.

B. Quantum Rotating Gates

Quantum genetic algorithm uses probability amplitude of qubits in order to encode the chromosome and quantum rotating gates to realize the update operation of the chromosome. When quantum rotating gates are used to realise the update operation of chromosomes, the offsprings being generated is not determined by the parents since the chromosomes are in superposition state [14]. But instead is being determined by both the optimal solution from the parent population as well as the probability amplitude of each state. The main difference between the QGA and GA lies in the application of quantum rotating gates in order to change the values of the probability amplitude. Thus quantum rotating gates is the main operation of QGA which mainly affects performance and is usually defined as

$$\cup(\theta_i)=\begin{bmatrix}\cos\theta_i & -\sin\theta_i\\ \sin\theta_i & \cos\theta_i\end{bmatrix} \tag{4}$$

and the updated values of chromosomes is given by

$$\begin{bmatrix} \alpha_i^{'} \\ \beta_i^{'} \end{bmatrix} = \cup(\theta_i) \begin{bmatrix} \alpha_i \\ \beta_i \end{bmatrix} = \begin{bmatrix} \cos\theta_i & -\sin\theta_i \\ \sin\theta_i & \cos\theta_i \end{bmatrix} \begin{bmatrix} \alpha_i \\ \beta_i \end{bmatrix} \quad (5)$$

where $\begin{bmatrix} \alpha_i \\ \beta_i \end{bmatrix}$ and $\begin{bmatrix} \alpha_i^{'} \\ \beta_i^{'} \end{bmatrix}$ denotes the probability amplitudes of i^{th} qubit of chromosome before and after application of the quantum rotating gates respectively and θ_i denotes the rotating angle.

IV. Proposed Model

In this work we have considered the multiprocessor task scheduling problem using the quantum genetic algorithm. It focuses on scheduling of tasks in order to minimize the makespan of the tasks. Makespan may be defined as the time required by all the tasks in a given schedule to execute. The flowchart of QGA is as shown in Fig 3.

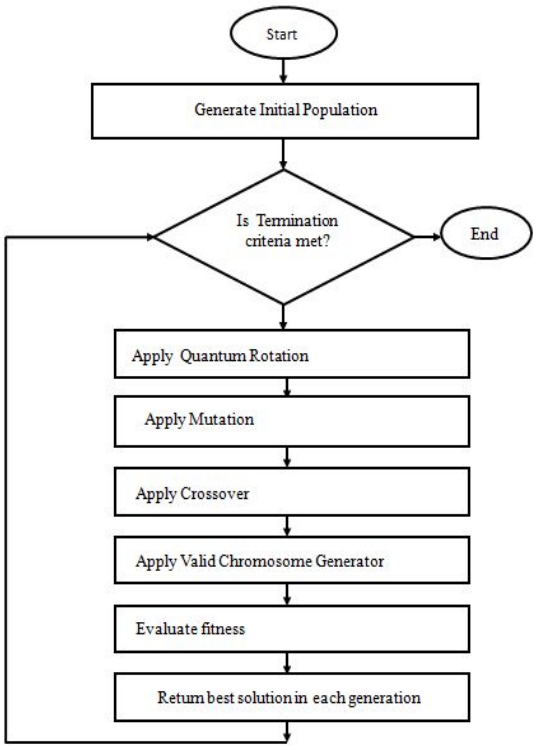

Figure. 3: Flowchart of QGA

1) Fitness Function

The fitness function can be defined as a function which takes the initial parameters as input and provides the best possible output for the given input values. The fitness function should be efficient enough to provide a practically feasible solution by using least resources. The fitness of individuals is being calculated repeatedly by using the fitness function. Thus the fitness function should be capable enough to provide faster results.The fitness function depends on the domain of the problem. The main aim of the work presented here is to optimize the scheduling length of tasks on multiprocessors which is being represented by DAG. Thus the solution can be obtained by makespan which provides the complete scheduling length of the tasks as defined below :

$$makespan = max\left(FT\left(n\right)_{i,j}\right) \quad (6)$$

where $FT\left(n\right)_{i,j}$ denotes the execution time of task n_i on the best suitable processor p_j and 1≤ i ≤ N, 1 ≤ j ≤ M. The fitness function used here is based on serial schedule scheme (SSS).[**?**]The SSS scheme basically schedules all the tasks based sequentially until a feasible solution is obtained. The SSS scheme is used when the chromosomes or solution generated is invalid.

2) Generation of a valid Chromosome

In order to ensure that a valid chromosome is being generated, the use of valid chromosome generator (VCG) is essential. Sometimes it happen that the updated chromosome may violate certain precedence constraints, thus the main purpose of a VCG is to convert an invalid chromosome to a valid one. The VCG thus keeps track of only the activities which result in invalid chromosome generation and thus swaps these suspected activities. Initially the indegree of each activity is being calculated . Those indegrees having zero values are being stored in ϕ_i . Then the out degree of each solution belonging to ϕ_i is being calculated . Now all the solutions are being checked whether they belong to ϕ_i or not. If the out degree of each individual is different and does not belong to ϕ_i , then the maximum out degree is being calculated and then swapped with the activity. Conversly, if all the out degrees belong to ϕ_i, then no such swapping is being performed.

A. Improvements in Quantum Genetic Algorithm

There are certain disadvantages of basic quantum genetic algorithm. The basic QGA uses only the fitness function for evaluating the solution, which causes the genes of some chromosomes which have larger fitness value to spread swiftly in the solution which in turn leads to premature loss of diversity and thus the solution always falls under the local optimal solution. Also the conventional QGA is good for global search ability but is still inadequate for local search performance which results in slow convergence in the later stages and thus do not converge towards the global optimal solution.

1) Self Adaptive Rotation Angle

The conventional QGA uses a fixed angle for the application of rotating gates operator. This paper uses the self adaptive rotation angle strategy.In this strategy, the value of the rotation angle is adjusted dynamically which is based on the evolutionary process. An adjustment strategy is being used for comparing the fitness of current value, g(x) of individual ${q^t}_j$ with the fitness of current optimal chromosome,g$(best_i)$. In other words, if g(x)¿g$(best)$ then adjust the qubits of ${q^t}_j$ so that its probability amplitude(α_i,β_i) converges towards the direction which is propitious for g(x) . Conversly, if g(x)¡g$(best)$,then adjust the qubits of ${q^t}_j$ so that its

probability amplitude(α_i ,β_i) converges towards the direction which is propitious for g($best$). The value of the rotating angle is adjusted according to the formula given below:

$$\theta_i = \theta_{max} - \left(\frac{\theta_{max} - \theta_{min}}{it_{max}}\right) * iter \qquad (7)$$

where iter denotes current iteration and it_{max} denotes the total number of iterations. The proper selection of angle of rotation is essential in order to find optimal solution in less number of iterations.It also helps to maintain a proper balance between the local and global optimal values. Therefore the value of angle of rotation is modified at each iteration using the self adaptive rotation angle approach.

2) Mutation and Crossover Operator

Mutation operator is being added to the conventional quantum genetic algorithm after the application of rotation gates strategy. It helps to deviate some chromosomes slightly from the current evolutionary direction and thus prevents the solution to fall under the local optimum solution. This operator swaps the values of probability amplitudes (α,β) of the chromosomes , which totally reverses the chromosome′s evolutionary direction. It also helps to enhance the local search capability of conventional QGA and thus prevents the loss of important data in the chromosome. It also helps to enhance the diversity in the population and reduces the chances of premature convergence. The quantum mutation is as shown in Figure 4. Cross over operator helps to diversify the so-

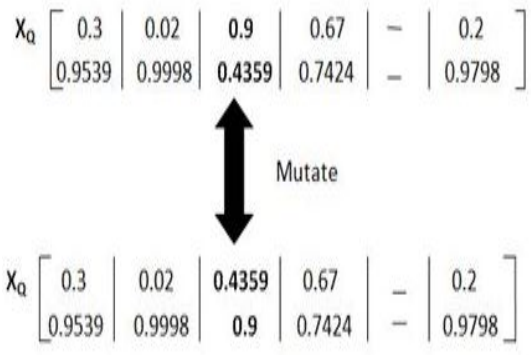

Figure. 4: Quantum Mutation Operator

lution. There are various methods of using cross over operation in literature such as one-point crossover, multi-point crossover and uniform crossover. In this work we have used the multi-point crossover which is the same as we use for genetic algorithms. The cross over operator used in this work is a two break point cross over operator. In this process two parents are being selected for mating. The two parents are being combined with certain mutation rate which is defined as the ratio of total off-springs generated to the total population. Two randomly generated break points are being selected. Then the values of α and β of the two selected parents lying in between the randomly selected break points are being interchanged and the rest of the values are kept same in order to produce two new off-springs. This is depicted in the Figure 5.

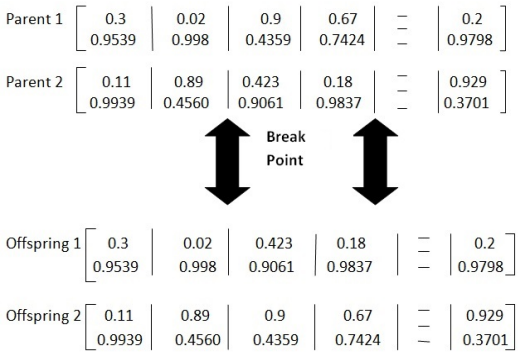

Figure. 5: Crossover Operator

B. Algorithm

The various steps involved in QGA are as follows :

1. *At t=0, Generate initial population $P_q(t)$ = {$p_1(t)$, $p_2(t)$,$p_3(t)$,.....,$p_n(t)$} randomly using equation(3) by converting Q-bit representation into chromosome and calculate fitness*
2. *Sort the initial population in descending order based on their fitness values*
3. *Choose the best initial solution using selection operator by replacing bottom $1/5^{th}$ individuals by top $1/5^{th}$ individuals of the population*
4. *Until the termination condition is satisfied do steps 5-12*
5. *t=t+1, Perform rotation of Quantum gates and update Pq(t) by Pq(t+1).*
6. *Perform Mutation operation with certain mutation probability*
7. *Perform Crossover operation*
8. *Apply valid chromosome generator(VCG)*
9. *Add the generated population to the initial best solution*
10. *Sort the newly generated population in descending order of their fitness values using the fitness function*
11. *Choose the best solution using selection operator by replacing bottom $1/5^{th}$ of the generated population by best $1/5^{th}$ generated population*
12. *Store the best solution in each generation*

The algorithm can be described as follows. Initially all input parameters such as number of tasks, task size, communication delay, precedence constraints, etc are provided as input. Then the initial population is generated randomly using equation(3) by initializing the values of α and β in the chromosome as $1/\sqrt{2}$ and $1/\sqrt{2}$ respectively. Then the fitness function is being used to calculate the best solution in the population using equation(6). The chromosomes are then sorted according to their fitness values in descending order. The best initial population is being selected by replacing $1/5^{th}$

of the generated population by best $1/5^{th}$ generated population. Increment the value of t by 1 and apply quantum rotation gates strategy using equation(5).The value of angle of rotation i.e, θ_i is tuned by initialising the value of θ_{min} and θ_{max} as 0.01*π and 0.05*π respectively.The proper selection of angle of rotation is essential in order to find optimal solution in less number of iterations. Apply quantum mutation and crossover to maintain the diversity in the solution. Now again calculate the fitness of the population generation after application of mutation operator. Apply valid chromosome generator(VCG) to ensure that valid chromosomes are being generated. Sort the chromosomes in decreasing order of their fitness values. Select the best initial population by replacing $1/5^{th}$ of the generated population by best $1/5^{th}$ generated population. Repeat the algorithm until the termination condition is reached which is the number of chromosomes in this case.

V. Experimental Evaluation

There are no benchmarks available in Literature for studying the performance of scheduling problems [15]. Researchers usually use the random graph generation for analyzing the performance. The DAG is generated using two standard problems of linear algebra that is Gauss Jorden Elimination(GJE) and LU decomposition. [16] The proposed QGA algorithm for multi-processor task scheduling problem is evaluated using the Gauss Jorden Elimination(GJE) and LU decomposition method of DAG generation[17].Fig 6 and 7 depicts the graphical representation of GJE and LU decomposition problem respectively. The experimental details for

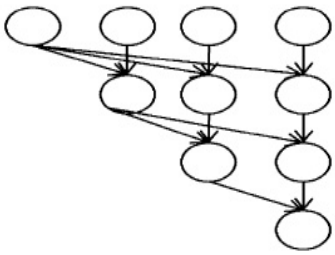

Figure. 6: Pictorial Representation of GJE graph

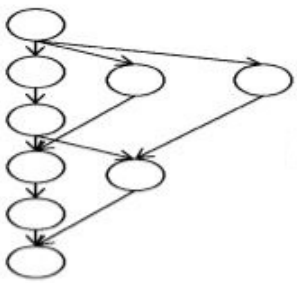

Figure. 7: Pictorial Representation of LU decomposition graph

the GJE task graph is as shown in table 1. All the parameters for evaluation are taken from available meta-heuristics in order to ensure fair comparison [16, 9]. The first row depicts the number of tasks (N) which are to be scheduled on the multi-processors.The second row depicts the processing time of tasks which is 40/task. The third row defines the communication delay involved which is 100 per edge. As shown in row 4 and 5, the number of processors used and the number of QGA iterations used are 4 and 10 respectively. The proposed model begins with initialisation of quantum bit updating vector with random values between 1 to n. Also, the number of chromosomes initially generated are 1000.

Table 1: GJE task graph experimental details

Number of Tasks(N)	15	21	28	36
Processing Time	40/task	40/task	40/task	40/task
Communication Cost	100/edge	100/edge	100/edge	100/edge
Processors(M)	4	4	4	4
QGA Iterations	10	10	10	10

Table 2: LU decomposition task graphs experimental details

Number of Tasks(N)	14	20	27	35
Processing Time	10s bottom layer task plus 10s for every layer			
Communication Cost	80/edge	80/edge	80/edge	80/edge
Processors(M)	4	4	4	4
QGA Iterations	10	10	10	10

The comparison of QGA with other meta-heuristics using GJE task graph and LU decomposition method is shown graphically in Fig 8 and Fig 10 respectively.
From Fig 8 and 10,it is observed that makespan obtained using QGA by varying the problem size using values given in Table 1 and Table 2 outperforms all the other meta-heuristics. Thus the graph proves the effectiveness of QGA against other techniques.The results also show that the GA and QGA are most promising heuristics than other counterparts.Thus,for further evaluation we consider these two techniques only.Fig 9 shows the comparative study of GA and QGA by varying the number of processors using GJE task graph of 465 nodes.Fig 11 depicts the comparative study of varied processors on LU graph of 464 nodes.From Fig 9 and 11,it is clear that when number of processors are varied with respect to the size of the GJE and LU graphs respectively,the QGA performs better than GA.Hence QGA outperforms all other meta-heuristics and hence is a promising solution for multi-processor task scheduling problem.

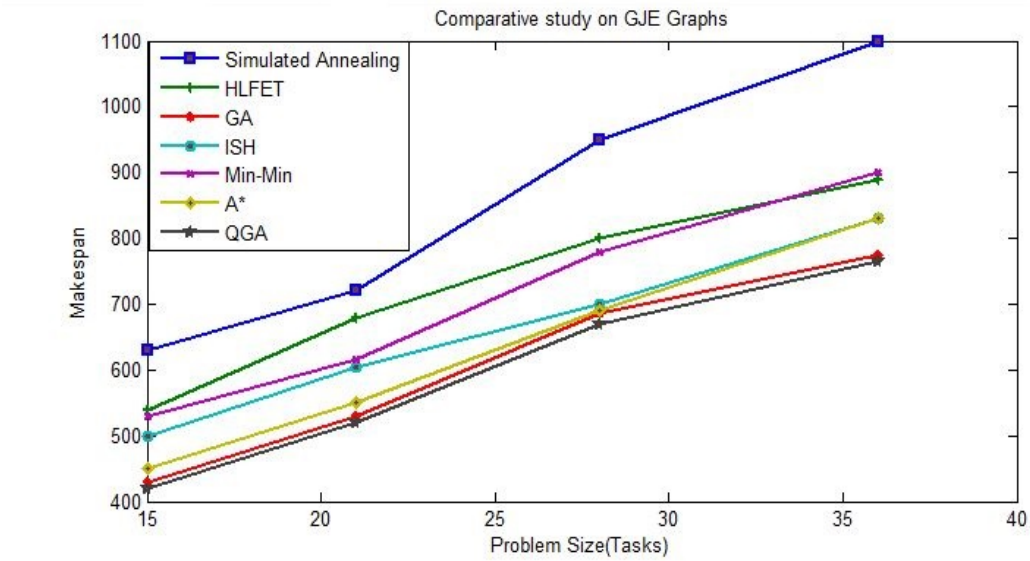

Figure. 8: Comparative study of 7 heuristics on GJE graph

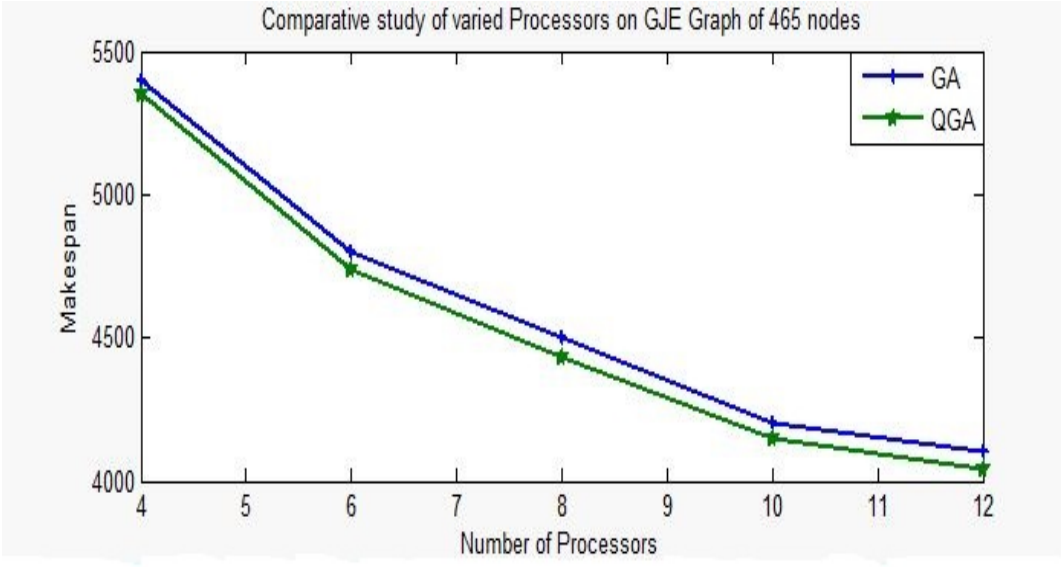

Figure. 9: Comparative study of two meta-heuristics using GJE taskgraph-465 with varied number of processors

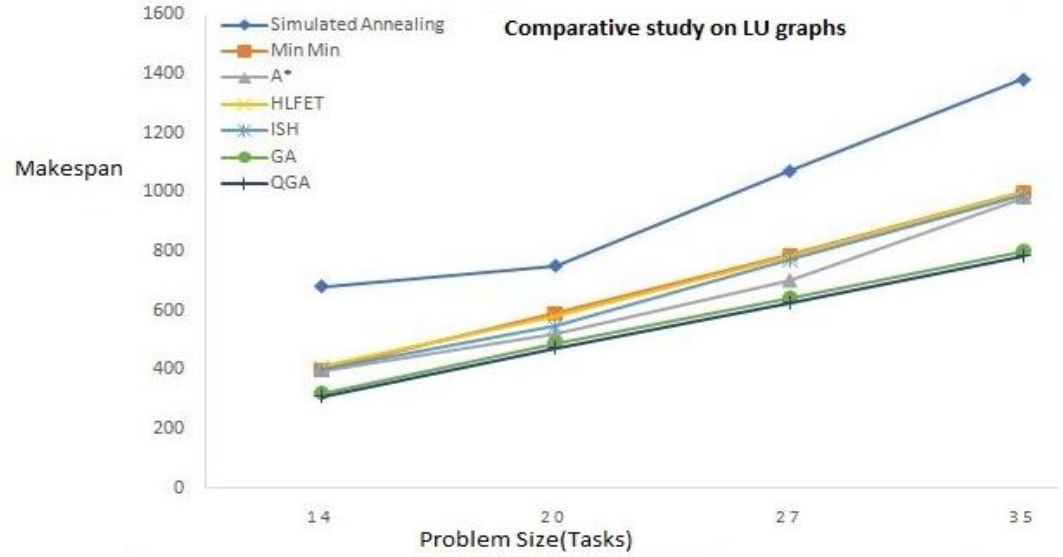

Figure. 10: Comparative study of 7 heuristics on LU task graph

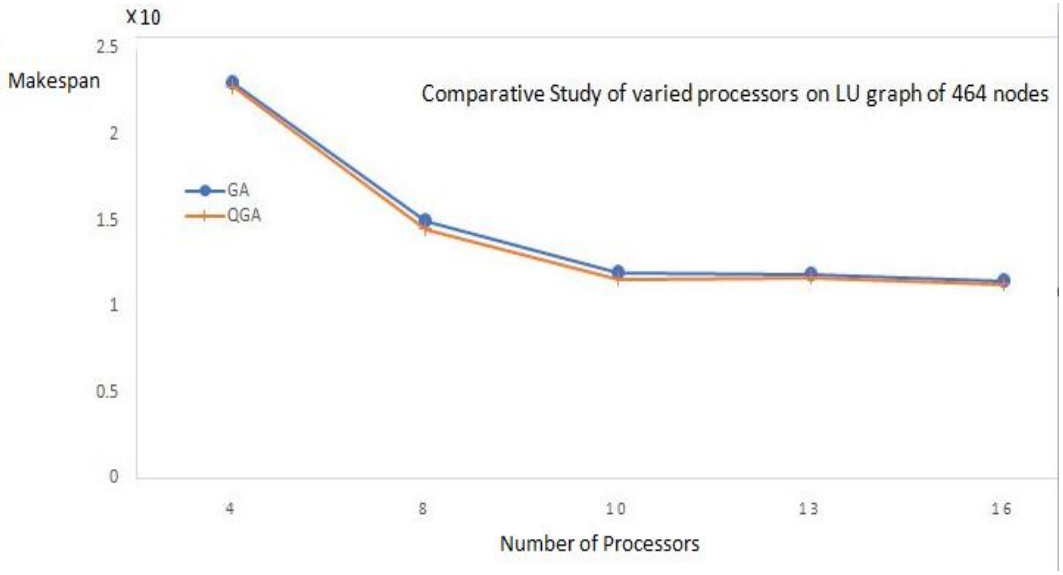

Figure. 11: Comparative Study of varied Processors on LU Graph of 464 Nodes

VI. Conclusion

This paper has discussed the multiprocessor task scheduling problem using Quantum Genetic Algorithm. The proposed QGA has been modified at certain points with some new operators to make it compatible for the same problem. The self adaptive rotation angle strategy is being used for varying the rotation angle instead of using a constant value of around 0.01π as used in conventional QGA. Other operators such as mutation and crossover are also used in order to diversify the solution and converge it towards the global optimum value. Further,the results obtained by applying quantum genetic algorithm on task scheduling problem are being compared with those obtained from other meta-heuristics and the Q-GA is found to be more effective than its counterparts. The performance of proposed QGA is verified on two standard problems of linear algebra i.e., Gauss Jordan Elimination (G-JE) and LU decomposition.Thus, it is concluded that the hybrid meta-heuristic, QGA provides an effective solution for the multiprocessor task scheduling problem and thus can also be used to solve other large and complex combinatorial optimization problems effectively.In the future work,the QGA can be used on other DAG or scientific applications.

Acknowledgments

We would like to show our gratitute to Professor Ajith Abraham,Director Machine Intelligence Research Labs (MIR Labs),Washington,USA for his comments that greatly improved the manuscript and we also thank our 3 anonymous reviewers for their insights.

References

[1] Jin, Shiyuan, Guy Schiavone, and Damla Turgut. "A performance study of multiprocessor task scheduling algorithms." The Journal of Supercomputing 43.1 (2008): 77-97.

[2] Ibarra, Oscar H., and Chul E. Kim. "Heuristic algorithms for scheduling independent tasks on nonidentical processors." Journal of the ACM (JACM) 24.2 (1977): 280-289.

[3] Kafil, Muhammad, and Ishfaq Ahmad. "Optimal task assignment in heterogeneous computing systems." Heterogeneous Computing Workshop, 1997.(HCW'97) Proceedings., Sixth. IEEE, 1997.

[4] Eliasi, R., T. Elperin, and A. Bar-Cohen. "Monte Carlo thermal optimization of populated printed circuit board." IEEE transactions on components, hybrids, and manufacturing technology 13.4 (1990): 953-960.

[5] Adam, Thomas L., K. Mani Chandy, and J. R. Dickson. "A comparison of list schedules for parallel processing systems." Communications of the ACM 17.12 (1974): 685-690.

[6] Kruatrachue, B., and T. G. Lewis. "Duplication scheduling heuristics (dsh): A new precedence task scheduler for parallel processor systems." Oregon State University, Corvallis, OR (1987).

[7] Kruatrachue, Boontee, and Ted Lewis. "Grain size determination for parallel processing." IEEE software 5.1 (1988): 23-32.

[8] Hou, Edwin SH, Nirwan Ansari, and Hong Ren. "A genetic algorithm for multiprocessor scheduling." IEEE Transactions on Parallel and Distributed systems 5.2 (1994): 113-120.

[9] Kumar, Neetesh, and Deo Prakash Vidyarthi. "A novel hybrid PSOGA meta-heuristic for scheduling of DAG with communication on multiprocessor systems." Engineering with Computers 32.1 (2016): 35-47.

[10] Talbi, Hichem, Amer Draa, and Mohamed Batouche. "A new quantum-inspired genetic algorithm for solving the travelling salesman problem." Industrial Technology, 2004. IEEE ICIT'04. 2004 IEEE International Conference on. Vol. 3. IEEE, 2004.

[11] Hwang, Reakook, Mitsuo Gen, and Hiroshi Katayama. "A comparison of multiprocessor task scheduling algorithms with communication costs." Computers & Operations Research 35.3 (2008): 976-993.

[12] Han, Kuk-Hyun, and Jong-Hwan Kim. "Quantum-inspired evolutionary algorithm for a class of combinatorial optimization." IEEE transactions on evolutionary computation 6.6 (2002): 580-593.

[13] Lahoz-Beltra, Rafael. "Quantum genetic algorithms for computer scientists." Computers 5.4 (2016): 24.

[14] Zhang, Gexiang, Weidong Jin, and Laizhao Hu. "A novel parallel quantum genetic algorithm." Parallel and Distributed Computing, Applications and Technologies, 2003. PDCAT'2003. Proceedings of the Fourth International Conference on. IEEE, 2003.

[15] Kwok, Y-K., and Ishfaq Ahmad. "Benchmarking the task graph scheduling algorithms." Parallel Processing Symposium, 1998. IPPS/SPDP 1998. Proceedings of the First Merged International... and Symposium on Parallel and Distributed Processing 1998. IEEE, 1998.

[16] Jin, Shiyuan, Guy Schiavone, and Damla Turgut. "A performance study of multiprocessor task scheduling algorithms." The Journal of Supercomputing 43.1 (2008): 77-97.

[17] Gerasoulis, Apostolos, and Tao Yang. "Performance bounds for column-block partitioning of parallel Gaussian elimination and Gauss-Jordan methods." Applied numerical mathematics 16.1-2 (1994): 283-297.

Author Biographies

Rashika Bangroo is currently working as Assistant Professor in the department of Computer Science and Engineering at DIT University,Dehradun.She holds a Masters degree in Computer Science and Engineering from Shri Mata Vaishno Devi University and has published some research papers in scopus indexed conferences.Her research interests include

parallel computing,data mining and quantum genetic algorithms.

Kushal Gupta is currently working as Assistant Professor in the department of Computer Science and Engineering at DIT University,Dehradun.He has obtained Masters degree in Computer Science and Engineering from Motilal Nehru National Institute of Technology,Allahabad.He is also pursuing his Phd. degree in Security and Wireless Networks from Uttrakhand Technical University.His research interests include Data structure and Algorithms.

Reya Sharma is currently pursuing Phd.in Computer Science and Engineering from Shri Mata Vaishno Devi University.Her research interests include Particle Swarm Optimisation, Machine learning and Artificial intelligence.