Proc. of the Intl. Conf. on Advances In Computer and Electronics Technology- ACET 2014.

ISBN: 978-1-63248-024-8 doi: 10.15224/ 978-1-63248-024-8-06

Variant of Genetic Algorithm and its applications

[Chanchal Kumar, Shiv Prakash, Tarun Kumar Gupta and Dinesh Prasad Sahu]

For very long time, finding solutions to problems like Travelling Salesmen Problem, Multiple Knapsack Problems, scheduling in heterogeneous system etc. were considered very tedious task. The solution to such problems was time consuming and requires high end computations. Genetic Algorithm (GA) emerged as an efficient solution and considered as a successful tool to solve such problems for many years. With the passage of time, researchers find GA traps near optimal solution. Therefore, the concept of Quantum computing is merged with GA and known as Quantum Genetic Algorithm (QGA). The evaluation of the performance of proposed model is done over scheduling in distributed computing.

***Keywords*— GA, GA and QGA Operators, QC, QGA, DC, Scheduling**

I. Introduction

In the field of research, many computational techniques and models were inspired from natural phenomenon or from deep observations of living systems. During early 1960's, evolutionary computation technique originated and leading researchers, industrialist, and scientists were influenced by its wide range of applicability to solve combinatorial optimization problems on a huge search space with a efficient time/cost. Particle Swarm Optimization (PSO) technique is inspired by the behavior of the bird flocking. Darwin's theory over reproduction in nature helped to develop Genetic Algorithm (GA). All these computational methodologies fall in the category of soft computing [1, 22].

Quantum theory explained complicated behaviors of atomic structure. It provided a uniform framework to construct many modern physical concepts and considered as the one of greatest scientific achievement of 20th century.

Chanchal Kumar
Independent Researcher
India

Shiv Prakash
Jawaharlal Nehru University
India

Tarun Kumar Gupta
Jawaharlal Nehru University
India

Dinesh Prasad Sahu
Jawaharlal Nehru University
India

The concept of quantum computing was first proposed by Feynman in early 80's.Researchers are working in the direction of constructing a realistic quantum computer. Present day computers can use binary bits i.e. either it can be 0 or 1 to represent data. On the other hand, quantum computer would use the concept of quantum bits or 'qubits'. Qubits can be either zero or one or superposition of both. This concept has been borrowed from concept of 'quirks' of quantum mechanics. This 'quirks' give quantum computers a huge edge in performing particular type of calculations. As the Quantum Computing research area includes concepts like quantum computers and quantum algorithms. Therefore, some quantum algorithms had been introduced in the literature [3,4]. During last decade some work has been reported in combining Quantum computing concept with nature inspired evolutionary computation [5, 6, 7, 11]. The quantum evolutionary algorithm showed better performance than evolutionary algorithm such as genetic algorithm. Ying had tried to explain the interplay between quantum theory and artificial intelligence. The researchers from all over the world are invited to fill up the gaps in this direction [2].

II. Quantum Computing

In quantum computing, the smallest unit of information is 'Qubit', which may be in the 1 state or 0 states or in the superposition of two states. In Quantum mechanics "State" means all the aspects together, whatever information one can have for the system should be contained in the description of the state. If we say that we know the state of the system at any instant that means we know everything about the system at that instant that is possible to know. For Quantum systems of a single particle, the information about the position, at time t, is given by $\Psi(x,t)$. At a given time t, there may not be a definite position of the particle as the particle may be found at various positions with probabilities proportional to $|\Psi(x,t)|^2$. But to know anything about its position at a time t we must know its wave function $\Psi(x,t)$ at a given instant of time t.

III. Problem Structure

Many optimization problems like Multiple Knapsack problem and Scheduling in heterogeneous system can't be solved by mathematical methods (linear/nonlinear programming).This paper focus on the basic structure of such problems and why the mathematical methods are not capable to find a suitable solution.

Proc. of the Intl. Conf. on Advances In Computer and Electronics Technology- ACET 2014.

ISBN: 978-1-63248-024-8 doi: 10.15224/ 978-1-63248-024-8-06

A. *Why Mathematical Techniques Failed*

There should be no doubt that mathematical methods were very successful to find solutions of small real world problems. Research is also moving in these fields for providing solutions to such complex mathematical problem. But theoretically it has its own limitations. Although the mathematical methods have the ability to find the solution in a finite number of steps and can be employed at reasonable cost for larger problems but it can't be used due to oversimplifications of non linearity towards linearity that throws away the optimal solutions at a large distance. Designing of an EA Methods and Scope

The search space of most real world applications is defined by a set of objects e.g. processors, number of machines, capacity, reliability of each nodes, consumption of energy etc. The evolutionary operators often work on abstract mathematical objects like binary string in the case of genetic operators. Genetic operators are cross over, mutation and inversion. The crossover explores search space while the mutation and inversion exploits search space. By using these operators, job will be assigned in such a way that make span should be minimum. Depending upon the nature of problem, initially the structure of chromosome is decided. After that the population is randomly create depending on its size. Then the fitness of chromosome is calculated. Out of various chromosomes, some chromosomes are selected as per selection process. EA operators are applied on these selected chromosomes. After a number of generations, the near optimum solution is obtained.

Each one of them has their own benefits and drawbacks. On one side in the first approach empirical and theoretical results are available for the standard instances of evolutionary algorithms.

IV. Proposed Quantum Genetic Algorithm

We have elaborated QGA in the following subsections.

A. *Genetic Algorithm (GA)*

Genetic algorithms (GA) [8, 9] are inspired by Darwinian's theory of natural selection and evolution. The simple genetic algorithm initiate with a population of randomly generated chromosomes, all chromosome are eligible to be a solution to the concrete problem but the best one is selected as per selection policy. Further three natural operators selection or reproduction, crossover and mutation are applied on chromosomes. Over the period of time, the population evolves through successive iteration process of controlled and competition variation. Every state of population is known as generation.

B. *Quantum Inspired -GA*

QGA introduced the idea of quantum computing with genetic algorithms. The concept of superposition, entanglements and intervention are taken from QC while selection, crossover and mutation are taken from GA. The Q bit population is used encode the chromosomes while quantum rotation and quantum mutation are used for updating the solution [12, 13, 14, 15]. A Q bit solution is m Q-bit string can be defined as follows:

$$\begin{bmatrix} \alpha_1(t) & \alpha_2(t) & \dots & \alpha_m(t) \\ \beta_1(t) & \beta_2(t) & \dots & \beta_m(t) \end{bmatrix}, \quad \text{where} |\alpha_i|^2 + |\beta_i|^2 = 1, \quad i = 1,2,.....m \tag{1}$$

Initial Population & Q bit Encoding

Initial population is randomly generated as:

$$p_Q(t) = \{p_1(t), p_2(t), p_3(t).....p_4(t)\} \tag{2}$$

where $p_i(t)$ is defined as follows:

$$p_j(t) = \begin{bmatrix} \alpha_1(t) & \alpha_2(t) & \dots & \alpha_m(t) \\ \beta_1(t) & \beta_2(t) & \dots & \beta_m(t) \end{bmatrix} \tag{3}$$

Quantum Rotation Operations

Quantum rotation gate is used for updating solution. The Quantum Rotation Gate $U(\Delta\theta)$ for a chromosome of length unto 2 had been defined in the literature as:

$$U(\Delta\theta) = \begin{bmatrix} \cos(\Delta\theta) & -\sin(\Delta\theta) \\ \sin(\Delta\theta) & \cos(\Delta\theta) \end{bmatrix} \tag{4}$$

The magnitude and direction of rotation angle $\Delta\theta$ are crucial. Method of obtaining $\Delta\theta$ directly impacts the convergence speed and algorithm's searching efficiency. The phase of chromosomes is moved by quantum rotation operation for each Q bit by $\Delta\theta$ to moves toward global optimal chromosomes. By studying the phase relation between these Q bits rotated population can be determined as follows:

$$\begin{bmatrix} \alpha(t+1) \\ \beta(t+1) \end{bmatrix} = U(\Delta\theta) \begin{bmatrix} \alpha(t) \\ \beta(t) \end{bmatrix} \tag{5}$$

Quantum Mutation Operator

To increase the searching ability in local space, the diversity of individual is increased and the probability of immature

Proc. of the Intl. Conf. on Advances In Computer and Electronics Technology- ACET 2014.

ISBN: 978-1-63248-024-8 doi: 10.15224/ 978-1-63248-024-8-06

convergence is reduced through quantum mutation operator. In general quantum 'Not' operation is used widely for mutation operation in quantum inspired evolutionary techniques. In each generation, a random number is generated in the range of 0 and 1. If the mutation probability Pm, is higher than this random number then chromosome is selected randomly and the positions of two quality parameter will be swapped. For example, if the randomly chosen chromosome is $[\alpha(t) \quad \beta(t)]^T$ and two randomly chosen quality parameter is first and the last then after mutation it will be converted according to following equation

$$\begin{bmatrix} \alpha'(t) \\ \beta'(t) \end{bmatrix} = \begin{bmatrix} \beta(t) \\ \alpha(t) \end{bmatrix} \qquad (6)$$

Proposed Algorithm

1. At t=0, Initialize population
   $P_Q(t) = \{ p_1(t), p_2(t), \ldots, p_N(t) \}$
2. Decode $P_Q(t)$ in accordance with the problem.
3. Evaluate the fitness value of each individual solution in decoded population $P_Q(t)$
4. Observe the best solution and store it.
5. If the stopping criteria is satisfied, then go to step 9;
6. At t = t+1 update $P_Q(t)$ to $P_Q(t+1)$ by applying quantum rotation
7. Again decode $P_Q(t+1)$ in accordance with the problem.
8. Evaluate the fitness value of each individual solution in decoded population $P_Q(t)$
9. Observe the best solution and store it.
10. Execute mutation operator.
11. If the stopping criterion is not satisfied, go to step 5.
12. Display the output

v. Applications

The mentioned algorithm can be applied on the problem like TSP, Knapsack, and Scheduling etc. In this paper only scheduling in distributed computing problem is focused.

A. *Scheduling problem in a Distributed Computing*

A distributed computing system has emerged as a powerful platform for providing higher computational power. The task assignment problem on such systems is a NP-complete problem. It is a mapping which assigns task of a program among different processors of distributed computer systems in order to optimize some characteristic parameters. The goal can be achieved by maximizing the utilization of resources and reducing the communication time between processors. These two criteria of the goal increases the complexity of the problem and a conflict arises in the problem as first criteria requires to distributes the tasks among different processors where second criteria require to distribute them on the same processors. Because of the intractable nature of the task assignment problem, it is desirable to develop good heuristic algorithm/hybrid algorithm according to the need of the problem.

The architecture of the Distributed Computing System (DCS) is shown in Fig. 1 below [17].In distributed computing systems all the stand alone machines have their own storage and connected through Internet. Being distributed over different geographical location such systems are loosely coupled and have high latency [18, 19, 20].

Scheduling of task among different machines is the main challenge in parallel and distributed computing systems. Scheduling is a fundamental issue to achieve high computing performance. Due to the involvement of large numbers of resources, prime objective is to optimize global use of resources. The scheduling problem of local resources on single system is already solved to much extend. There are two types of scheduler terms as local scheduler and global scheduler. Local scheduler is responsible of the scheduling inside each element of DC locally while global scheduler will take care of scheduling globally. In this paper, it is assumed that the local scheduler works in FCFS manner. Scheduling of jobs on DC nodes with appropriate resources is a NP-hard problem [21]. In more realistic cases, various issues have to consider during design of scheduling algorithm. To obtain parallelism, task graphs structure is considered. Other factors like arbitrary computation, task granularity and communication costs can be also considered to achieve parallelism.

The priorities of nodes are also determined statically before the beginning of scheduling process. The nodes with higher priority are preferred first for the scheduling. On the next, the best processor i.e. the processor with minimum task completion time is selected which results in better scheduling. The static approach may leads to an inefficient schedule. It may possible that some higher priority task can arrive after priority assignment during the scheduling. In such scenario, as there is no option to change the priority once after it has been decided. Thus it will reduce the overall performance of the system. In distributed computing system, nodes are connected via network but unaware about location of other computing nodes. All the nodes are capable to access the service of any other node.

The parameters considered during scheduling [22] might be turnaround time, system utilization, response time, throughput and waiting time etc. The factors that may have an effect on the scheduling decision are as below [22]

- *Node speed:* The speed of the computing processor on which the job is to execute. Faster job execution is expected on faster machine.
- *Number of processors in the node:* A node may consist of multiple processors. The performance of a node is directly propositional the number of processors.

Proc. of the Intl. Conf. on Advances In Computer and Electronics Technology- ACET 2014.

ISBN: 978-1-63248-024-8 doi: 10.15224/ 978-1-63248-024-8-06

- *Existing workload:* This refers to the number of modules (jobs) already allocated on the node. If the pre-assigned workload is higher, then turnaround time will also be higher.
- *Node's local scheduling policy:* DC scheduler only schedules the job on the appropriate nodes. The local scheduler is equally important to achieve high performance. It is impossible to achieve peak performance of DC with considering local scheduling policy.

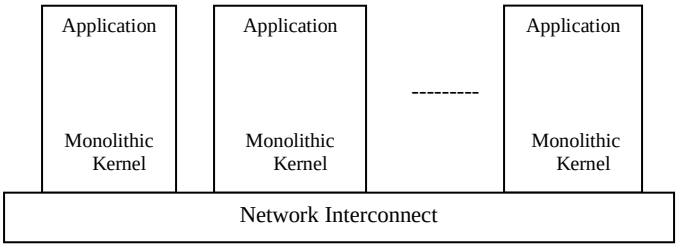

Figure 1. Architecture of a Distributed Computing System

B. *Scheduling in DC*

Initially, the tasks are distributed randomly by global scheduler. Makespan is defined as time required finishing latest task. Assumption for scheduling in distributed environment, tasks are independent, computing nodes have different computing speed, and uniform task migration cost. We have m computing node in distributed environment and n jobs. We want distributed the n jobs in m computing node so that the make span is minimal. This is combinatorial problem falls in NP Class [21]. GA is applied scheduling in distributed [22].Therefore we apply QGA for solving this. For implementing scheduling in distributed environment we generate the initial population using equation (1). Apply the initial coding using equation (2). Perform evolutionary rotation using equation (3) and equation (4). Mutation can be performed using equation (5). Generate random numbers uniformly between 0 to1, and generate randomly n/2+1 threshold points. Here threshold points map the parameters of the problem. If square of the random number lies between 1^{th} and $(l+1)^{th}$ threshold then we select $\alpha_i(t)$ and $\beta_i(t)$. For scheduling problem in distributed computing environment generate random numbers uniformly between 0 to1, and generate randomly. If square of the random number lies between l^{th} and $(l+1)^{th}$ threshold then we select $\alpha_i(t)$ and $\beta_i(t)$. Fitness of Population is calculated by using makespan. We perform the experiments after implementing algorithm discussed above in C++. The integrated development environment is Visual C++. System requirement is Windows 7, OS with 2 GB RAM and 3.2 GHz processor speed. The experiments have been simulated in C++.

VI. Experimental Results

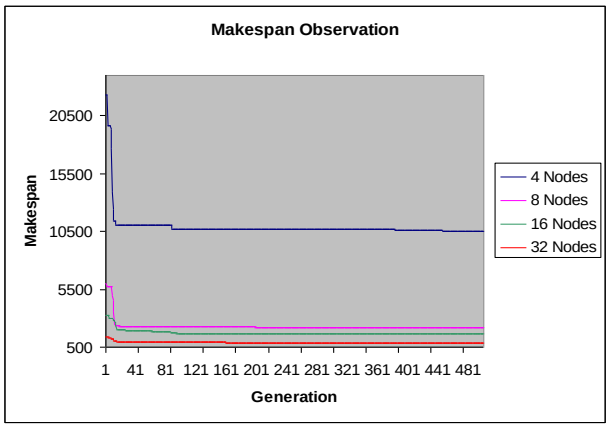

Figure 2. Makespan Observation with varying nodes

Generations- 500, Number of Task-500, Population Size-100 Arrival Rate Range 1-100 MIPS, Service Rate 101-200 MIPS, Range of Task Size 2000-5000 MI.The observations drawn from the above graphs are as follows:

1. By 300 generations solution converges.
2. From Fig. 2, it is observed that makespan decreases with increment of the number of distributed nodes i.e. in the case of for 500 tasks the makespan with 4 nodes is 10529.39 which becomes 2205.911 for 8 nodes and further 1623.609 16 nodes for 32 nodes 867.4721.

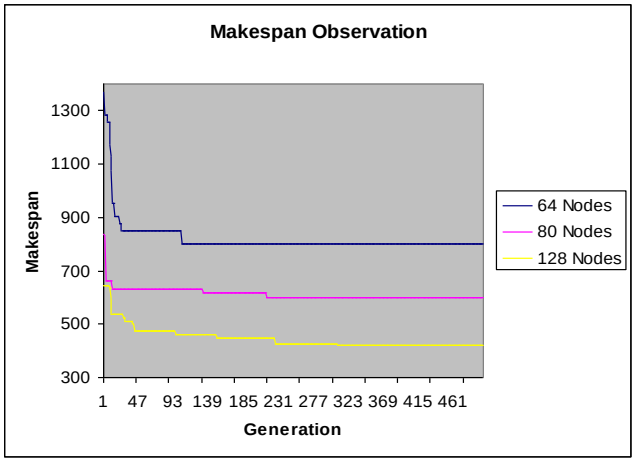

Figure 3. Makespan Observation with varying nodes

Number of Task-500, Population Size-100 Arrival Rate Range 1-100 MIPS, Service Rate 101-200 MIPS, Range of Task Size 2000-5000 MI

The observations drawn from the above graphs are as follows:

1. By 300 generations solution converges.
2. From figure 3, it is observed that makespan decreases with increment of the number of distributed nodes i.e. in the case of for 500 tasks the makespan with 64 nodes is 801.2102 which becomes 601.2674 for 80 nodes and further 421.0609 128 nodes.

Proc. of the Intl. Conf. on Advances In Computer and Electronics Technology- ACET 2014.

ISBN: 978-1-63248-024-8 doi: 10.15224/ 978-1-63248-024-8-06

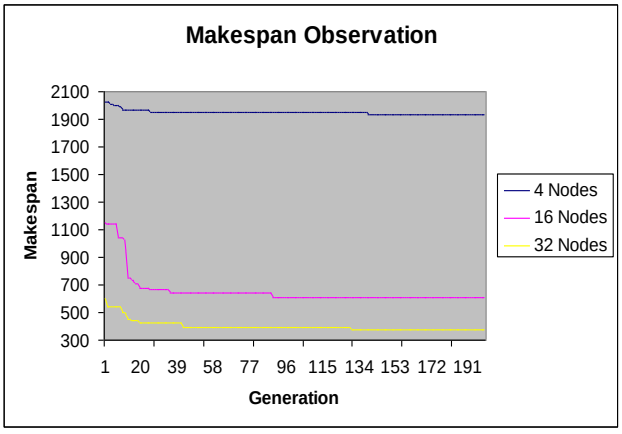

Figure 4. Makespan Observation with varying nodes

No of Task-200, Population Size-100 Arrival Rate Range 1-100 MIPS, Service Rate 101-200 MIPS, Range of Task Size 2000-5000 MI.

The observations drawn from the above graphs are as follows:

1. By 150 generations solution converges.
2. From Fig. 4, it is observed that makespan decreases with increment of the number of distributed nodes i.e. in the case of for 200 tasks the makespan with 4 nodes is 1936.83 which becomes 608.506 for 16 nodes and further 373.709 32 nodes.

VII. Conclusion and Future Scope

As discussed above, there is a variety of problems for which exact algorithm doesn't exists. Evolutionary techniques provide much efficient solutions for such problems. It is the point of research that for what type of problem which evolutionary technique may be applied and how efficiently it is possible to map the parameters of the various applications to that of the different heuristic methods. In recent, GA, PSO, etc. have been applied on such problems. Sometimes, two or more methods are mixed together making a hybrid algorithm to explore good results for specific type of problem [10, 16].In future it is our plan to propose QGA to solve channel allocation problem.

References

[1] S. Prakash and D.P. Vidyarthi, "Observations on Effect of IPC in GA Based Scheduling on Computational Grid", International Journal of Grid and High Performance Computing, 4(1): 67-80(2012)

[2] Y. Mingscheng, "Quantum computation, quantum theory and AI", Artificial Intelligence 174, 162-176 (2010).

[3] K.L.Grover,"A fast quantum mechanical algorithm for database search," in Proc. 28th ACM Symp. Theory Comput., Philadelphia, 212-221 (1996).

[4] W.P.Shor,"Algorithms for Quantum Computation:Discrete logarthims and factoring," in Proc. 35th Symp. Found. Comput. Sci., Los Alamitos, CA 20-22 (1994)

[5] K.Han, J.Kim, "Genetic quantum algorithms and its applications to combinatorial optimization problem," In Proc. of 2000 Congr. on Evol. Comput., USA 1354-1360 (2000).

[6] K.Han, J.Kim,"quantum inspired evolutionary algorithm for a class of combinatorial optimization," Computer Journal of IEEE Trans. on Evol. Comput., 6(6), 55-59 (2000).

[7] K.Han, J. Kim, " A quantum inspired evolutionary algorithm with a new termination criterion, he gate and two phase scheme," IEEE Trans. Evol. Comput. 6(6) 580-593 (2002).

[8] D.E. Goldbeg, Genetic Algorithms in Search, Optimization, and Machine Learning. Pearson, (2005).

[9] T.Mitchell, Machine-Learning. Mcgraw-Hill Series in Computer Science, (1997).

[10] N.Zhao, Z.Wu,Y.Zhao,T.Quan,"Ant colony optimization algorithms with mutation mechanism and its applications," Expert System with Applications, 37 4805-4810 (2010).

[11] T.Hogg, D.Portnov,"Quantum optimization" Inform.Sci. 128(3-4), 81-97 (2000).

[12] S. Prakash and D.P. Vidyarthi "A Novel Scheduling Model for Computational Grid using Quantum Genetic Algorithm", Journal of Supercomputing Springer, 65 (2), 2013, pp.742-770.

[13] Q.Niu, F.Zhou,T. Zhou,"Quantum genetic algorithm for hybrid flow shop scheduling problem to minimize total completion time," LNCS 6329 Part(II,) 21-29 (2010).

[14] J.Gu, X.Gu, M.Gu,"A novel parallel quantum genetic algorithm for stochastic job shop scheduling," J. Math. Anal. Appl. 355, 63-81 (2009).

[15] J.Xiao, Y.Yan, J. Zhang, Y.Tang,"A quantum inspired genetic algorithm for k- mean clustering," Expert Systems with Applications 37 (7) , 4966–4973 ,(2010)

[16] G.Huang, A.Lim," A hybrid genetic algorithm for the three index assignment problem," Europian Journal of Operational Research, 172(1), 249-257 (2006).

[17] S. Prakash and D.P. Vidyarthi, "Maximizing availability for task scheduling in computational grid using GA", Concurrency Comput. Pract. Ex. 2014.DOI:10.1002./cpe.3216

[18] S. Prakash and D.P. Vidyarthi, "A model for load balancing in computational grid", In Proc. High Performance Computing (HiPC11) Bangalore, 2011; 1-5.

[19] A .S. Tanenbaum, Modern Operating Systems, Second Edition PHI 2004.

[20] P.K. Sinha, Distributed Operating Systems Concepts and Design ,PHI 2005.

[21] S. Prakash and D.P. Vidyarthi, "Load Balancing in Computational Grid Using Genetic Algorithm", International Journal of Advances in Computing, Scientific and Academic Publishing, 1 (1), 2011, pp. 8-17.

[22] D.P.Vidyarthi, B.K. Sarker, A.K. Tripathi, and L.T. Yang, Scheduling in distributed computing systems: Analysis, design, and models, New York, Springer(2009).