International Journal on Recent and Innovation Trends in Computing and Communication ISSN: 2321-8169
Volume: 2 Issue: 11 3569 – 3573

Makespan Minimization Using WQACO Algorithm

Manoranjitham.M

Assistant Professor, Department of Information Technology,
Apollo Engineering College, Chennai – 600 124, Tamil Nadu, India.
m.manoranjitha@gmail.com

Abstract: Grid computing, a next leap in communication technology, a new trend in distributed computing system that enables utilization of idle resources existing worldwide, to solve data intensive and computationally intensive problems. The resources may either be homogeneous or heterogeneous in nature and they are shared from multiple administrative domains. The problem is divided into independent tasks and the tasks are executed by the resources available in grid. Scheduling these tasks to various resources in a grid is a very important problem and it is NP Complete. Hence we need a good task scheduling strategy to utilize the grids effectively such that make span is minimized. In literature, many heuristic approaches for scheduling are available that give near optimal solution. In this paper we propose a weighted QoS factor enabled ant colony algorithm for scheduling independent tasks on heterogeneous grid resources. The main contributions of our work are to minimize the makespan with Qos satisfaction and the results are compared with max-min and min-min algorithm.

Keywords: *Grid Computing, Heuristic, Ant Colony, Makespan, QoS*

*****

1. Introduction

Grid computing is a term referred to a collection of resources from multiple administrative domains to reach a common goal. Currently, scientists, engineers and animation designers are working with huge volume of data. They require high storage and high computation power to process their problems. It takes days, even months or years to complete their own task if they use only their own resources. Grid computing enables them to use the idle resources existing worldwide and they can carry out their tasks faster with the help of these heterogeneous resources [4] . Grid computing enables them to reduce the upfront cost as the highly expensive resources are being shared instead of being owned. People from many institutions working to solve a common problem forms a virtual organisation [7] with the help of grids.

In grids we must consider both the network status and the resource status. If either the network or the resource becomes unstable then the tasks would be failed or the overall completion time would be very large. Hence the traditional FCFS (First Come First Served) or SJF (Shortest Job First) algorithms cannot be used to schedule the tasks to grid. Therefore we need an efficient task scheduling algorithm for these problems in the grid environment [5].

In grids users may face hundreds of thousands of resources to utilize. It is impossible for anyone to manually assign tasks to computing resources in grids. Therefore task scheduling is a very important issue in grid computing. For example BOINC, open source software for volunteer computing and grid computing, task scheduling is one of the important key factors for achieving teraflops of performance. Because of its importance, many job scheduling algorithms for grids have been proposed.

A good scheduler would adjust its scheduling strategy according to the changing status of the entire environment and the types of tasks. Therefore, a dynamic algorithm in tasks scheduling such as Ant Colony Optimization (ACO) [6] is appropriate for grids.

ACO is a heuristic algorithm [8] with efficient local search for combinatorial problems. ACO imitates the behaviour of real ant colonies in search for its food and connect to each other by pheromone laid on paths travelled. Many researchers solve NP-hard problems such as Travelling salesman problem, graph colouring problem using ACO approach.

This paper applies the ACO algorithm to schedule the tasks of the problem in grid computing. We assume that each task is an ant and the algorithm sends the ant to search for resources. We also assume that the tasks are independent of each other. We also modify the pheromone update based on the QoS factor and the availability of the resources. Finally we compare the results with min-min algorithm and max-min algorithm for scheduling the same tasks.

The rest of the paper is organised as follows. Section 2 contains the literature survey done on existing scheduling algorithms for grids. Section 3 describes the proposed Weighted QoS Ant Colony Optimization algorithm (WQACO) and Section 4 describes experimentation results.

2. Literature Survey

2.1. Balanced Ant Colony Algorithm

Balanced Ant Colony Optimization [1] inherits the basic ideas of ant colony algorithm and schedules the tasks in grid based on the load.

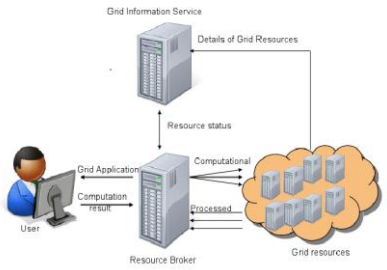
2.2. Min-min and Max-Min Approaches

Min-min [2] set the tasks which can be completed earliest with highest priority. Max-min [2] sets the tasks which has the maximum earliest completion time with the highest priority. For instance, if there is only one long task, min-min will execute short tasks in parallel and then execute long task. Max-min will execute short task and long task in parallel.

3. Proposed Weighted QOS Ant Colony Optimization Algorithm (WQACO)

WQACO inherits the basic ideas from ACO algorithm to minimize makespan of tasks in grid environment and it also considers the task allocation to a particular resource satisfies the QoS factors such as cost, RAM and deadline.

3.1. Basic Grid model



Basic Grid Model

The basic grid model generally is a collection of various types of systems; each consists of several computational resources which may be homogeneous or heterogeneous. The four main components of grid model are user, resource broker, Grid Information Service (GIS) and the resources, as shown in the above figure. The problem is submitted to the broker in the grid. The broker splits the problem into various independent tasks and distributes them to several resources based on the user's requirements and availability of resources. GIS keeps the status information of all resources which helps the broker for scheduling.

3.2. Formulation of WQACO

Generally, the WQACO is described as follows. There are n tasks to be processed by m machines where n>m. The following assumptions are made

- All jobs are independent of each other and no priorities are among them.
- All machines and jobs are simultaneously available at the initial time.
- All operations of a given job must be processed according to their precedence.
- One machine can only process an operation of a job at the same time and the processing cannot be interrupted before an operation is completed.
- The transportation time of a job from one machine to another is negligible and the machine setup time for an operation is included in its processing time.

The eleven state mapping heuristics [3] were evaluated using simulated execution times for an HC environment. Because these are static heuristics, it is assumed that an accurate estimate of the expected execution time for each task on each machine is known prior to execution and contained within an ETC (Expected time to compute) matrix.

WQACO changes the pheromone density based on the availability of the machine and QoS factors satisfaction by applying the local and global pheromone update functions. The purpose is to minimize the completion time for each task and hence reducing the overall makespan.

3.3. WQACO Definition and Methodology

The relationship between the ant system and the grid system is mapped as follows.

- An ant – An ant in the ant system is a task in the grid system.
- Pheromone – pheromone value on a path in the ant system is a weight for a resource is the grid system.

The weight of resource is the capability of the resource; the value represents the QoS satisfaction of the resource. A resource with a larger weight value means that the resource has a better computing power.

The scheduler or resource broker collects data from GIS or the information server and uses the data to calculate the weight value of a resource. The pheromone (weight) value is stored in the scheduler and the scheduler uses it as the parameter of WQACO algorithm. At last, the scheduler selects a resource by scheduling algorithm and sends the task to the selected resource.

Let us assume n tasks are scheduled in m machines where n>m. The ETC (Expected Time to Compute) matrix is of form, n x m, where ETC_{ij} represents accurate estimate of expected execution time for each taski on machinej.

Machine Capability matrix for each machinej, indicates the QoS factors (Cost, RAM and deadline) associated with machinej for each taski. Let k indicates the number of QoS factors, (k=3 in our case).Hence the machine capability matrix for a machinej is given by,

$$(\text{MachineCapability}_{ik})j=\begin{pmatrix} cost_1 & RAM_1 & deadline_1 \\ \vdots & \vdots & \vdots \\ cost_n & RAM_n & deadline_n \end{pmatrix}$$

For each machine the $\text{MachineCapability}_{ik}$ values are calculated.

Task requirements matrix represents the user requirements of QoS factors for executing the particular task.

$$\text{Task Requirements}_{ik}=\begin{pmatrix} cost_1 & RAM_1 & deadline_1 \\ \vdots & \vdots & \vdots \\ cost_n & RAM_n & deadline_n \end{pmatrix}$$

Since we manipulate multiple QoS factors, we assign a guided probability value for each QoS factor. Hence

$$w =< w_1..w_k > 0 \le w_k \le 1 \quad \sum_{i=1} w_i = 1$$

The above equation gives the guided probability value for cost, RAM and deadline respectively. Let us have the guided probability values as 0.5, 0.25 and 0.25 for cost, RAM and deadline respectively. We introduce satisfy operator ₪. R_j ₪ T_i means that the resource Rj can satisfy the task Ti and guarantees QoS parameters.

$$R_j \text{ ₪ } J_i = \sum_{i=1}^{k} \frac{QOS^{Task\ Requirements_i}}{QOS^{Machine\ Capability_j}} * W_1 \ge 1$$

 ISSN: 2321-8169
3569 – 3573

(k = the number of QoS parameters) **-- 1**

If all the QoS factors are satisfied then, the initial pheromone indicator value is calculated based on the QoS factor values in machine capability and task matrix and guided probability value as,

$$PI_{ij} = \frac{(MachineCapability_{i1})_j}{TaskRequirements_{i1}} * 0.5 + \frac{TaskRequirements_{i2}}{(MachineCapability_{i2})_j} * 0.25 +$$

Where PI_{ij} indicates the pheromone indicator value for taski assigned to machine. Task Requirementsil represents the value of user expectation of QoS factors (Cost, RAM and deadline) for each taski. (MachineCapabilityil)j represents the value of machine capability matrix, for the particular machine j, the MachineCapabilityil indicates the QoS factors of machine for executing taski.

Hence for m tasks and n machines, PI matrix will be of the form,

$$PI = \begin{array}{c} \\ t_1 \\ \\ t_n \end{array} \begin{array}{c} \begin{array}{cccc} m1 & m2 & \ldots. & m_m \end{array} \\ \begin{pmatrix} PI_{11} & PI_{12} & ..... & PI_{im} \\ . & . & ..... & \\ PI_{n1} & PI_{n2} & ..... & PI_{nm} \end{pmatrix} \end{array}$$

In each iteration we select the largest entry from the matrix, for instance if PI11 is the largest entry in the matrix and if no other taski is allocated to the machine1 already, then the taski is allocated to machinej else the task is allocated to the next machine that has the next highest pheromone value. After a task is allocated to a machine a local pheromone (column) update is made. The local pheromone update is given by the formula,

$$PI_{ij} = (1-\rho)PI_{ij} + (\rho * \frac{1}{N_r}) \quad \text{-- 3}$$

Where PI_{ij} indicates the PI values on the machine j where the task is allocated. ρ indicates the evaporation rate of the pheromone (0< ρ<1) , Nr indicates the number of resources or machines in the grid. Local update is made in order avoid same machine to be over loaded with multiple tasks and to avoid stagnation.

After the task is completed on the machines, global update is made to the entire PI matrix. The PI matrix is modified with a update function based on the availability value of the machinej after completing the taski,

$$PI_{modified} = (1-\rho)PI_{initial} * \frac{1}{Machine\ Availability_j} \quad \text{--- 4}$$

3.4. WQACO Algorithm

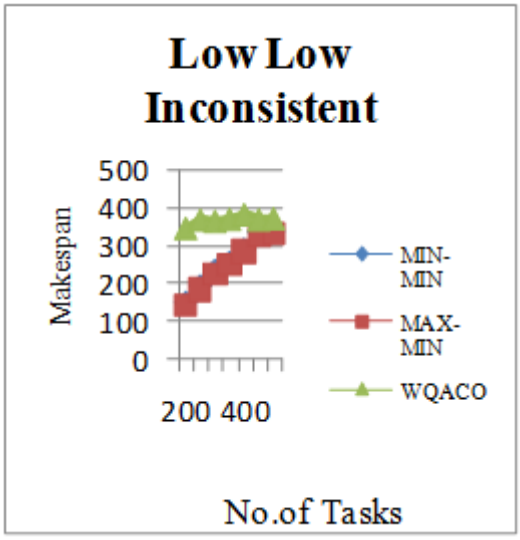
1. ***Input*** *: ETC matrix of size n x m*
   Task Requirement matrix, Machine capability matrix for K QoS parameters

2. ***Method:***
   2.1. *First check for the Qos Requirement satisfaction for all K QoS parameters using the equation 1.*
   2.2. *Calculate the initial pheromone value using the equation 2.*
   2.3. *Select the machine which has highest pheromone value and QoS satisfied.*
   2.4. *Assign that machine to the corresponding task.*
   2.5. *Update the local pheromone value using the equation 3.*
   2.6. *After completion of the task update the global pheromone value using the equation 4.*
   2.7. *Repeat the step 2 until all tasks are assigned.*

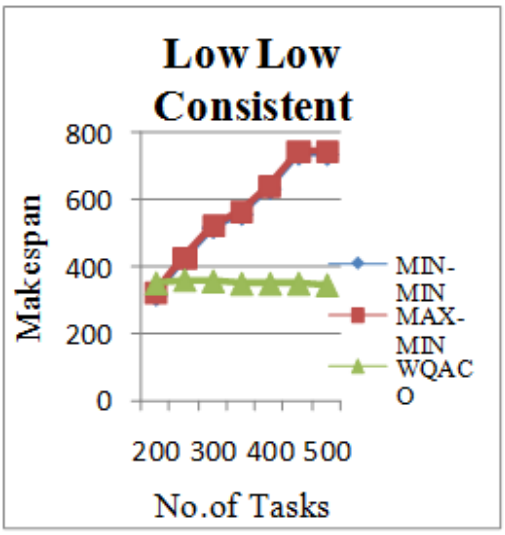
3. *Calculate the makespan.*

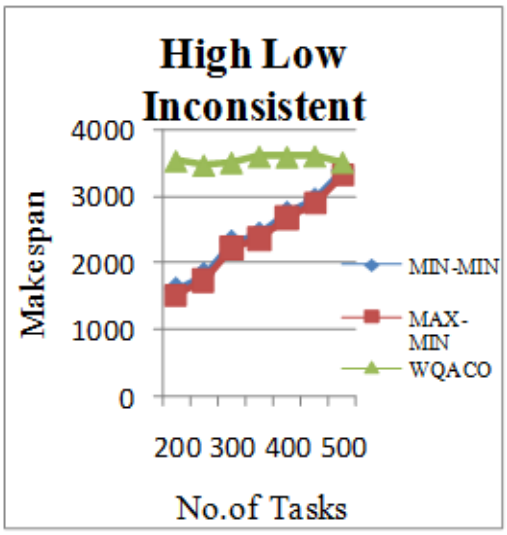
4. *End*

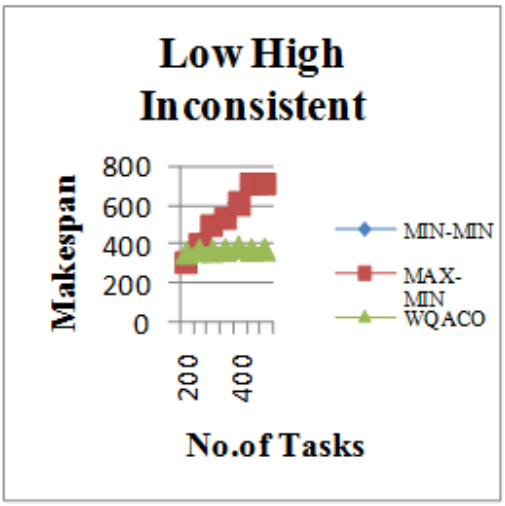
4. Experimentation Result

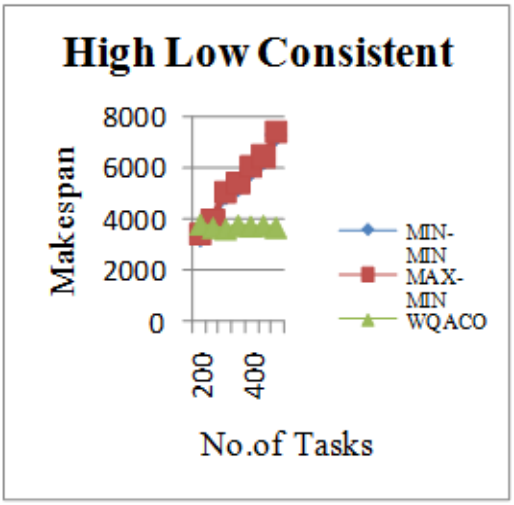


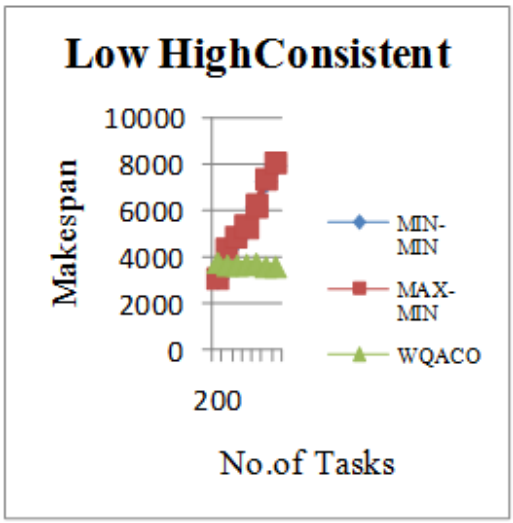
ISSN: 2321-8169
3569 – 3573

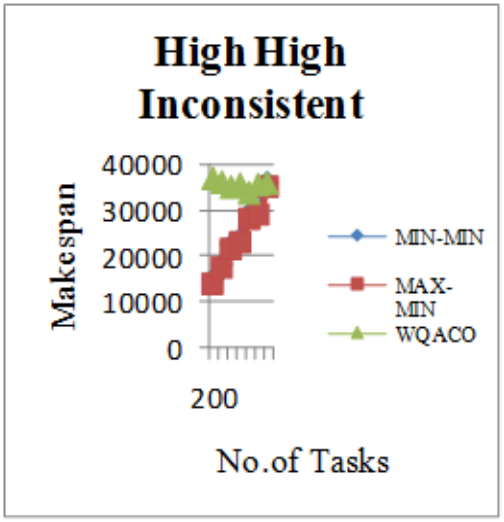


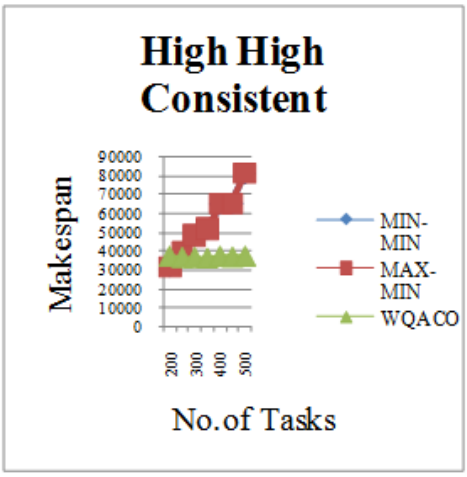












Low indicates Low task and low machine heterogeneity.

4. Conclusion

In this paper we presented weighted Qos Ant colony, which is an extension of existing Ant colony algorithms. Our algorithm yields significant improvements in performance over existing heuristic algorithms in most of the cases. In inconsistent combinations for all four cases, our algorithm performance is low when the number of tasks is low and it improves when the number of tasks increases. This algorithm removes stagnation by local as well as global pheromone updation. This algorithm first checks the Qos Satisfying criteria and then schedules the task to the relevant machine.

References

[1] Ruay-Shiung Chang, Jih Sheng Chang, Po-Sheng Lin, "*An algorithm for balanced job scheduling in grids*", Future Generation Computer Systems 25(2009) 20-27

[2] Kobra Etminani, M. Naghibzadeh, "*A Min-Min Max-Min Selective Algorithm for Grid Task Scheduling*", Internet 2007,ICI 2007,3rd IEEE/IFIP International Conference in Central Asia on, Sept. 2007.

[3] T. Braun, H. Siegel, N. Beck, L. Boloni, M. Maheswaran, A. Reuther, J. Robertson, M. Theys, B. Yao, D. Hensgen, and R. Freund. "*A comparison study of static mapping heuristics for a class of meta tasks on heterogeneous computing systems*", 8th IEEE Heterogeneous Computing Workshop (HCW'99), pages 15-29, April 1999.

[4] J.G.Webster, "*Heterogeneous Distributed Computing*," Encyclopaedia of Electrical and Electronics Engineering, Vol. 8, pp. 679-690, 1999.

[5] D.Feitelson,L.Rudolph,U.Schwiegel shohm,K.Sevcik and P.Wong, "*Theory and Practice in Parallel Job Scheduling*", JSSPP, pp. 1-34, 1997.

[6] Jamshid Bagherzadeh, Mojtaba Madadyar Adeh "*An Improved Ant Algorithm for Grid Scheduling Problem*", International CSI Computer Conference CSICC'09 pp. 323 – 328, 2009

[7] Bing Tang, Yingying Yin, Quan Liu and Zude Zhou "*Research on the Application of Ant Colony Algorithm in Grid Resource Scheduling*" Journal on Wireless Communications, Networking and Mobile Computing, 2008, Page(s):1 - 4.

[8] Li Li, Wang Keqi, Zhou Chunnan "An Improved Ant Colony Algorithm Combined with Particle Swarm Optimization Algorithm for Multi-objective Flexible Job Shop scheduling Problem" , International Conference on Machine Vision and Human-machine Interface 2010