International Journal of Modern Engineering Research (IJMER)
www.ijmer.com Vol.1, Issue1, pp-134-139 ISSN: 2249-6645

IMPLEMENTATION OF OBJECT ORIENTED APPROACH TO MODIFIED ANT ALGORITHM FOR TASK SCHEDULING IN GRID COMPUTING

CH SOMESWARA RAO[1], V MNSSVKR GUPTA[2], K.V.S. MURTHY[3], J RAJANIKANTH[4]
[1-4] Department of CSE, S.R.K.R Engg. College, Affiliated to Andhra University, Bhimavaram, W.G.District, Pin-534 204, A.P. INDIA

***Abstract*— Now a day's Resource management and task scheduling are very important and complex problems in grid computing environment. It is necessary to predict resource state to get proper task scheduling. In this paper we propose object oriented modified ANT algorithm is a new heuristic algorithm. The inherent parallelism and scalability make the modified ANT algorithm very suitable to be used in grid computing task scheduling, whose structure is changes dynamically almost all the time. The scalability of proposed algorithm is validating by proposing a simple Grid simulation architecture for resource management and task scheduling. The algorithm when implemented in simulation environment produced good results in terms of minimal response time; maximum resource average utilization and task fulfill proportion.**

***Index Terms*— Grid Computing, Task Scheduling, ANT Algorithm, Modified ANT Algorithm, Resource Management**.

I. INTRODUCTION

Grids enable the sharing, selection, and aggregation of a wide variety of resources including super computers, storage systems, data sources, and specialized devices that are geographically distributed and owned by different organizations for solving large-scale computational and data intensive problems in science, engineering, and commerce. Grid computing, most simply stated, is distributed computing taken to the next evolutionary level [1,2].

Resource management and task scheduling are very important and complex problems in grid computing [3]. In grid environment, autonomy resources are linked through internet to form a huge virtual seamless environment. The resources in the grid are heterogeneous and the structure of the grid is changing almost all the time. In a grid, some resources may fail, some new resources enroll the grid, and some resources resume to work. So, it is necessary to do resource state prediction to get proper task scheduling. Since the static algorithms like round robin algorithm, fastest processor-largest task first algorithm etc, are no longer suitable for the effective utilization of the grid, a good task scheduling method should be used which is distributable, scalable and fault tolerant[4,5].

Manuscript received September 4th , 2011
F. A. Author is with the SRKR Engineering College, Andhra Pradesh, INDIA
S. B. Author is with the SRKR Engineering College, Andhra Pradesh, INDIA

The aim of this paper is to simulate and analyze a task scheduling algorithm which is based on well known ant-algorithm. The task scheduling algorithm should work better in the grid environment, in which the structure of the grid changes dynamically almost all the time. It should reduce the total execution time of jobs submitted to the grid, by effectively scheduling the jobs on to the appropriate resources in the grid. It should also reduce the cost of using the resources to execute the jobs.

So, both the total execution time of the jobs and the cost of executing the jobs should be taken into consideration in scheduling the jobs on to the resources. If the cost of resources is not a factor modified forms of ant-algorithm, which can further reduce the total execution of the jobs.

A. ANT ALGORITHM

ANT algorithm is a new heuristic, predictive scheduling algorithm; it is based on the behavior of real ants. When the blind insects, such as ants look for food, every moving ant lays some pheromone on the path, then the pheromone on shorter path will be increased quickly, the quantity of pheromone on every path will affect the possibility of other ants to select path. At last all the ants will choose the shortest path.

- When a resource 'j' enrolls the Grid,

$$\tau_j(0) = m \times p + c / s_j$$

where 'm' is the No. of PEs, 'p' is the MIPS of one PE, 'c' is the size of the parameters, 'sj' is the parameter transfer time from resource 'j' to resource monitor.

- The pheromone on path from schedule centre to the corresponding resource will be changed as

$$\tau_j^{new} = \rho.\tau_j^{old} + \Delta\tau_j$$

is the change of pheromone on path from the schedule center to resource j;
ρ is the permanence of pheromone ($0 < \rho < 1$)
$1 - \rho$ is evaporation of pheromone
when task is assigned to resource j,

= - k, k is the compute and transfer quality of the task.
when task successfully returned from resource j,
= Ce x k, Ce is the encourage factor.
when task failed returned from resource j,
= Cp x k, Cp is the punish factor.

- The possibility of task assignment to every resource will be recomputed as:

$$\rho_j^k(t) = \frac{[\tau_j(t)]^\alpha * [\eta_j(t)]^\beta}{\sum_u [\tau_u(t)]^\alpha * [\eta_u]^\beta}$$

$$= \quad 0 \qquad \text{others}$$

is the pheromone intensity on the path from schedule center to resource j

$\eta_j(t)$ is the innate performance of the resource;
α is the importance of the pheromone;
β is the importance of resource innate attributes;

B. Drawbacks of the ANT algorithm

- The scheduler schedules a task based on the possibilities of the resources.
- The problem with this algorithm is, it may schedule a task to a resource with low possibility even if the resources with high possibility are free.
- But this randomness is required because if the tasks are always scheduled to the resource with high possibility, then the load on the resource may be increased and the jobs may be kept waiting in the queue for the resource to be free.
- It uses centralized scheduling scheme.

II. RELATED WORD

The efficient scheduling of independent computational jobs in a heterogeneous computing (HC) environment such as a computational grid is clearly important if good use is to be made of a valuable resource. Scheduling algorithms can be used in such a system for several different requirements [6]. The first, and most common, is for planning an efficient schedule for some set of jobs that are to be run at some time in the future, and to work out if sufficient time or computational resources are available to complete the run a priori. Static scheduling may also be useful for analysis of heterogeneous computing systems, to work out the effect that losing (or gaining) a particular piece of hardware, or some sub network of a grid for example, will have. Static scheduling techniques can also be used to evaluate the performance of a dynamic scheduling system after it has run, to check how effectively the system is using the resources available. The Ant Colony Optimization (ACO) meta-heuristic was first described in[7] as a technique to solve the traveling salesman problem, and was inspired by the ability of real ant colonies to efficiently organize the foraging behavior of the colony using external chemical pheromone trails as a means of communication. ACO algorithms have since been widely employed on many other combinatorial optimization problems including several domains related to the problem in hand, such as bin packing and job shop scheduling, but ACO has not previously been applied to finding good job schedules in an HC environment.

Although various classification of task scheduling strategies exist, depending on the type of the grid, the scheduler organization and task scheduling strategy has to be chosen such that the resources in the grid are effectively utilized. Along with the scheduling organization, state estimation is also necessary for the dynamic grid. To achieve maximum resource utilization and high job throughput, re-scheduling of the jobs on different resources is also sometimes required.

Since the static algorithms like round robin algorithm, fastest processor-largest task first algorithm etc, are no longer suitable for the effective utilization of the grid, a good task scheduling method should be used which is distributable, scalable and fault tolerant.

In high throughput computing, the Grid is used to schedule large numbers of loosely coupled or independent jobs, with the goal of putting unused processor (resource) cycles to work. Build on the Internet and World Wide Web, the grid has emerged as a new class of infrastructure. By discovering and obtaining access to remote resources, the grid can provide scalable, secure, high performance calculating resources to make it possible for distributed scientific groups to work together to share resources in a large scale and to solve complex scientific problem that never practice in the previous time.

In grid, resources are distributed, heterogeneous, dynamic and instability. Dorigo et.al proposed [8], which is a new heuristic algorithm and based on the behavior of real ants. When the blind insects, such as ants look for food, the moving ant lays some pheromone on the ground, thus making the path it followed by a trail of this substance. While an isolated ant moves essentially at random, an ant encountering a previously laid trail can detect it and decide with high probability to follow it, thus reinforcing the trail with its own pheromone. The collective behavior that emerges means where the more the ants are following a trail, the more that trail becomes attractive for being followed. The process is thus characterized by a feedback loop, where the probability with which an ant chooses an optimum path

increases with the number of ants that chose the path in the preceding steps. These observations inspired a new type of algorithm called ant algorithms or ant systems.

The task-scheduling algorithm should work better in the grid environment, in which the structure of the grid changes dynamically almost all the time. It should reduce the total execution time of jobs submitted to the grid, by effectively scheduling the jobs on to the appropriate resources in the grid. It should also reduce the cost of using the resources to execute the jobs. So, both the total execution time of the jobs and the cost of executing the jobs should be taken into consideration in scheduling the jobs on to the resources. If the cost of resources is not a factor (situations like, all the resources of our interest belong to the same organization), modified forms of ant-algorithm, which can further reduce the total execution of the jobs.

III. SYSTEM ARCHITECTURE

In Grid environment, the client nodes can enroll the Grid at any time, deliver requests to the schedule center, and monitor the implementation of themselves tasks. There are five important modules in the schedule center; the architecture of the scheduler center can be expressed as shown in the Fig 1. Each node denotes a client or a Grid resource, and each edge denotes a link between nodes.

When a grid client delivers a request, the grid works as follows:

- The client delivers a request that contains an application description to the task receiver. The description is about the work load of the application, communication load, and time limit, etc.
- The task receiver queues the tasks in priority, and delivers the first task in the queue to scheduling manager. The receiver maintains an unscheduled task queue; record the client name and user requirements, application workload, communication load, and time limit, etc.
- The scheduling manager selects a most appropriate scheduling scheme from all schemes according to the resource graph and user requirement, then delivers the scheme to task dispatcher and inform resource monitor. This is the most important and complex scheduling in the schedule center, there are many scheduling strategies can be put in the scheduling manager. We design the ant algorithm based strategy, and will discuss it in the following section.
- Task dispatcher delivers the task to the selected resource, and gives transfer delay and actual task assignment result to the resource monitor. The dispatcher maintains a scheduled task queue; record the assigned resource name, application work load, communication load, and time limit, etc. When some task is finished or failed, the dispatcher delete the task in the queue or put the task back to unscheduled task queue, and notifies the resource monitor.
- The resource monitor maintains the up to date of every resource and revises the resource graph.

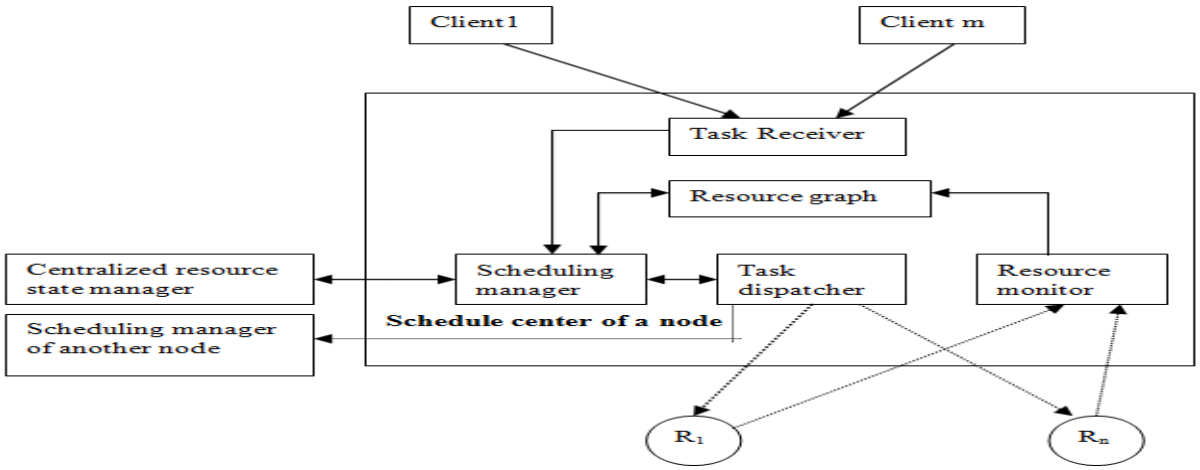

Fig 1 Detailed System Design

- We use a graphic to express the grid environment, each node denotes a client or a grid resource (number of processors, speed of each processor, I/O bandwidth, RAM capability, and disk capacity, etc.), and each edge denotes a link between nodes (bandwidth, delay, quantity of pheromone, etc.).

A. Improvements to the above said ANT algorithm

- The algorithm can be modified by using a threshold to minimize the total processing time.
- The processing costs of the tasks also can be controlled.

B. Modified Ant Algorithm

- When a resource 'j' enrolls the Grid,

$$\tau_j(0) = m \times p + c / s_j$$

where 'm' is the No. of PEs
'p' is the MIPS of one PE
'c' is the size of the parameters
'sj' is the parameter transfer time from resource 'j' to resource monitor.
The pheromone on path from schedule centre to the corresponding resource will be changed as

$$\tau_j^{new} = \rho.\tau_j^{old} + \Delta\tau_j$$

is the change of pheromone on path from the schedule center to resource j;
ρ is the permanence of pheromone (0 < ρ <1)
1 - ρ is evaporation of pheromone.
when task is assigned to resource j,
= - k, k is the compute and transfer quality of the task.
when task successfully returned from resource j,
= Ce x k, Ce is the encourage factor.
when task failed returned from resource j,
= Cp x k, Cp is the punish factor.

- The possibility of task assignment to every resource will be recomputed as:

$$\rho_j^k(t) = \frac{[\tau_j(t)]^\alpha * [\eta_j(t)]^\beta}{\sum_u [\tau_u(t)]^\alpha * [\eta_u]^\beta}$$

$$= \quad 0 \text{ others}$$

$\tau_j(t)$ is the pheromone intensity on the path from schedule center to resource j
$\eta_j(t)$ is the innate performance of the resource;
α is the importance of the pheromone;
β is the importance of resource innate attributes;

- The scheduler finds a resource 'j' on which a new task 'k' is to be scheduled at time 't' with a probability equal to it's corresponding $\rho_j^k(t)$ until $(\rho_h^k(t) - \rho_j^k(t)) \le T_h$.

where 'h' is the resource with highest $\rho^k(t)$.
T_h' is the threshold which will be taken as 1/ (no. of resources).

- The scheduler finds a resource 'i' taking one resource at a time in the ascending order of C_i / ($MIPS_i$) until $|\rho_j^k(t) - \rho_i^k(t)| \le T_i * \ell$

where C_i is the cost of the resource 'i' per second
$MIPS_i$ is the total MIPS of the resource 'i'
$T_i = Th - (\rho_h^k(t) - \rho_j^k(t))$ and
'ℓ' is the cost reduction factor which is selected by the grid user who submits the task. $0 \le \ell \le 1$.
The scheduler schedules the new task 'k' on to the resource 'i'.

IV. IMPLEMENTATION

In this paper we consider Grid Simulator(GridSim) toolkit, which supports modeling and simulation of a wide range of heterogeneous resources, such as single or multiprocessors, shared and distributed memory machines such as PCs, workstations, SMPs, and clusters managed by time or space-shared schedulers. GridSim can be used for modeling and simulation of application scheduling on various classes of parallel and distributed computing systems such as clusters, Grids, and P2P networks. The resources in clusters are located in a single administrative domain and managed by a single entity whereas, in Grid and P2P systems, resources are geographically distributed across multiple administrative domains with their own management policies and goals. The schedulers in cluster systems focus on enhancing overall system performance and utility, as they are responsible for the whole system.

The GridSim toolkit provides a comprehensive facility for simulation of different classes of heterogeneous resources, users, applications, resource brokers, and schedulers. It can be used to simulate application schedulers for single or multiple administrative domain(s) distributed computing systems such as clusters and Grids. Application schedulers in Grid environment, called resource brokers, perform resource discovery, selection, and aggregation of a diverse set of distributed resources for an individual user.

A. GridSim Entities

GridSim supports entities for simulation of single processor and multiprocessor, heterogeneous resources that can be configured as time or space shared systems. It allows setting their clock to

different time zones to simulate geographic distribution of resources. It supports entities that simulate networks used for communication among resources. The design and implementation issues of GridSim entities are discussed below:

i. **User** – Each instance of the User entity represents a Grid user. Each user may differ from the rest of the users with respect to the following characteristics:
   - Types of job created e.g., job execution time, number of parametric replications, etc.,
   - Scheduling optimization strategy e.g., minimization of cost, time, or both,
   - Activity rate e.g., how often it creates new job,
   - Time zone, and
   - Absolute deadline and budget.

ii. **Broker** – Each user is connected to an instance of the Broker entity. Every job of a user is first submitted to its broker and the broker then schedules the parametric tasks according to the user's scheduling policy. Before scheduling the tasks, the broker dynamically gets a list of available resources from the global directory entity. Every broker tries to optimize the policy of its user and therefore, brokers are expected to face extreme competition while gaining access to resources.

iii. **Resource** – Each instance of the Resource entity represents a Grid resource. Each resource may differ from the rest of resources with respect to the following characteristics:
   - Number of processors;
   - Cost of processing;
   - Speed of processing;
   - Internal process scheduling policy e.g., time shared or space shared;
   - Local load factor; and
   - Time zone.

iv. **Grid Information Service** – It provides resource registration services and maintains a list of resources available in the Grid.

v. **Input and Output** –The flow of information among the GridSim entities happen via their Input and Output entities. Every networked GridSim entity has I/O channels, which are used for establishing a link between the entity and its own Input and Output entities.

V. RESULTS

Graphical User Interface (GUI) is used for better usability of the system. In order to facilitate software (class) reuse, the GUI is solely implemented in a separate class. Another class is used for reading data (resource information) from user specified resource file. Resource information, user information, and task information should be maintained in the Graphical user Interface. We give appropriate data for source info, resource info and user info, based on the given input we have apply the ANT, Round Rabin, and Modified ANT. Finally Fig 6 shows the Modified ANT algorithm is better in performance wise.

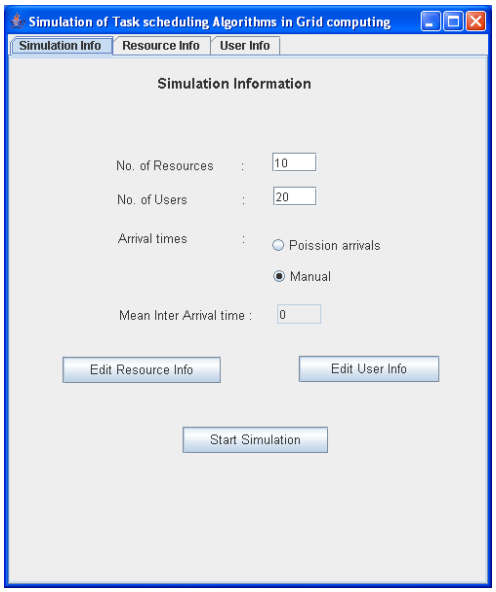

Fig 2. Source info

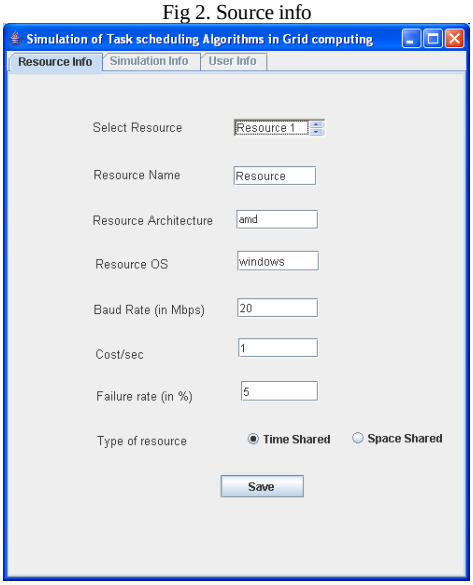

Fig 3 Resource Information

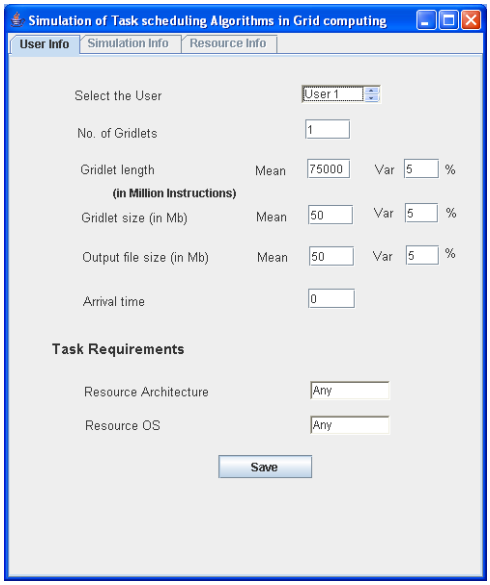

Fig 5 User information

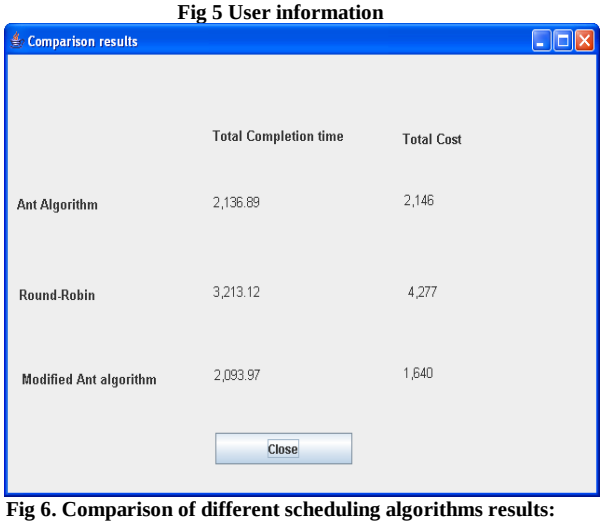

Fig 6. Comparison of different scheduling algorithms results:

VI. CONCLUSION AND FUTURE WORK

Since the structure of the grid dynamically changes, there is no particular scheduling algorithm, which can effectively utilize all the resources in a grid. But, the algorithm should be distributable, scalable and fault tolerant. It should also estimate the state of the resources, which are currently in the grid. And predictive state estimation is better than non-predictive state estimation because it uses the current state as well as the historical status of the resources. So, the static algorithms like round robin scheduling are no longer suitable.

The Ant algorithm is a heuristic predictive state estimating scheduling algorithm, which is distributable, scalable, and fault tolerant. The inherent parallelism and scalability make the algorithm very suitable to be used in grid computing task scheduling. The modified form of ant-algorithm can be used if cost of using the resources is not a concern. The selection of threshold value in the modified ant-algorithm is very important.

As a future work, this scheduling method can be put into actual Grid environment for validation to make QOS scheduling. This algorithm can be made more suitable for wide use if pricing factor will also be included. The ant algorithm can also be applied to search for optimized path in computer networks.

References

[1] Foster I, Kesselman C (eds.). The Grid: Blueprint for a Future Computing Infrastructure. Morgan Kaufmann: San Francisco, CA, 1999

[2] Chetty M, Buyya R. Weaving computational Grids: How analogous are they with electrical Grids? Journal of Computing in Science and Engineering (CiSE) 2001; (July–August).

[3] Sinha PK. Distributed Operating Systems: Concepts and Design. IEEE Press: New York, NY, 1997.

[4] Ekemecic I, Tartalja I, Milutinovic V. A survey of heterogeneous computing: Concepts and systems. Proceedings of the IEEE 1996; **84**(8):1127–1144.

[5] Buyya R, Abramson D, Giddy J. Nimrod/G: An architecture for a resource management and scheduling system in a global computational Grid. The 4th International Conference on High Performance Computing in Asia-Pacific Region (HPC Asia'2000), Beijing, China, 2000. IEEE Computer Society Press: Los Alamitos, CA, 2000.

[6] Braun TD, Siegel HJ, Beck N, Boloni LL, Maheswaran M, Reuther AI, Robertson JP, Theys MD, Yao B. A taxonomy for describing matching and scheduling heuristics for mixed-machine heterogeneous computing systems. Proceedings IEEE Workshop on Advances in Parallel and Distributed Systems, October 1998.

[7] M. Dorigo et L.M. Gambardella, Ant Colony System : A Cooperative Learning Approach to the Traveling Salesman Problem, IEEE Transactions on Evolutionary Computation, volume 1, numéro 1, pages 53-66, 1997

[8] M. Dorigo, V. Maniezzo, et A. Colorni, Ant system: optimization by a colony of cooperating agents, IEEE Transactions on Systems, Man, and Cybernetics--Part B , volume 26, numéro 1, pages 29-41, 1996.