Volume 5, Issue 8, August 2015 ISSN: 2277 128X

International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: www.ijarcsse.com

A Time Efficient Task Allocation Algorithm

Vaishali Gupta
M.tech, CSE Dept.
GIMT Kanipla (KUK)
Haryana, India

Er. Pankaj Dev Chadha
Assistant Professor, CSE Dept.
GIMT Kanipla (KUK)
Haryana, India

Abstract: Abstract - Grid computing is a form of distributed computing that involves sharing of resources that are heterogeneous and geographically distributed to solve various complex problems and develop large scale applications. In 1990's, the term 'Grid' was originated to denote a proposed distributed computing architecture for advanced science and engineering. In today's competitive environment the goals and objectives of the producers (also called resource owners) and consumers (also called end users) are different. Computational grid has been considered as the best paradigm for handling large scale systems that are distributed having geographically allocated resources. Task allocation algorithms are important in the research of network applications. In this paper we present an algorithm which reduces the average execution time and cost of the tasks. This method considers both time and cost constraints. The proposed algorithm is implemented with Gridsim toolkit which can simulate a decentralized module. The GridSim toolkit abstracts the behavior and features of complex fundamental grid elements such as grid tasks, grid resources and grid users. The algorithm provides services like resource management. For evaluation purpose a comparison of execution time and total cost of proposed algorithm and other similar algorithm is also given in this paper. Results support the proposed approach.

Keywords - Grid, heuristic task allocation algorithm, Gridsim, computation time.

I. INTRODUCTION

Grid computing was introduced by Poster and Kesselman. They explain it as a way to utilize the geographically distributed available idle workstation's computation power to remote grid users for the execution of their computation requiring jobs or tasks or processes. It involves the runtime aggregation of these resources in the form of virtual organization, [3, 8], according to the need of the jobs submitted by the grid users. The user essentially interacts with a resource broker that hides the complexities of Grid computing [4, 5]. The broker discovers resources that the user can access using information services, negotiates for access costs using trading services, maps tasks to resources (scheduling), stages the application and data for processing (deployment), starts job execution, and finally gathers the results. It is also responsible for tracking and monitoring application execution progress along with adaption of the changes in Grid runtime environment conditions and resource failures as the grid resource utilization is gaining significance various scheduling methods or task allocation across the grid is required in order to improve the performance of the grid system. While allocating the tasks, certain type of information such as the job arrival rate, CPU processing rate, number of jobs waiting in queue and so forth at each processor, as well as at neighbouring processors, may be exchanged among the processors for improving the overall performance. Based on the information that can be used, task allocation algorithms are classified as static, dynamic or adaptive [7, 10]. Static approach assumes that prior information about all the characteristics of the jobs, the computing devices and the communication network are known and provided. Task allocation decisions are made probabilistically or deterministically at compile time and remain constant during runtime. Dynamic load balancing algorithm attempts to use the runtime state information to make more informative decision in sharing the system load. However, dynamic scheme is used a lot in modern load balancing method due to their robustness and flexibility. Dynamic algorithm is characterised by parameters such as i) Centralized vs. Decentralized. An algorithm is centralized if the parameters necessary for making the load balancing decision are collected at, and used by, a single device. In decentralized approach all the devices are involved in making the load balancing decision. Decentralized algorithms are more scalable and have better fault tolerance. ii) Cooperative vs. Non-cooperative. An algorithm is said to be cooperative if the distributed components that constitute the system cooperate in the decision-making process. Otherwise, it is non-cooperative.and iii) Adaptive vs. Non-adaptive. If the parameters of the algorithm can change when the algorithm is being run, the algorithm is said to be adaptive (to the changes in the environment in which it is running). Otherwise, it is non-adaptive. The characteristic of grid makes resource scheduling even more challenging in the computational grid environment. In the paper we proposed a new task allocation algorithm which is heuristic in nature. Unlike previous algorithms which considered the system load of grid nodes or the completion time for a job but do not take into account job personal resource requirements (such as cost , QOS of a node) . In this paper we presented an algorithm which considers the job size and we mainly focussed on formulating a decentralized heuristic based load balancing algorithm for resource scheduling. The rest of the paper is organised as follows: Section II gives a detailed Literature review. Section III deals with the system model of the Grid. Section IV will describe the proposed algorithm concept and design. Section V will show the simulation experiment and results, and finally Section VI will conclude the whole paper.

II. RELATED WORK

Numerous researchers have proposed scheduling algorithms for parallel and distributed systems as well as Grid computing environment. For a dynamic load balancing algorithm, it is unacceptable to exchange state information frequently because of the high communication overhead. In order to reduce the communication overhead Martin et al. [18] studied the effect of communication latency, overhead and bandwidth in cluster architecture for application performance. Distributed network computing environments have become a cost effective and popular choice to achieve high performance and to solve large scale computation problems. There are many types of algorithms that have been used in resource balancing in Grid computing system. Martino and Mililotti addresses the use of Genetic Algorithm (GA) and Tabu Search (TS) to solve the grid load balancing problem in the dynamic environment [5]. These algorithms have their limitations like these algorithms require extra storage and processing requirement at the scheduling nodes. Grid computing involves coupled and coordinated use of geographically distributed resources for purposes such as large scale computation and distributed data analysis [17].
A new hybrid load balancing policy can be integrated in static and dynamic load balancing techniques with the objective to allocate effective node, identify the system imbalance immediately when a node becomes unable to participate in task processing. Ibarra and Kim in [15] used a combination of intelligent agents and multi-agent that work in grid load balancing area. In static grid load balancing, the iterative heuristic algorithm is better than the FCFS algorithm. Grid infrastructures are dynamic in nature in the sense of resource availability and hence a changing network topology. Resource heterogeneity and network heterogeneity also exits in Grid environment. Scheduling jobs onto resources is NP hard problem. So, we need an algorithm that consider optimization in terms of time and cost for efficient scheduling and execution. Current grid resource scheduling algorithms are mainly based on User-Directed Assignment (UDA), Minimum Completion Time (MCT), Minimum Execution Time (MET), Min-Min [5], Max-Min, heuristic algorithm, genetic algorithm (GA)[5], Ant Algorithm (AA) [6], multi-Agent, computational economy model [7]. In [19] authors proposed a dynamic load balancing algorithm which considers CPU length, CPU and memory utilization and network traffic as load metric.
Although [6][14] presented scheduling algorithms based on ant algorithm, but the authors just analyzed ant algorithm based classified task scheduling method and ACA based scheduling without considering about the price attribute. In this paper, price factor and time factor are taken into consideration, and the objective is to save total execution time and cost.

III. SYSTEM MODEL OF THE GRID

Grid system consists of sharing of heterogeneous resources (like different capacity processors, memory, networks etc) [2]. In order to maximize the performance of computational Grid environment, it is essential to distribute the load on different grid nodes. In grid computing environment, there exists more than one resource to process jobs. One of the main challenges is to find the best or optimal resources to process a particular job in term of minimizing the job computational time. Optimal resources refer to resources having high CPU speeds and large memory spaces. Computational time is a measure of how long that resource takes to complete the job. In order to maximize the performance of computational Grid environment, it is essential to distribute the load on different grid nodes. Following are the members of Grid system:-

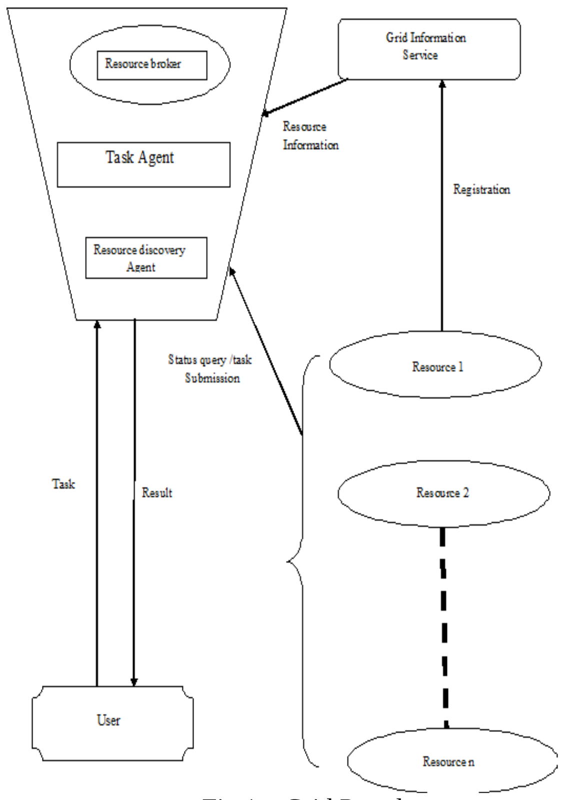

Fig.1. Grid Portal

Grid Resource Broker: It is an important entity of a grid. A grid resource broker is connected to an instance of the user. Each grid job (composed of gridlets or tasks) is first submitted to the broker, which then schedule the grid job according to the user scheduling policy.

GIS: Grid resource is a member of grid. All the available resources register themselves to the GIS (Grid information Service). GIS contains all information about all available resources with their computing capacity and cost at which they offer their services to grid users. All Grid resources that join and leave the grid are monitored by GIS. Whenever a Grid broker has jobs to execute, it consults GIS to give information about available grid resources.

Grid Use: They register themselves to the GIS by specifying the QOS requirement such as cost of computation, deadline to complete the execution, the number of processors, speed of processing of internal scheduling policy and time zone.

Task Agent: TA, by deploying Ant algorithm, select a resource for next task assignment and dispatch the task to selected resource through RMA. After task execution, results are received from resources and are returned to user by TA.

Resource discovery Agent: RDA queries GIS regarding resources. It send query to registered resources for their availability status and gets the status information and makes it available to TA. Every task is dispatched through RDA. Since in a Grid environment, the network topology is varying, our model captures this constraint as well by considering an arbitrary topology. The data transfer rate is not fixed and varies from link to link. The aim of this paper is to present task allocation algorithms adapted to the heterogeneous Grid computing environment. In this paper, we attempt to formulate an adaptive decentralized sender-initiated task allocation algorithm for computational Grid environments. Interaction among the entities is shown in the Fig.1.

IV. HEURISTIC TASK ALLOCATION ALGORITHM

An algorithm for task allocation is successful, if the task executes within the deadline. All the tasks in this algorithm are distributed according to Ant Colony Algorithm. HTTA is inspired on an analogy with real life behaviour of a colony of ants when looking for food, and is effective algorithm for the solution of many combinatorial optimization problems. Investigations show that: Ant has the ability of finding an optimal path from nest to food. On the way of ants moving, they lay some pheromone on the ground. 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 probability of ant chooses a way is proportion to the concentration of a way's pheromone. In HTTA, the pheromone is associated with resources rather than path. The increase or decrease of pheromone depends on task status at resources. The main objective of algorithm is reduction in total cost and execution time. The process of Heuristic based Task Allocation Algorithm is shown below:

Let the number of tasks (ants) in task set T maintained by task agent A and the number of registered resources (pheromone) is in set Q.

```
Step 1: [Initialize Pheromone Value of Each Resource]
For every new resource Ri registered to Grid Information Server, the initial pheromone Ip (0) is given as:
Ip(0) = [(N×M) + Cs ] / Rc
Step 2: [Select Task]
 Repeat 3 to 5 While (T ≠ Φ)
        Select task t from task set T.
Step 3: [Select Resource]
        Determine the resource Ri for task t having higher transitions probability P (higher pheromone intensity) Such as:
        [Initially Cpj = Ipj ]
```

$$P_j(t) = Max\left[\frac{[C_{pj}(t)^a] \times [I_{pj}(t)^b]}{(\sum_r [C_{pj}(t)]^a \times [I_{pj}\,(t)]^b) \times R_c}\right]$$

```
 (Eq-2)
        [ j, r are available resources.]
Step 4: [Schedule Task to Selected Resource]
   Schedule task t to Rj and remove it from T
                    T=T-{t}
Step 5: [Update Pheromone]
 For every resource Rj assigned a task pheromone is
        [1- Pd = Pheromone Permanence]
```

$$C_{pj}^{new} = P_d \times C_{pj}^{old} + \Delta_j$$

```
          (Eq-3)
 [Pheromone Variance Δ_j = −Complexity of task (C)]
     If (Task_Status = Succesful_Complete)
               Δ_j=φ×C                          (Eq-4)
        If (Task_Status = Failure)
                    Δ_j=Θ×C                          (Eq-5)
Step 6: EXIT.
```

In the given algorithm the Step 1 initialize the pheromone value associated with each resource which depends on the number of processing elements (N) a resource have, execution speed (M) in terms of millions of instruction per second, the communication speed (Cs) of the resources and inversely proportional to the cost (Rc) of using a resource as shown in Equation 1.Step 2 selects the tasks t from the task set T until the task set gets empty. In Step 3 transition probability is used to select the resource having the maximum probability or pheromone value as shown in Equation 2 .The probability (P) of a resource to get selected is depends on the current pheromone(Cp) value, historic information i.e. initial pheromone (Ip) value and inversely proportional to the cost of using that particular resource. Task is assigned a resource Rj (selected in Step 3) in Step 4. In Step 5 the value of the pheromone is updated according to Equation 3, if task returns successfully then pheromone value is increased by encouragement argument (Φ) otherwise it is decreased by punishment argument(Θ) according to the Equation 4 and Equation 5 respectively. Same algorithm is repeated for each task t in task set. For simulation of the above algorithm the parameters and their values are given in next section.

```
Phase 1 (Initialization Phase)
 Begin
Initialise all parameters including resources (processing elements, MIPS rating), pheromone intensity, Task set etc.
Phase 2 (Operational Phase)
While (Task set T != Φ)
        Begin
        Select the next task't' from Task set T.
        Determine next resource Ri for task assignment having high transition probability (or pheromone intensity).

Phase 3 (Result Phase)
         Schedule task 't'  to Ri and update Task set by T = T - {t}.
        If any node fails or complete its execution part update its pheromone intensity of that corresponding resource.
END While
END
```

All tasks are allocated to the resources depending on the value of pheromone or transition probability, which varies according to the status of the task at a particular resource. Resource having highest pheromone intensity would get the task before any other resource having lower than that intensity value. The algorithm is further elaborated by flowchart as given in Fig.2:

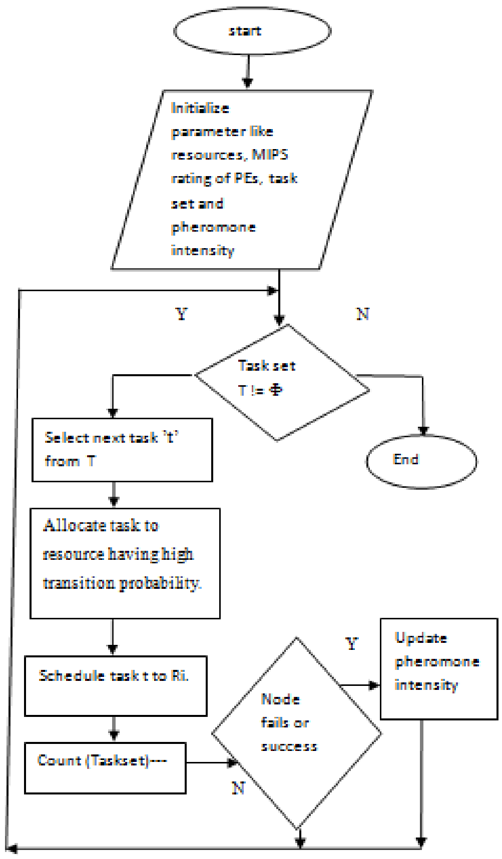
start
Initialize parameter like resources, MIPS rating of PEs, task set and pheromone intensity
Y
N
Task set T != Φ
Select next task 't' from T
End
Allocate task to resource having high transition probability.
Y
Update pheromone intensity
Schedule task t to Ri.
Node fails or success
Count (Taskset)---
N

Fig.2. Flowchart for Heuristic Load Balancing Algorithm

V. SIMULATION RESULTS

To compare the performance of the proposed approach and earlier developed approach simulation results are used. For simulation GridSim simulator is used which is a toolkit for the modelling and simulation of distributed resource management and scheduling for Grid computing [11]. In order to test the efficiency of the improved task allocation approach a comparison is done for the execution time (time utilized by PE for processing a task) of heuristic task allocation algorithm and non-heuristic load balancing algorithm.

Two experiments have been conducted to evaluate the performance of the proposed algorithm in terms of their processing time. First experiment processed 10-50 tasks with fixed number of resources taken as 25 while the second experiment processed 25 tasks with varying number of resources from 10-50.

Two experiments have been conducted to evaluate the performance of the proposed algorithm in terms of their execution cost. First experiment processed 10-50 tasks with fixed number of resources taken as 25 while the second experiment processed 25 tasks with varying number of resources from 10-50.

Details of the scheduling parameters are shown in the Table 1.

Table 1 Scheduling Parameters for the Experiments

Experiment	Ist	2nd
No. of machines per resource	25	10-50
No. of Processing Elements(PEs) per machine	10-50	25

Average computation time (in seconds) has been used to compare the performance of algorithms. Fig. 3 depicts the comparison between the average computation times of the algorithms for the first experiment. In this number of resources remain fixed as 25 and the number of tasks varied from 10 to 50. It can be seen that the proposed algorithm labelled as Heuristic based load task allocation algorithm has the lower execution time as compared to Non-Heuristic based task allocation algorithm at different instants.

A comparison between the computation times for the second experiment is shown in Fig. 4. In the second experiment the number of task remain fixed as 25 and number of resources varied from 10 to 50 and again compute the total execution time(in seconds) for different number of resources.

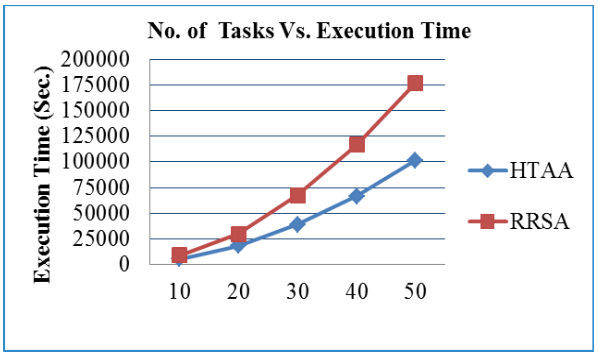

Fig. 3. Comparison Among the Execution Times of Heuristic and Non-Heuristic Algorithms.

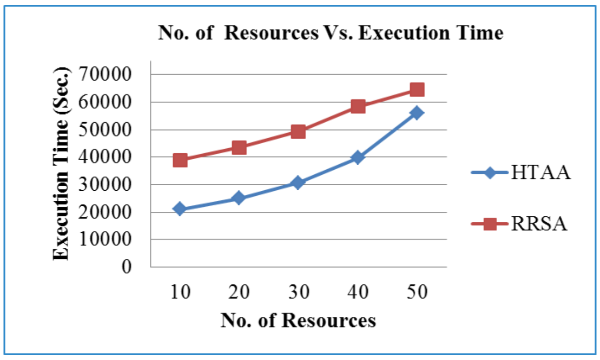

Fig. 4. Comparison Among the Execution times of Heuristic and Non-heuristic Load Balancing Algorithms for Experiment 2.

Fig. 3 and Fig. 4 depict the result of the total execution time for two different scenarios. Results support the proposed approach by reducing the total execution time of jobs.

A comparison between the two algorithms on the basis of cost can be seen in Fig. 5. In this number of resources remain fixed as 25 and the number of tasks varied from 10 to 50. In Fig. 6 execution cost is compared with varied number of resources 10 to 50 and fixed number of task 25. Heuristic task allocation algorithm still performs better than non-heuristic load balancing algorithm.

The performance of Heuristic approach is better because the communication that occur when each ant taking a step for searching the best resource is less when processing a large amount of jobs. Each ant will carry the history of visited node and will refer to it to find the best resource to process the job.

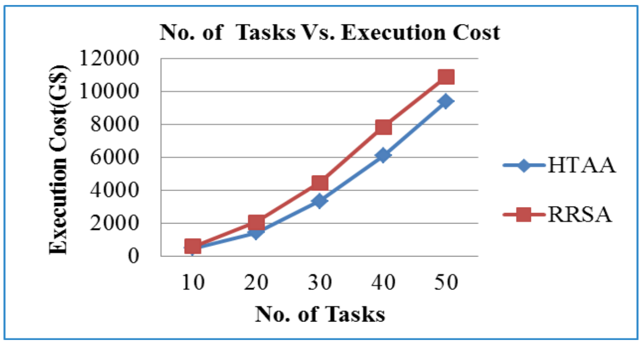

Fig. 5. Comparison Among the Execution Costs of Heuristic and Non-He uristic Algorithms.

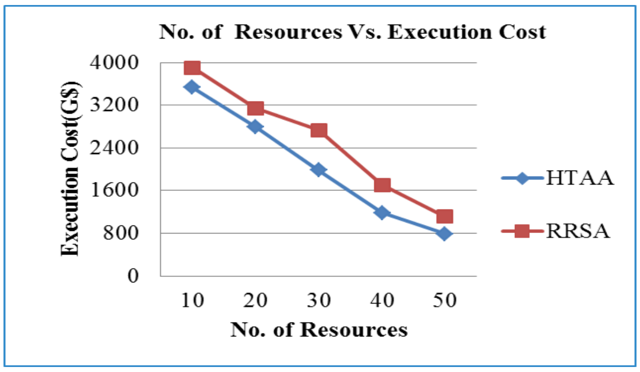

Fig. 6. Comparison Among the Execution Costs of Heuristic and Non-Heuristic Algorithms.

VI. CONCLUSION

In this paper an algorithm that is adaptive, decentralized and distributed for task allocation among the aggregated resources in the grid has proposed. Through a series of simulations by varying the number of tasks and resources we obtain the results. It is found out that computation decreases until it reaches the optimal level i.e resources with maximum computation capability. When the number of users competing for the same set of resources increases, there will be proportional impact on others depending on each user's strategies and constraints. Apart from complementing and strengthening the results of the proposed method, the simulations demonstrate the capability of GridSim and the ease with which it can be used for evaluating performance of new scheduling algorithms. Result of the simulation shows that HTTA enhances the performance of the system by reducing the execution time of the jobs.

REFRENCES

[1] Kumar Mishra, "An Agent Based Dynamic Resource Scheduling Model with FCFS-Job Grouping Strategy In Grid Computing", Waset, ICCGCS 2010.

[2] I. Foster, and C. Kesselman(editors), *The Grid 2: Blueprint for a New Computing Infrastructure*, Morgan Kaufmann, USA, 2003.

[3] I. Foster, C. Kesselman, and S. Tuecke, "The anatomy of the grid: enabling scalable virtual organizations", *International Journal of High Performance Computing Applications*, vol. 15, no. 3, pp. 200-222, 2001.

[4] X.S. He, X.H. Sun, and G.V. Laszewski, "QoS guided min-min heuristic for grid task scheduling", *Journal of Computer Science & Technology*, vol.18, no. 4, pp. 442-451, 2003.

[5] V.D. Martino, and M. Mililotti, "Scheduling in a grid computing environment using genetic algorithms", Proceedings of the 16th International Symposium on Parallel and Distributed Processing, pp. 235-239, 2002.

[6] Z.H. Xu, X.D. Hou, and J.Z. Sun, "Ant algorithm-based task scheduling in grid computing", Proceedings of 2003 IEEE Canadian Conference on Electrical and Computer Engineering, vol. 2, no. 3, pp. 1107-1110, 2003.

[7] R. Buyya, D. Abramson, J. Giddy, and H. Stockinger, "Economic models for resource management and scheduling in Grid computing", Concurrency and Computation: Practice and Experience, vol. 14, no. 2, pp. 1507-1542, 2002.

[8] R. Buyya, D. Abramson, and S. Venugopal, "The Grid Economy", Proceedings of the IEEE, 3, vol. 93, no. 3, pp. 698-714, 2005.

[9] O.O. Sonmez, and A. Gursoy, "A novel economic-based scheduling heuristic for computational grids", *International Journal of High Performance Computing Applications*, vol.21, no. 1, pp. 21-29, 2007.

[10] R. Buyya, and Manzur Murshed, "GridSim: A Toolkit for the Modeling, and Simulation of Distributed Resource Management, and Scheduling for Grid Computing", Concurrency and Computation: Practice and Experience, vol. 14, no. 1, pp. 1175-1220, 2002.

[11] Li Hao,"Implement of Computational Grid Application Scheduling Simulation with GridSim Toolkit", *Journal of Jilin Normal University* (Nature Science Edition), vol. 3, no. 1, pp. 63-64, 2003.

[12] Wang yue, and Tao Hongjiu, ”Application in TSP Based on Ant Colony Optimization”, *Journal of Wuhan University of Technology* (Informationand Managing Engineering Edition),vol. 28, no. 11, pp. 24-26, 2006.

[13] Marco Dorigo, and Luca Maria Gambardella, ”Ant colonies for the traveling salesman problem”, BioSystems, vol. 43, no. 2, pp. 73-81, 1997.

[14] Chen Ye, ”Ant Colony System for Continuous Function Optimization”, Journal of Sichuan University (Engineering Science Edition), vol. 36, no. 4, pp. 117-120, 2004.

[15] Quan Liu, Yeqing Liao, “Grouping-Based Fine-grained Job Scheduling Grid Computing”, Vol.1, pp. 556-559, IEEE First International Workshop Education Technology and Computer Science, 2009.

[16] C. Sosa, and A.S. Grimshaw, ”Bringing the Grid Home”, Proceedings of 9th IEEE/ACM International Conference on Grid Computing, pp. 152-159, 2008.

[17] Casey, L.M., ”Decentralized Scheduling”, *The Australian Computer Journal*, vol. 13, no. 2, pp. 58- 63, 2010.

[18] R. Martin, A.Vahdat, D. Culler, and T. Anderson, ”Effects of Communication Latency, Overhead, and Bandwidth in a Cluster Architecture”. Proceedings of 24th Annual International Symposium Computer Architecture (ISCA ’97), pp. 85-97, 2009.

[19] Xu Zhihong, and Gu Junhua, ”Research on Ant Algorithm Based Classified Task Scheduling in Grid Computing”, Journal of Hebei University of Technology, vol. 35, no. 3, pp. 68-71, 2006.