Resource Management in Grid Computing Using Enhanced Ant Colony Optimization

Ku Ruhana Ku-Mahamud[1], Aniza Mohamed Din[2], Husna Jamal Abdul Nasir[3]
College of Arts and Sciences, Universiti Utara Malaysia
ruhana@uum.edu.my[1], anizamd@uum.edu.my[2], husna.jamanas@gmail.com[3]

ABSTRACT

Efficient resource management is needed to overcome stagnation problem in grid computing. Scheduling of jobs is one of the activities of resource management. This activity is complicated due to the distributed and heterogeneous nature of the resources. An enhanced ant colony optimization algorithm for job and resource scheduling is proposed in this paper. The proposed algorithm focuses on global pheromone update and the use of grid resource table to store all information about jobs, resources and pheromone value. Simulation approach has been used to test the performance of the algorithm. The credibility of the proposed algorithm is compared with other approaches and results produced showed that the algorithm can balance the load of the resources.

Keywords: Grid computing, resource management, job scheduling, ant colony algorithm.

Introduction

Computational grids are emerging as the new generation computing paradigm for tackling large scale problems in a wide range of scientific fields. It is a collection of geographically distributed, loosely coupled, heterogeneous, non-dedicated computing resources and belonging to several organizations. Computational grid provides computing services without users know the location and features of the involved resources (Foster and Kesselman, 1999).

Foster and Kesselman (1999) define that cluster and grid computing are several ways for establishing distributed system. Distributed system consists of multiple computers that communicate through computer networks. Several personal computers or workstation in cluster computing are combined through local networks in order to develop distributed applications. In cluster computing, applications are being inflexible in variation because they are limited to a fixed area. From this disadvantage, grid computing has been proposed to solve this problem. Grid computing is based on large-scale resources sharing in a widely connected network such as the Internet (Yan et al., 2009). This makes grid computing different from conventional distributed computing and cluster computing.

In grid computing environment, managing resource is a critical task since the resources are distributed and heterogeneous. Resource management is the process of managing available jobs and grid resource accordingly. This process includes the resources that are allocated, assigned, authorized, assured, and authenticated to process the request jobs (Sharma & Bawa, 2008). In grid environment, resource management includes resource discovery, resource monitoring, resource inventory, resource provisioning, fault isolation, autonomic capabilities and service level management activities. The optimization and the coordination of resource usage is a key issue in grid environment. This can be achieved by having an efficient grid scheduling policy.

Grid scheduling systems can be classified into centralized, hierarchical and decentralized (Attanasio et al., 2005). In a centralized controller scheme, it is assumed that a single machine controls the scheduling policies for all the nodes. In a hierarchical system, the resource controllers are organized

as a hierarchy in which higher resource controllers manage units of resources while lower level controllers schedule smaller units of resources. Ant colony algorithms have been applied in scheduling the jobs in the grid environment (Fidanova and Durchova, 2006). ACO algorithm is used in grid computing because it is easily adapted to solve both static and dynamic combinatorial optimization problems.

ACO is inspired by a colony of ants that work together in foraging behavior. This behavior encouraged ants to find the shortest path between their nest and food source. Every ant will deposit a chemical substance called pheromone on the ground after they move from the nest to food sources and vice versa. Therefore, they will choose an optimal path based on the pheromone value. The path with high pheromone value is shorter than the path with low pheromone value. This behavior is the basis for a cooperative communication.

ACO has been applied in solving many problems in scheduling such as Job Shop Problem, Open Shop Problem, Permutation Flow Shop Problem, Single Machine Total Tardiness Problem, Single Machine Total Weighted Tardiness Problem, Resource Constraints Project Scheduling Problem, Group Shop Problem and Single Machine Total Tardiness Problem with Sequence Dependent Setup Times (Dorigo and Stutzle, 2004).

This paper presents a static job scheduling mechanism in the grid environment. Section 2 describes the use of ant colony optimization algorithms in grid computing while the proposed resource management is discussed in Section 3. The experimental results are presented in Section 4 and concluding remarks are highlighted in Section 5.

Ant Algorithm in Grid Environment

There are various types of ACO algorithm such as Ant Colony System (ACS), Max-Min Ant System (MMAS), Rank-Based Ant System (RAS) and Elitist Ant System (EAS) (Dorigo and Stutzle, 2004). A recent approach of ACO researches in the use of ACO for resource management in grid computing.

An ant colony optimization for dynamic job scheduling in grid environment was proposed by Lorpunmanee et al. (2007) which aimed to minimize the total job tardiness time. The initial pheromone value of each resource is based on expected execution time and actual execution time of each job. The process to update the pheromone value on each resource is based on local update and global update rules as in ACS. In the study, ACO algorithm performed the best when compared to First Come First Serve, Minimal Tardiness Earliest Due Date and Minimal Tardiness Earliest Release Date techniques.

The study by Li (2006) proposed a bio-inspired adaptive job scheduling mechanism in grid computing. The purpose of the research is to minimize the execution time of the computational jobs by effectively taking advantage of the large amount of distributed resource. Various software ant agents were designed with simple functionalities. The pheromone value of each resource depends on their execution time. Resource with high execution time will receive a large number of pheromone. In this research, the comparison was also performed between the bio inspired adaptive scheduling with the random mechanism and heuristic mechanism. Experimental results showed that a bio-inspired adaptive job scheduling has good adaptability and robustness in a dynamic computational grid.

Simple grid simulation architecture for resource management and task scheduling was proposed by Xu et al. (2003). The study also validated the scalability of ant algorithm. The ant algorithm for grid

task scheduling is integrated into the simulation architecture and good results were obtained in terms of resource average utilization, response time and task fulfill proportion.

Proposed Ant Scheduling Algorithm

The proposed enhanced ant algorithm (Eant) takes into consideration the processor speed of the resources and the characteristics of jobs in determining the best resource to process a job. This is different from the approach by Moallem and Ludwig (2009) which proposed AntZ algorithm that did not consider the characteristics of jobs and the current load of each resource during the scheduling process.

Eant technique selects the resources based on the pheromone value on each resource which is recorded in a matrix. This technique has been implemented in the grid system architecture and consists of four main components namely the grid information server, grid resource broker, jobs and resources. The technique works as follows:

1. User will send request to process a job. Details about the job such as the total number of jobs, size of each job, and CPU time needed by jobs will be included in the request.
2. Grid resource broker starts to calculate the relevant parameter to schedule the job after receiving the message from the user. The information server also provides the resource information to grid resource broker.
3. The largest entry in the pheromone value (PV) matrix will be selected by proposed technique as the resource to process the submitted job. A local pheromone update is performed after a job is assigned to a resource.
4. A global pheromone update is performed after a resource completed processing a job.
5. The execution results will be sent to the user.

In this proposed technique, an ant represents a job in the grid system. The grid resource broker is an intelligent agent, will find available resources from grid information server. Ant will move randomly in grid system and check the status of each resource. Pheromone value on a resource indicates the capacity of each resource in grid system. Pheromone value will be determined by two types of pheromone update technique which are local pheromone update in ACS (Dorigo and Stutzle, 2004) and global pheromone update in MMAS (Dorigo, 1992).

The initial pheromone value of each resource for each job is calculated based on the estimated transmission time and execution time of a given job when assigned to this resource. The estimated transmission time can be determined by $\frac{S_j}{bandwidth_r}$ where S_j is the size of a given job *j* and $bandwidth_r$ is the bandwidth available between the grid resource broker and the resource. The initial pheromone value is defined by:

$$PV_{rj} = \left[\frac{S_j}{bandwidth_r} + \frac{C_j}{MIPS_r * (1 - load_r)} \right]^{-1}$$

where PV_{ij} is the pheromone value for job j assigned to resource r, C_j is the CPU time needed of job j, $MIPS_r$ is the processor speed of resource r and $1 - load$ is the current load of resource r. The load, processor speed and bandwidth can be obtained from grid information server.

Assume that there are *n* jobs and *m* resources in the PV matrix as shown below:

$$PV = \begin{array}{c} \\ r_1 \\ r_2 \\ .. \\ r_m \end{array} \begin{array}{c} \begin{array}{cccc} j_1 & j_2 & .. & j_n \end{array} \\ \begin{bmatrix} PV_{11} & PV_{12} & .. & PV_{1n} \\ .. & .. & .. & .. \\ .. & .. & .. & .. \\ PV_{m1} & PV_{m2} & .. & PV_{mn} \end{bmatrix} \end{array}$$

In each iteration, the largest entry from PV matrix will be selected. Assuming PVij is selected then job j will be processed by resource r. The local pheromone update is performed after job j has been assigned to resource r. This formula only applied to unassigned jobs in the PV matrix. The local pheromone update is formulated as follow:

$$PV_{rj} = (1-\xi).\tau_{jr} + \xi.\tau_0$$

where ξ,0< ξ<1 and τ_0 are two parameters. The value of τ_0 is set to be the same as the initial value for the pheromone trails. A good value for ξ was found to be 0.1, while a good value for τ_0 was found to be $1/nC^{nn}$, where n is the number of resources and C^{nn} is the resource with high pheromone value. The effect of the local pheromone update is to make an already chosen resource less desirable for the following ant (Gambardella and Dorigo, 1996). So, the exploration of not yet visited resource is increased.

When a job is completely processed, global pheromone update is performed to recalculate the entire *PV* matrix. After all ants have constructed a solution, the pheromone trails are updated according to the following formula:

$$PV_{rj}(t+1) = (1-\rho).\tau_{jr} + \rho\Delta\tau_{jr}^{bs}$$

where Δ τjrbest = 1/Lbest . The ant which is allowed to add pheromone may be the iteration-best solution or global best solution. If a specific resource is often used in the best solution, it will receive a larger amount of pheromone and stagnation will occur. So, lower and upper limits of the possible pheromone strengths on any resource are imposed to avoid stagnation. The imposed trails limits have the effects of limiting the probability ρiu of selecting resource u when ants is in resource i to an interval [pmin,pmax], with 0<pmin≤pij≤pmax≤1. With this minimum trail limit, the resource is less desirable to be selected by the jobs since it will select the resource that has the upper trail limit.

Table 1: Scheduling Parameters for the Experiments

Experiment	1	2	3	4	5	6	7	8	9	10
No of machine per resource	1	1	1	1	1	1	1	1	1	1
Number of PEs per machine	1 – 5	1 – 5	1 – 5	1 – 5	1 – 5	1 – 5	1 – 5	1 – 5	1 – 5	1 – 5
PE ratings (MIPS)	10 / 50	10 / 50	10 / 50	10 / 50	10 / 50	10 / 50	10 / 50	10 / 50	10 / 50	10 / 50
Bandwidth B/S	1000 / 5000	1000 / 5000	1000 / 5000	1000 / 5000	1000 / 5000	1000 / 5000	1000 / 5000	1000 / 5000	1000 / 5000	1000 / 5000

Experimental Results

The performance of the Eant in terms of resource utilization has been compared with the AntZ (Moallem and Ludwig, 2009) algorithm. The details of the parameters used in comparing the utilization of each resource are shown in Table 1(previous page). Each resource has a random number of PEs ranging between 1 and 5 and has different processing power. Jobs that have been submitted to the grid system are supposed to be independent of each other.

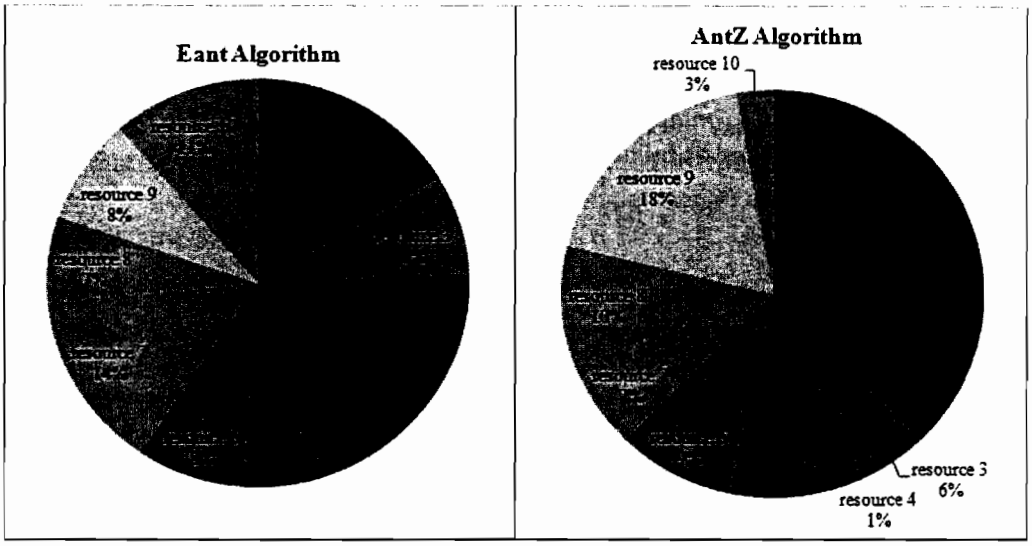

Figure 1: Resource utilization for Eant and AntZ

Figure 1 show the comparison of the utilization of each resource for both algorithms when 100 jobs and 10 resources were used in the experiment. It can be seen that Eant recorded a balanced utilization as compared to AntZ. The difference between the highest and lowest utilization of resources for Eant is 9 while the difference for Ant Z is 23. This is expected because Eant keeps track of the state of all the resources at all time. Eant perform better than AntZ because it considers the local and global pheromone updates which help to update the current state of the resource.

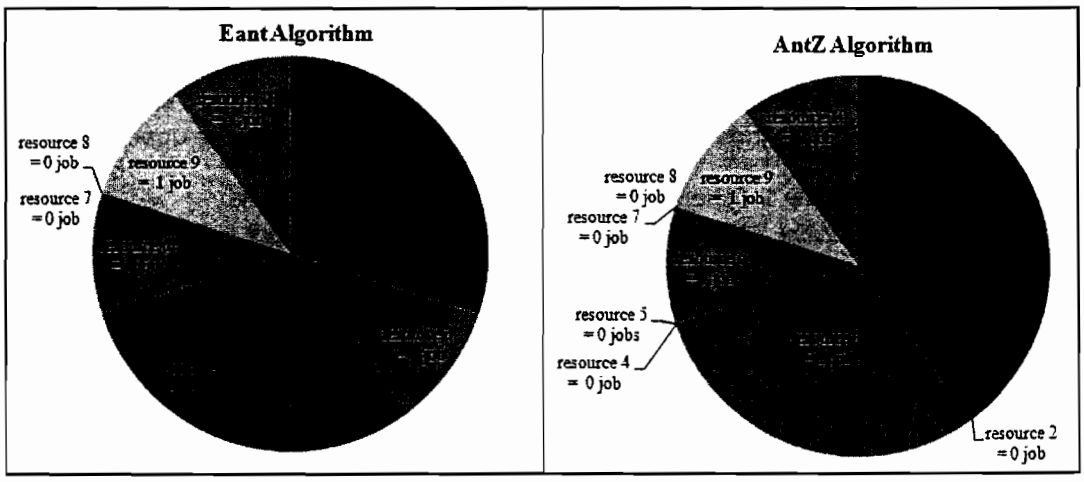

Figure 2: Number of processed jobs for each resource

Figure 2 shows the comparison of the number of jobs that were processed by each resource. A total of 10 jobs and 10 resources were used in this experiment. It can be seen that from 10 resources that available to process the jobs, only 2 have been idle in Eant algorithm. On the other hand, 5 from 10 resources are idle when AntZ algorithm was applied in scheduling the jobs. This will lead to an increased in the total time to process all the jobs. Eant performs better than AntZ because the algorithm will always try to distribute the jobs among the available resources.

Conclusion

The enhanced ant colony optimization algorithm for job and resource scheduling proposed in this paper was able to balance the load on of the resources thus minimizing the total processing time of the jobs. The proposed algorithm focuses on global pheromone update and the use of grid resource table to store all information about jobs, resources and pheromone value. This is a technique to control the value of pheromone updated on each resource. The local pheromone trail update will reduce the amount of pheromone in visited resource, so the resource they has visited is less desirable for other ants while the trail limit, which is the allowed range of the pheromone strength, is limited to maximum and minimum trail strength.

References

[1]. *Attanasio, A., Ghiani, G., Granddinetti, L., Guerriero, E. and Guerriero, F. (2005). Operations research methods for resource management and scheduling in a computational grid: A survey. Grid computing: The New Frontier of High Performance Computing. Lucio Grandinetti (Editor), 53-81.*

[2]. *Dorigo, M. (1992). Optimization, learning and natural algorithms, Ph. D. Dissertation, Politecnico di Milano, Milan, Italy, 1992.*

[3]. *Dorigo M and Stützle, T. (2004). Ant colony optimization. Cambridge, Massachusetts, London, England: MIT Press.*

[4]. *Fidanova, S. and Durchova, M. (2006). Ant algorithm for grid scheduling problem, Lecture Notes in Computer Science, 3743, 405-412.*

[5]. *Foster, I. and Kesselmeon, C. (editors) (1999). The grid: Blueprint for a future computing infrastructure. Son Francisco, USA: Morgon Kaufmann Publishers.*

[6]. *Gambardella, L. ond Dorigo, M. (1996). Solving symmetric and asymmetric TSPs by ant colonies. Proceedings of the IEEE Conference on Evolutionary Computation, Nagoya, Japan, 622-627.*

[7]. *Li, Y. (2006). A bio-inspired adaptive job scheduling mechanism on a computational grid, International Journal of Computer Science and Network Security (IJCSNS), 6(3), 1-7.*

[8]. *Lorpunmanee, S., Sap, M., Abdullah, A. ond Chompoo-inwai, C. (2007). An ant colony optimization for dynamic job scheduling in grid environment, International Journal of Computer and Information Science and Engineering, 1(4), 207-214.*

[9]. *Moallem, A. and Ludwig, S.A. (2009). Using ortificial life techniques for distributed grid job scheduling. Proceedings of the ACM Symposium on Applied Computing, Hawaii, U.S.A., 1091 – 1097.*

[10]. *Sharma, A. and Bawa, S. (2008). Comporative analysis of resource discovery approaches in grid computing. Journal of Computers, 3(5). 60-64.*

[11]. *Xu, Z., Hou, X. and Sun, J. (2003). Ant algorithm-based task scheduling in grid computing, Proceedings of the IEEE Conference on Electrical and Computer Engineering, Canada, 1107-1110.*

[12]. *Yan, K.Q., Wang, S.S., Wang, S.C. and Chang, C.P. (2009). Towords a hybrid load balancing policy in grid computing system. Expert Systems with Applications, 36(10), 12054-12064.*