ENHANCED ANT ALGORITHM BASED LOAD BALANCED TASK SCHEDULING IN GRID COMPUTING

Kuppani Sathish , A Rama Mohan Reddy

[1]*Department of Computer Science and Engineering, Sree Rama Engineering College, Tirupati*
[2]*Department of Computer Science and Engineering, Sri Venkateswara University College of Engineering, Tirupati*

Abstract
Load balanced task scheduling is very important problem in complex grid environment. Finding optimal schedules for such an environment is an NP-hard problem, and so heuristic approaches must be used. Antalgorithm is a heuristic task scheduling algorithm which is distributable, scalable and fault tolerant. It uses the state prediction of the resources for scheduling which is necessary for effective utilization of resources. In this paper, we propose an enhanced ant-algorithm for task scheduling in grid which gives better throughput with a controlled cost. The simulation results of various scheduling algorithms are also compared. The results also show that the enhanced version works better than the ant-algorithm. The inclusion of price factor into the ant-algorithm makes this new scheduling algorithm more suitable for wide use.

1. Introduction

In the last decade, even though the computational capability and network performance have gone to a great extent, there are still problems in the fields of science, engineering, and business, which cannot be effectively dealt with using the current generation of supercomputers. In fact, due to their size and complexity, these problems are often resource (computational and data) intensive and consequently entail the use of a variety of heterogeneous resources that are not available in a single organization. Grid is an aggregation of a wide variety of resources that may include super computers, storage systems, data sources, and specialized devices that are geographically distributed and owned by different organizations. The technology used for solving large-scale computational and data intensive problems in science, engineering, and commerce [1] using the Grid is Grid computing. The Grid system is responsible for the execution of the jobs submitted to it. The advanced Grid systems would include a task scheduler which automatically finds the most appropriate machine on which a given job is to run.

This resource selection [2] is very important in reducing the total execution time and cost of executing the tasks which depends on the task scheduling algorithm. The scheduling policy followed by the scheduler determines the Grid system throughput and utilization of the resources in the grid. The goal of the scheduler is to maximize the system throughput and utilization of the resources by balancing the loads on them.

This paper presents and evaluates a dynamic scheduling strategy that maximizes the utilization of Grid resource processing capabilities, and reduces the processing cost and processing time taken to execute the jobs on the Grid. The proposed job scheduling strategy takes into account: *(i)* the processing requirements for each job, *(ii)* the current state of the available resources, *(iii)* the current load and capacity of those resources, and *(iv)* the processing cost of those resources. In order to evaluate the proposed job scheduler, GridSim toolkit, as discussed in Buyya and Murshed (2002), is used to model and simulate grid resources, grid users and job scheduling.

The rest of this paper is organized as follows: Section 2 briefly discusses the ant algorithm, whereas Section 3 presents the enhancement to the ant algorithm and Section 4 presents the proposed task scheduling algorithm. Some simulations and experiments were conducted on the proposed scheduling algorithm using GridSim toolkit and the results are presented in Section 5. Finally, Section 6 concludes the paper and mentions some future work.

2. Ant Algorithm

Ant algorithm[6] is a new heuristic, predictive scheduling algorithm which is distributable, scalable and fault tolerant[3,4,5]; it is based on the behavior of real ants. The algorithm has been successfully used to solve many NP problems, such as TSP, assignment problem, job-shop scheduling and graph coloring. The algorithm has inherent parallelism, and we can validate its scalability. So, it's obvious that ant algorithm is suitable to be used in Grid computing task scheduling.

3. Enhancement to the basic task scheduling ant algorithm

The scheduler schedules a task based on the possibilities [$_k(t)$ $_j$ ρ] of the resources. The problem with this

Manuscript received October 5, 2008
Manuscript revised October 20, 2008

algorithm is, it may schedule a task to a resource with low possibility even if the resources with high possibility are free. To avoid this, 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 waiting for the resource to be free even though the other resources are free. So, the algorithm can be modified in such a way that if the difference between the possibility of the resource selected for execution of a task using ant-algorithm and the possibility of the resource with highest possibility is less than certain threshold, then the task will be scheduled to the resource selected by the ant-algorithm. Otherwise, the scheduler selects another resource and the above procedure will be repeated. The selection of the threshold plays an important role. Since this modified algorithm takes the resources with highest possibility into consideration, although the processing time is reduced, the processing cost of the tasks may increase when compared to that of ant-algorithm. The inclusion of price factor into this modified algorithm which is the proposed algorithm minimizes the total execution time as well as the processing cost of the tasks.

4. Proposed task scheduling algorithm

1. When a resource 'j' enrolls the Grid, it is asked to submit its performance parameters, No. of PE, MIPS of every PE etc. Resource monitor tests these parameters for validation, and initialize the links pheromone as:

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

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

2. Every time a new resource joins the grid, a resource gets failed, a task is assigned, or there is some task returned, the pheromone on path from schedule centre to the corresponding resource will be changed as

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

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

$\Delta\tau_j$ = - k , k is the compute and transfer quality of the task.

when task successfully returned from resource 'j',

$\Delta\tau_j$ = Ce x k, C_e is the encourage factor.

when task failed returned from resource 'j',

$\Delta\tau_j$ = Cp x k, C_p is the punish factor.

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

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

= 0 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, that is $\tau_j(0)$.
α is the importance of the pheromone.
β is the importance of resource innate attributes.
The parameter values $\alpha = 0.5, \beta = 0.5, \rho = 0.8$,
Ce = 1.1, Cp = 0.8 will be taken for here.
This gives the probabilities of the resources in the grid which helps the task scheduler in scheduling.

4. 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 $(\rho_h^k(t) - \rho_j^k(t)) \leq T_h$.

where 'h' is the resource with highest $\rho^k(t)$.

'T_h' is the threshold which will be taken as 1/ (no.of resources) .

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

where C_i is the cost of the resource 'i' per second
$MIPS_i$ is the total MIPS of the resource 'i'

$T_i = T_h - (\rho_h^k(t) - \rho_j^k(t))$ and

'ℓ' is the cost reduction factor which is selected by the grid user who submits the task. $0 \leq \ell \leq 1$.

6. The scheduler schedules the new task 'k' on to the resource 'i'.

5. Evaluation

5.1 Implementation with GridSim

GridSim[7] toolkit is used to conduct the simulations based on the developed scheduling algorithm. Grid resources information and Grid users information will be given as input to start the simulation. Then the simulation starts by resource creation followed by the creation of user tasks. The user tasks are created based on user specified parameters (total number of jobs, average MI of each job, and the deviation percentage of the MI). Then, the system starts the GridSim simulation. It first gathers the characteristics of the available Grid resources created in the resource creation section in the system. The scheduler maintains the pheromone value of each resource (from schedule center) and will compute the possibilities of the current resources available in the Grid every time when a job is to be scheduled. Then, the scheduler selects a resource based on the developed algorithm and schedules the job. When a job is submitted to a resource it's pheromone value will be decreased by the compute and transfer quality of the task.. The Grid resources process the received jobs and send back the processed jobs to the grid user.

The system then gathers the processed jobs sent back to the user. If the job is successfully is executed then, the pheromone value of the resource on which this job is submitted will be increased. If the job execution fails, the pheromone of the resource will be decreased. The simulation will be completed when all the user tasks get executed on the grid resources and get back the results.

5.2 Experiments, Results and discussions

Simulations are conducted to analyze and compare the differences between various scheduling algorithms, in terms of processing times and processing costs. Resources R1 through R20 are used for these simulations. Users U1 to U40 submit tasks to the grid. The resource and user tasks characteristics taken for the simulation are shown in the table 5.1 and table 5.2 respectively.

Resource	No. of PEs	MIPS of PE	Communication rate (Mb/s)	Cost/sec
R1	1	50	10	1
R2	4	377	30	2
R3	2	380	10	1
R4	16	410	40	7
R5	4	410	20	2
R6	2	200	20	2
R7	6	410	20	3
R8	16	410	50	7
R9	4	377	20	2
R10	16	410	50	7
R11	1	50	10	1
R12	4	377	30	2
R13	2	380	10	1
R14	16	410	40	7
R15	4	410	20	2
R16	2	200	20	2
R17	6	410	20	3
R18	16	410	50	7
R19	4	377	20	2
R20	16	410	50	7

Table 5.1: Resource Task Characteristics

User	Job length (MI)	Job size (Mb)	Output size (Mb)
U1&21	75000	100	50
U2&22	65000	80	40
U3&23	8000	15	10
U4&24	5000	15	10
U5&25	70000	80	35
U6&26	75000	85	50
U7&27	200	10	10
U8&28	45000	90	40
U9&29	3000	30	10
U10&30	100	10	10
U11&31	70000	95	50
U12&32	2200	50	20
U13&33	500	15	10
U14&34	75000	100	50
U15&35	32000	40	25
U16&36	68000	80	45
U17&37	7000	20	10
U18&38	300	10	10
U19&39	1000	10	10
U20&40	54000	60	40

Table 5.2 User task characteristics

The following are the results of simulation of various algorithms for the 40 grid users as the number of resources in the grid increases. Adding of resources to the grid is taken in the same order as mentioned in the input resource characteristics above. Here the ℓ of all the user tasks is taken as 1.

5.2.1 Experiment 1: Simulation results of various scheduling strategies

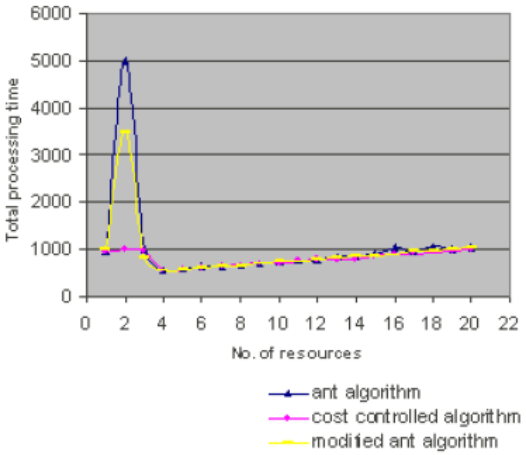

Fig 5.1 No. of resources Vs total processing time

As shown in the Fig 5.1, the processing time in the modified ant-algorithm (up to step 4 in the algorithm given in Section 4) and the developed algorithm which is inclusive of price factor is reduced when compared to that of ant algorithm.

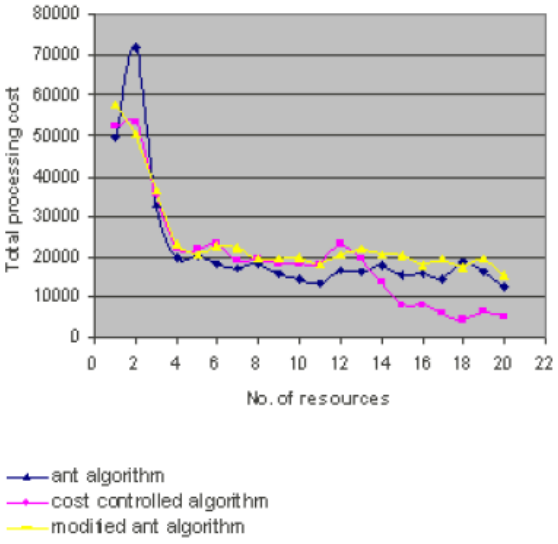

Fig 5.2 No. of resources Vs total processing cost

As shown in the Fig 5.2, the processing costs in the developed algorithm is lot more controlled when compared to that of modified ant-algorithm in which the price factor is not included.

5.2.2 Experiment 2: Simulation results for different threshold values in the algorithm

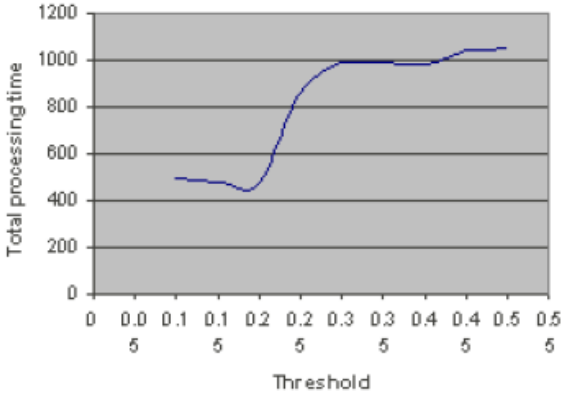

Fig 5.3 Threshold Vs total processing time

Fig 5.3 shows the total processing times for different values of threshold in the scheduling algorithm. If the threshold value increases certain value, then the algorithm results will be similar to that of ant-algorithm.

6. Conclusion and future work

The proposed scheduling strategy results in increased performance in terms of low processing time and low processing cost if it is applied to a Grid application with a large number of coarse granularity tasks like parameter sweeps application. This works effectively in minimizing both the processing time of the tasks as well as the processing cost of the tasks depending upon the CRF value chosen by the grid user who submits the task. Future work would involve developing a more comprehensive distributive scheduling system that takes into account the hierarchy of clusters of resources. And also to reduce the transmission costs and delays in the applications with large number of light weight jobs, job grouping can be done before load balancing the jobs on to the resources.

Acknowledgements

The authors wish to thank Mannem Rami Reddy, Chairman, Sree Rama Engineering College, Tirupati, and also to the authorities of Sri Venkateswara University, Tirupati.

References

[1] Mark Baker, Rajkumar Buyya, and Domenico Laforenza, "Grids and Grid Technologies for Wide-Area Distributed Computing", *Software: Practice and Experience (SPE) Journal*, Wiley Press, USA, 2002.

[2] Klaus Krauter, Rajkumar Buyya, and Muthucumaru Maheswaran, "A Taxonomy and survey of Grid Resource Management Systems for Distributed Computing", *Software: Practice and Experience (SPE) journal*, Wiley Press, USA, 2001.

[3] Quinn Snell, Kevin Tew, Joseph Ekstrom, Mark Clement, "An Enterprise-Based Grid Resource Management System", *Proceedings of the 11th IEEE International Symposium on High performance Distributed computing HPDC-11* 2002 (HPDC'02).

[4] F. De Turck, S. Vanhaste, B. Volckaert, P. Demeester, "A generic middleware-based platform for scalable cluster computing", *Future Generation Computer Systems*, vol. 18, 2002, pp.549-560.

[5] Ron Oldfield, David Kotzl, "Armada: a parallel I/O framework for computational grids", *Future Generation Computer Systems*, vol 18, 2002, pp.501-523.

[6] Xhihong Xu, Xiangdan Hou, Jizhou Sun, "Ant Algorithm based task scheduling in Grid Computing", *Proceedings of the IEEE Canadian conference on Electrical and Computer Engineering*, vol. 2, 2003, pp. 1107-1110, Montreal, May 2003.

[7] Rajkumar Buyya, and Manzur Murshed, *GridSim: A Toolkit for the Modeling, and Simulation of Distributed Resource Management, and Scheduling for Grid Computing***,** The Journal of Concurrency, and Computation: Practice, and Experience (CCPE), Volume 14, Issue 13-15, Pages: 1175-1220, Wiley Press, USA, November - December 2002.

[8] Huiyan, Xue-qinshen, Xing li,Ming-Hui Wu, " *an improved ant algorithm for job scheduling in Grid computing*", Proceedings of the Fourth International Conference on Machine Learning and Cybernetics, Guangzhou,18-21 august 2005.

About Authors

Kuppani Sathish received Bachelor's degree in Electronics and Communication Engineering from Bangalore University in 1997 and Master's of Technology degree in Computer Science Engineering and at present he is working as an Associate Professor in Sri Rama Engineering College which is affiliated to Jawaharlal Nehru Technological University, Ananthpur. He is having eleven years of experience in teaching and research and taught for bachelor and master degree courses. His area of interest includes Network Systems, Software Architecture and Communication Systems. He made significant contributions in the area of Networking in Rayalaseema region to provide internet access to common man. He attended many conferences and workshops and communicated papers to reputed National and International journals.

A Rama Mohan Reddy, obtained his Bachelor Degree in Mechanical Engineering and Master's degree in Computer Science Engineering and Ph.D degree from Sri Venkateswara University and at present working as an Associate Professor in Department of Computer Science and Engineering, Sri Venkateswara University College of Engineering, Sri Venkateswara University, Tirupati. His areas of interest include Software Architecture and Computer networking, data mining, and other latest trends in technology. He has more than 14 years of experience in teaching and research in the area of Computer Science and Engineering.