Volume 3, Issue 7, July 2013 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Implementation of Load Balancing Algorithm for Grid by Heuristic Approach

Kanika Kaushal[1], Jaidev Gurjar[2],
[1.2]Research Scholar
Department of Computer Science UIET ,
KUK, India

Anu Rawal[3]
[3]Assistant Professor
Department of Computer Science SKIET,
Kurukshetra, Haryana, India

Abstract-The aim of grid computing is to promote the development and advancement of technologies that provide seamless and scalable access to wide-area distributed resources. Computational grid has been considered as the best paradigm for managing very large scale system which is distributed geographically and having allocated resources world wide. Load balancing algorithms are very big issues in the development of network applications. In this thesis we present an algorithm which reduces the task average execution time and cost of the tasks. The methods presented in this thesis include both time and cost factors. In this thesis we use Gridsim simulator for simulation of the decentralized modules. Gridsim allows modeling and simulation of entities in parallel and distributed computing systems such as users, applications, resources, and resource. The algorithm presented in this thesis gives resource discovery service. For determining the advantages of this algorithm we presented the comparison of average execution times and cost of the tasks to other algorithms and resulted generated through it support our work.

Keywords- Load balancing algorithm, heuristic load balancing, Gridsim, average execution time, cost.

I. INTRODUCTION

Grid computing is a model of distributed computing that uses geographically and administratively disparate resources [5]. In Grid computing, individual users can access computers and data, transparently, without having to consider location, operating system, account administration, and other details. In Grid computing, the details are abstracted, and the resources are virtualized.

Grid Computing has emerged as a new and important field and can be visualized as an enhanced form of Distributed Computing [6]. With the advent of new technology, it has been realized that parallelizing sequential applications could yield faster results and sometimes at a lower cost. Possessing multiprocessor systems was not possible for everyone. Thus, organizations started looking for a better and feasible alternative. It was found that though every organization possessed a large number of computers, which meant that they had huge processing power, but it remained underutilized. A new type of computing then came into existence, known as Distributed Computing. In Distributed Computing, the problem to be solved is divided into numerous tasks that are then distributed to various computers for processing. The computers being used are generally connected through a Local Area Network (LAN). Also the problem needs to be divided into modules that can be executed in parallel to each other. Then the Grid computing comes into existence. Grid computing is the next generation IT infrastructure that promises to transform the way organizations and individuals compute, communicate and collaborate. Sharing in a Grid is not just a simple sharing of files but of hardware, software, data, and other resources [6]. Thus a complex yet secure sharing is at the heart of the Grid. The goal of Grid computing is to create the illusion of a simple but large and powerful self-managing virtual computer out of a large collection of connected heterogeneous systems sharing various combinations of resources.

In this paper we propose a new load balancing 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 present an algorithm which considers the job size and we mainly focused on formulating a decentralized heuristic based load balancing algorithm for resource scheduling. The rest of the paper is organized 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

A lot of research had already been done in the field of distributed environment related to load balancing. Most DS S.Ratnasamy,P.Francis,M.Handley,R.Karp and S.Shanker[2001] [7]Ion Stonica,Robert Morris,David Kanger,M.Frans Kaashoek and Hari Balakrishnan[2001][8] Kris Hildrum,John D.Kubatowicz,Satish Rao,and Ben Y.Zhao[2002][9] assume that object IDs are uniformly distributed. In Linvy M,and M.Melman[1982][2] authors proposed the notion of virtual servers as a means of improving load balance for the first time. David Karger and Matthias Ruhl[2003][11] Presented algorithms which dynamically balance load among peers without using multiple virtual servers by reassigning

lightly loaded devices to be neighbors of heavily loaded devices. However, they do not fully handle the case of **heterogeneous device capacities.**

In Brighten Godfrey,Karthik lakshminarayan,Sonesh Surana,Richard Karp,Ion Stonica[2004][13] authors presented an algorithm for LB in dynamic, heterogeneous P2P systems. Presented algorithm may be applied to balance one of several different types of resources, including storage, bandwidth, and processor cycles.

The stability of an approximate linear model Qingyuang Meng,Jianzhong Qiao,Jun Liu and Shukuan Lin[2008][25] in load balancing system was analyzed and then proofs the asymptotic stable condition via some classical control theories. In this way, the relations between load balancing gain and scalability of system was found. The deficiencies were that this model assumes the **input data is deterministic,** but in real parallel computing environment, the disturbances often exist. So a method thinks over stochastic disturbances needs to be improved. Moreover, the **topology structure, heterogeneous nature and time discreteness of DS** is not considered much in this work.

In Paul Werstein,Hailing Situ and Zhiyi Huang[2006][17] authors proposed a dynamic load balancing algorithm which considers CPU length, CPU and memory utilization and network traffic as load metric. In this result was compared with the results of algorithm that is using only queue length as metric.

In Xio Qin and Hong Jeong[2005][15] authors developed a LB strategy for Communication–Intensive applications to improve the band-width across network of cluster. This scheme can make use of an application model to quickly and accurately determine the load induced by a variety of parallel applications.

A communication-sensitive load balancer has been proposed by authors in S.Hotovy,D.Schneider and T.O'Donnell[1996][4]. The balancer uses run-time communication pattern between processes to balance load.

In K.Shahu Chatrapati,K.Purna Chand,Y.Ranjith Kumar,A.Vinaya Babu[2008][22] authors presented a load metric "Tendensive Weight" and this metric consider both CPU and Input-Output utilization including memory access rate. And they developed an algorithm for calculation of the Tendensive Weight of each task to be distributed to the node.

The workload estimation of each device for LB using fuzzy system is implemented in Yu-Kwong,Lap-Sun[2004][14] in which run-queue length and CPU utilization are used as the input variables for fuzzy sets and a set of membership function was defined. This scheme focuses only on run queue length & CPU utilization factors, other factors such as cost of migration, reliability etc. were not considered.

In M.Arora,S.K.Das and R.Biswas[2002][10] authors proposed a decentralized load-balancing algorithm for a Grid environment. Presented method does not consider the actual cost for a job transfer. In Y.Murata,H.Takizawa,T.Inaba and H.Kobayashi[2006][16] and H.Shan,L.Oliker and R.Biswas[2003][12], a sender processor collects status information about neighboring processors by communicating with them at every load-balancing instant.

In order to reduce the communication overheads, in R.Martin,A.Vahdat,D.Culler and T.Anderson[1997][5] authors studied the effects of communication latency, overhead, and bandwidth in cluster architecture to observe the impact on application performance and in Siri Luck Corpunmanee,Mohd.Noor Sap,Abdul Hanan Abdullah and Chaichompooinwai[2007][18],Bing Qi Chunnhui Zhao[2007][19] authors presented an ant colony based method for load distribution in which appropriate attention is not given for job migration cost. In B.Yagoubi,Y.Slimani[2007][21] authors presented a tree based algorithm for dynamic load balancing in grid environment. L. Anand, D. Ghose, and V. Mani,[1999][6] in which job migration cost, resource heterogeneity, and network heterogeneity when load balancing are considered.

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:-

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.

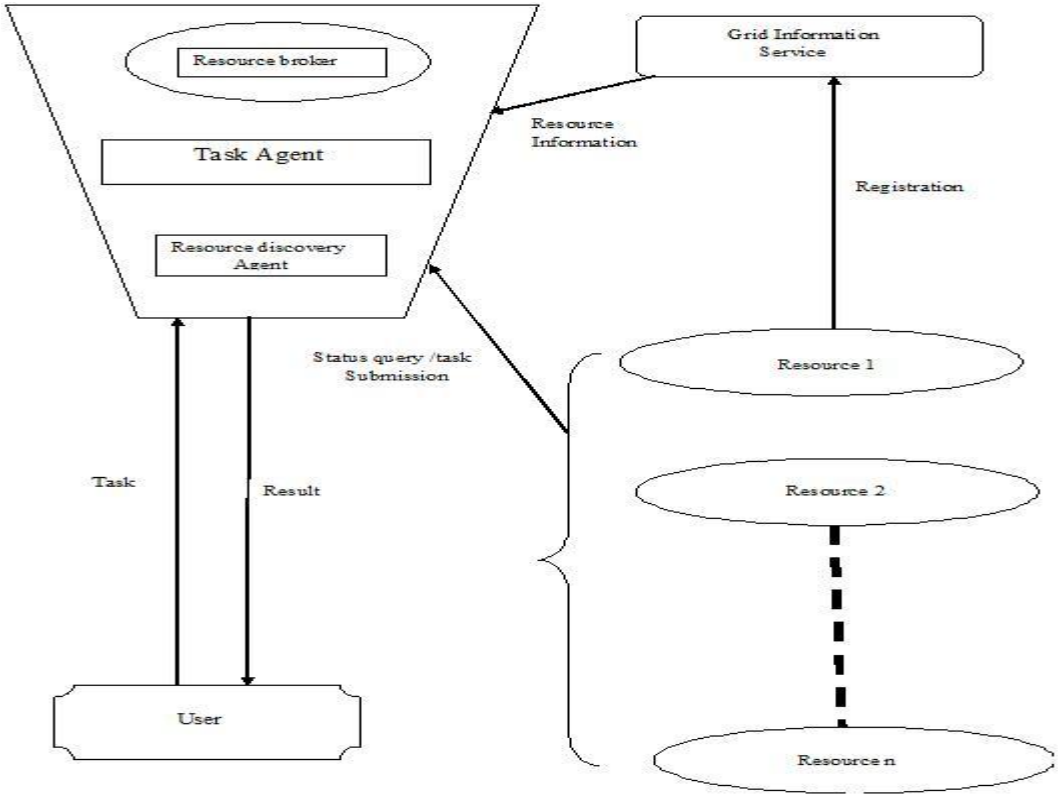


Fig.1. Grid Portal

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 load-balancing algorithms adapted to the heterogeneous Grid computing environment. In this paper, we attempt to formulate an adaptive decentralized sender-initiated load- balancing algorithms for computational Grid environments.

IV. HEURISTIC LOAD BALANCING ALGORITHM

An algorithm for load balancing is successful, if the task executes within the deadline. All the tasks in this algorithm are distributed according to Ant Colony Algorithm. HLBA is inspired on an analogy with real life behavior 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 HLBA, 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 Load Balancing Algorithm is shown below:

Let the number of tasks (ants) in task set *T* maintained by task agent is *P* and the number of registered resources is *Q*. When a new resource *i R* is registered to GIS, then it will initialize its pheromone based on:

$$\tau i(0) = N \, x \, M$$

where *N* represents the number of processing elements and *M* corresponds to MIPS rating of processing element. Whenever a new task is assigned to or some task is returned from *Ri*, then the pheromone of *Ri* is changed as:

$$i\tau_i^{new} = \rho * \tau_i^{old} + \Delta\tau_i$$

where Δ_i is pheromone variance and ρ, $0 < \rho < 1$ is a pheromone decay parameter. When a task is assigned to R_i, its pheromone is reduced i.e. $\Delta_i = -C$, where *C* represents the computational complexity of assigned task. When a task is successfully returned from R_i, $\Delta_i = \Phi * C$, where Φ is the encouragement argument. Pheromone increases when task execution at a resource is successful. The possibility of next task assignment to resource R_j is computed as:

$$p_j(t) = \frac{[\tau j(t)]^{\alpha} * [\eta j]^{\beta}}{\sum r [\tau j(t)]^{\alpha} * [\eta j]^{\beta}}$$

where *j, r* ∈ *available resources*. τ_j (*t*) denotes the current pheromone of resource R_j and η_j represents the initial pheromone of R_j i.e. $\eta_j = \tau_j(0)$. α is the parameter on relative performance of current pheromone trail intensity , β is the parameter on relative importance of initial performance attributes.

So the algorithm is designed in order to influence the performance of the system which depends upon several factors and these factors could be simplified for reducing the complexities of the method to be used. Heuristic Load Balancing Algorithm is presented below:

Phase 1 (Initialization Phase)

Begin
Initialize 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).

$p_i(t) = max\ p_l(t)$
$l \in q$

Phase 3 (Result Phase)
Schedule task't' to R_i 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.

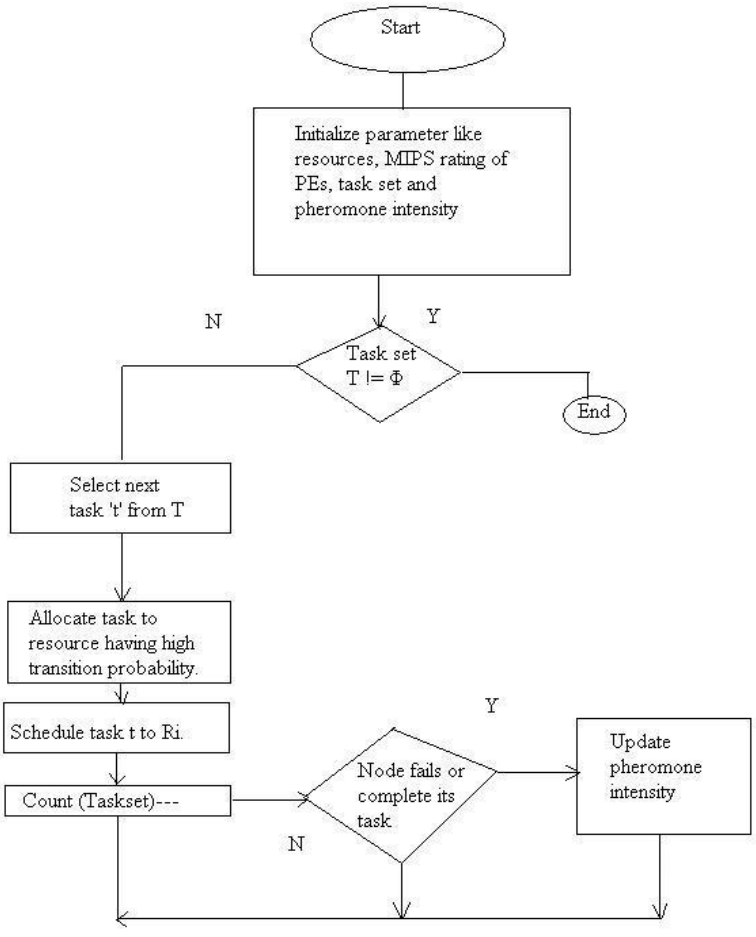


Fig.2. Flowchart for Heuristic Load Balancing Algorithm

V. 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 modeling and simulation of distributed resource management and scheduling for Grid computing [11].

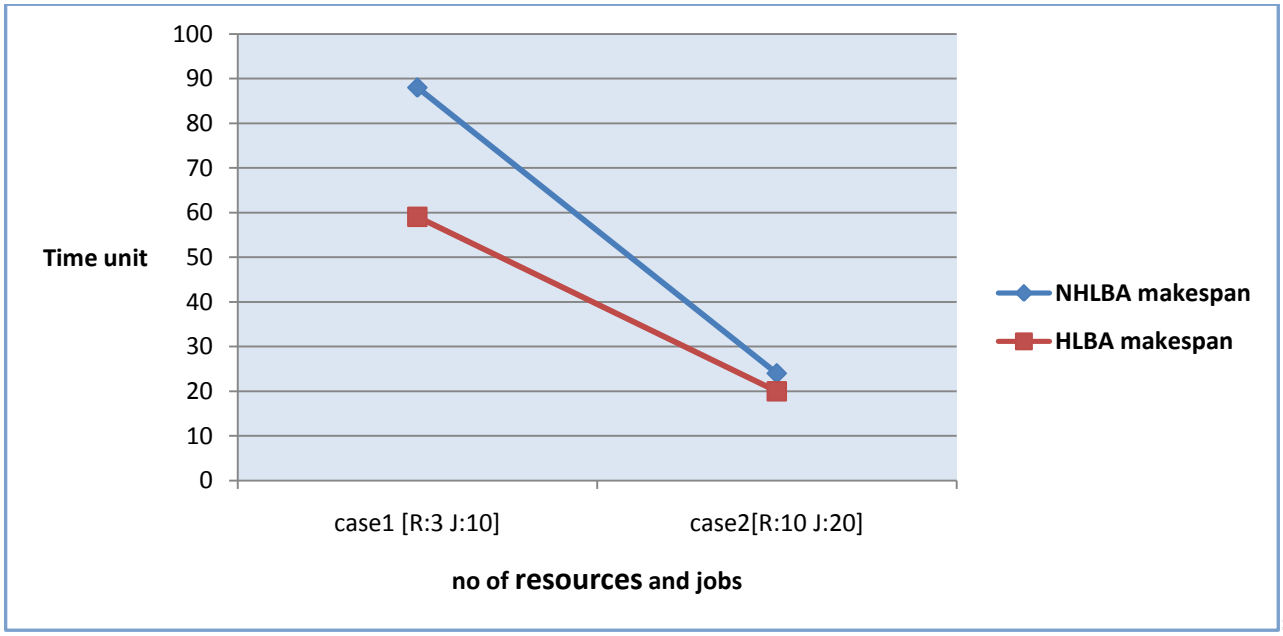
In order to test the efficiency of the improved load balancing approach a comparison is done for the execution time (time utilized by PE for processing a task) of heuristic load balancing algorithm and non-heuristic load balancing algorithm.

Two experiments have been conducted to evaluate the performance of the proposed algorithm in terms of their processing time. First experiment processed 10 jobs with fixed number of resources taken as 3 while the second experiment processed 20 jobs with number of resources 10. Details of the scheduling parameters are shown in the Table 1.

Case	**Ist**		**2nd**	
	Existing NHLBA	Proposed HLBA	Existing NHLBA	Proposed HLBA
No. of resources	3	3	10	10
No. of Jobs	10	10	20	20
Makespan	88	59	26	20
Average Execution Time	8	5	1	1

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 3 and the number of tasks varied from 10. It can be seen that the proposed algorithm labeled as Heuristic based load balancing algorithm has the lower execution time as compared to Non-Heuristic based load balancing algorithm at different instants. A comparison between the computation times for the second experiment is shown in Fig. 4. Heuristic load balancing 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.

In the second experiment the number of task remain fixed as 20 and number of resources 10 and again compute the total execution time(in seconds) for different number of resources. Result supports the use of HLBA for time efficient execution of tasks then NHLBA.



s

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

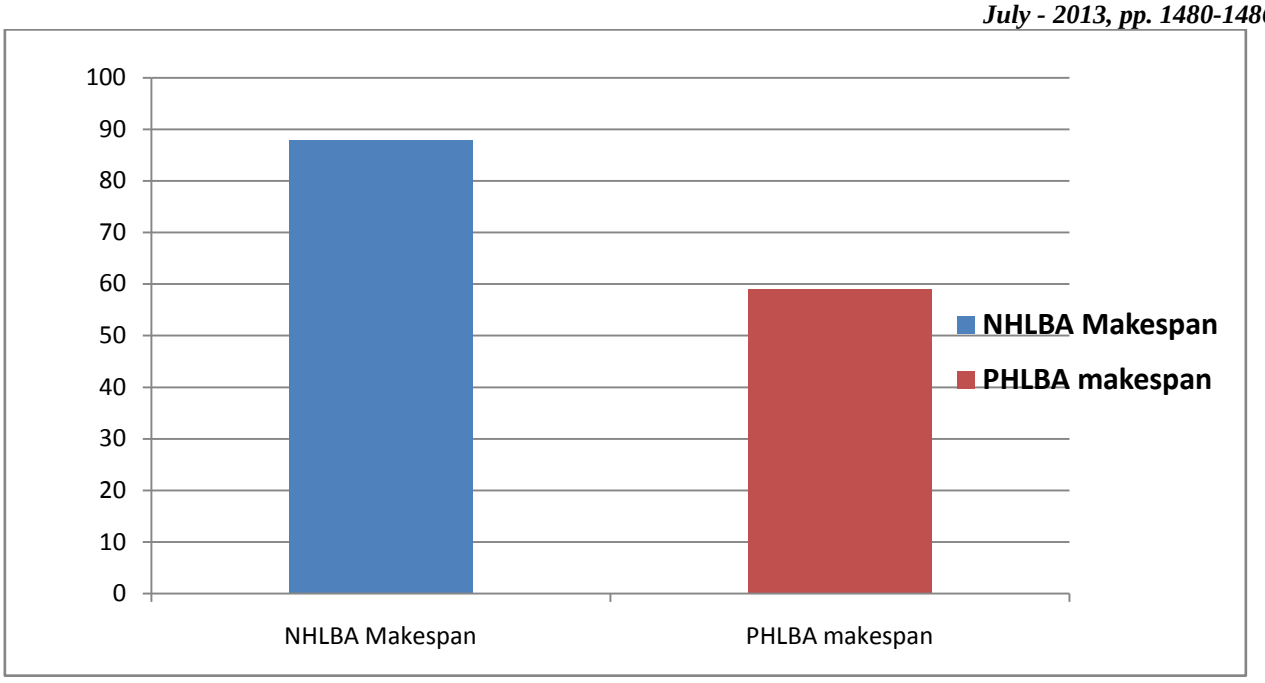


Fig.4.Comparison among the Execution times of Heuristic and Non-heuristic Load Balancing Algorithms

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.

VI. CONCLUSION

Multitasking and multiprocessing systems allow concurrently running tasks to share system resources such as processors, memory, storage, I/O, and network by scheduling their use for very short time intervals. In this paper an algorithm that is adaptive, decentralized and distributed for load balancing 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 HLBA enhances the performance of the system by reducing the execution time of the jobs.

REFRENCES

[1] I. Foster, and C. Kesselman, ”Globus: a metacomputing infrastructure toolkit”, International Journal of High Performance Computing Applications, vol. 11, no. 2, pp. 115-128, 1997.

[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] O. Ibarra and C. Kim, ”Heuristic algorithms for scheduling independent tasks on nonidentical processors”, *Journal of the ACM (JACM)*, vol. 24, no. 2, pp. 280-289, 1977.
[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, 1981.
[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, 2007.
[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.

