Balanced Distribution of Load on Grid Resources using Cellular Automata

Amir Akbarian Sadeghi
Department of Computer Engineering
South Tehran Branch, Islamic Azad University, Tehran, Iran

Ahmad Khademzadeh
Research Institute for Information & Communication Technology
Tehran, Iran

Mohammad Reza Salehnamadi
Department of Computer Engineering
South Tehran Branch, Islamic Azad University, Tehran, Iran

***Abstract*—Load balancing is a technique for equal and fair distribution of workloads on resources and maximizing their performance as well as reducing the overall execution time. However, meeting all of these goals in a single algorithm is not possible due to their inherent conflict, so some of the features must be given priority based on requirements and objectives of the system and the desired algorithm must be designed with their orientation. In this article, a decentralized load balancing algorithm based on Cellular Automata and Fuzzy Logic has been presented which has capabilities needed for fair distribution of resources in Grid level.**

Each computing node in this algorithm has been modeled as a Cellular Automata's cell and has been provided with the help of Fuzzy Logic in which each node can be an expert system and have a decisive role which is the best choice in a dynamic environment and uncertain data.

Each node is mapped of one of the VL, L, VN, H and VH state based on information exchange on certain time periods with its neighboring nodes and based on fuzzy logic tries to decrease the communication overhead and estimate the state of the other nodes in subsequent. The decision to send or receive the workload is made based on each node state. Thus, an appropriate structure for the system can greatly improve the efficiency of the algorithm. Fuzzy control does not search and optimize, just makes decisions based on inputs which are effective internal parameters of the system and are mostly based on incomplete and nonspecific information.

Each node based on information exchange at specific time periods with its neighboring nodes, and according to Fuzzy Logic rules is mapped of one of the VL, L, N, H and VH states. To reduce communication overhead, with the help of Fuzzy Logic tries to estimate the state of the other nodes in subsequent periods, and based on the status of each node, makes a decision to send or receive workloads. Thus an appropriate structure for the system can improve the efficiency of the algorithm. In fact, Fuzzy Logic does not search and optimize, just makes decisions based on the input parameters which are often incomplete and imprecise.

Keywords*—Computing Grid; Load balancing; Cellular Automata; Fuzzy Logic

I. Introduction

The need for high computational power and organizational limitations have created a new type of shared computing environment, which is called grid computing. Grid computing is a computing infrastructure Which provides access to high-performance computing resources. End users and applications see this environment as a large virtual computing system. Systems that are connected to Grid may be distributed globally and be running on different hardware platforms and operating systems and belong to various organizations. In a short definition, Grid can be considered as a system for distributed resource sharing on a large scale and indeed without borders. To improve the global throughput of the Grid computing, requests must be divided evenly among the available resources. Resource management is a major and infrastructure issues in this environment. The overall objective of resource management is the effective scheduling to run jobs that need to use resources in Grid environment. In a general definition, the purpose of load balancing algorithms is improving the distribution of workloads across resources, maximizing throughput, and minimizing response time, which means the difference between the overloaded and under-loaded resources should be minimal. The desirable characteristics of a load balancing solution include: scalability, adaptability, stability, application transparency, fault tolerant and minimal overhead. The load balancing methods are generally classified as centralized or decentralized, static or dynamic, periodic or non-periodic, and with threshold or without threshold.

Cellular Automata answer this question that How complex systems can be studied. There is the ability to predict the next state of cells in this system based on the status of each cell and its neighboring cells which can help in proper distribution of load among nodes. Fuzzy Logic can make effective decision by imprecise and incomplete Information. Cellular Automata can evaluate the current and future status of each resource by Fuzzy Logic rules to improve load distribution among the heterogeneous resources [14-19].

This article introduce a Grid load balancing algorithms based on Cellular Automata and fuzzy rules. The remainder of the paper is organized as follows: In section 2, the definition of concepts such as Grid, Load balancing, Cellular Automata and Fuzzy Logic. Section 3 describes our proposed algorithm in detail. Section 4 discusses our simulation and results of evaluation. Finally, section 5 concludes this paper

II. Definition of Concepts

Advances in areas technical constantly need to have faster computing, but computer hardware manufacturers have reached fundamental limitations in the physical speed [1]. Electronics and hardware advances in technology alone cannot meet the demand for increased computing speed. Parallel

processing is the emerging response to this problem in which large problems can often be divided into smaller ones, which can then be solved at the same time on several processors [2, 3].

Although writing code that is flexible enough to be split among several processors is more complicated, but the tendency toward parallel processing hardware and software has increased [2]. Instead of limiting the execution of a task in a processor, parallel processing divide many calculations among several processors and solve this issue through team work [3]. Reduce the cost of computers and advances in communication networks have increased the tendency for the use of large-scale parallel systems and distributed computing systems. In fact, recent studies in the field of computing architecture have led to the emergence of a new computing paradigm which is Grid computing[11]. A computational Grid is a hardware and software infrastructure that provides dependable, consistent, pervasive, and inexpensive access to high-end computational capabilities [12]. This technology is a type of distributed system that collects computer resources from multiple locations to reach a common goal to solve a single task. Nowadays, a variety of Grid systems is manufactured with various definitions and facilities which have different objectives. Thus, providing a single definition that covers all aspects of Grid computing technology is not easy nor true. Various experts have provided different definitions according to different pursued goals.

Ian Foster who was the main inventor of Grid and founder of Globus defines Grid as follows [12]:

"Grid technology is seeking to create the possibility of large-scale and controlled resource sharing which is flexible and is after creating protocols, services and software packages."

Gird is defined as follows in IBM Company which is among pioneers of Grid:

"Grid is a set of distributed computing resources in a local area network or a wide area network which seems like a computer and virtual computing system for end-user or applications. Its main goal is creating dynamic virtual organizations through sharing resources using coordinated and safe methods among users, universities and organizations."

Grid is a distributed system which contains following items [8]:

- Resources (software and hardware) are heterogeneous
- Resources are coordinated but are not under a centralized management
- The use of all-purpose standard protocols and interfaces
- Grid may have Multiple administrative domains or in other words be made of several Virtual Organizations (VO)
- Ensuring the quality of the services provided

Resource management is one of the important and infrastructure issues in such environment. The overall objective of resource management is the effective scheduling to run jobs that need to use resources in Grid environment. In a general definition, the purpose of load balancing algorithms is improving the distribution of workloads across resources, and maximizing their performance as well as reducing the overall execution time [9]. In another definition, the load balancing algorithm is an algorithm which ultimately allows all nodes to task at once [10-11]. The desirable characteristics of a load balancing solution include: Scalability, Adaptability, Stability, Application Transparency, Fault Tolerant and minimum overhead imposed on the system. The mentioned specifications are interdependent. For example, delays such as Computation Delay and Communication Delay have abnormal effects on the stability and thus comparability of the algorithm. Due to the many parameters involved in the problem of load balancing as well as contradictory of some mentioned features, meeting all the them in the form of a single algorithm is practically difficult or even impossible. Most of the existing methods try to satisfy one or more of the above objectives [12-14].

For better efficiency and more use of dynamic algorithms and considering that the main focus of this article is on the same set of algorithms, in general, the process of dynamic load balancing algorithms has four main procedures:

A. Load Measuring procedure

B. Information Exchange procedure

C. Initiation procedure

D. The final load balancing procedure

Load measuring procedure is an expression of CPU load in a way that heavier load on processors will increase it, and its reduction will reduce it. Because this routine repeatedly execute in this algorithms, the calculation of it should be as simple and efficient as possible [17]. Information Exchange procedure determine the method of collecting necessary task load for load balancing decisions. Initiation procedure decides about the time of starting load balancing. This decision-making is along with determining the ratio of efficiency to imposing overhead. Load balancing methods attempt to achieve goals such as minimizing the average response time for processing or maximizing resource efficiency by running processes on distributed resources. This target may initially be a demand or take place after the start of its execution. Of course, in any case, a good and efficient algorithm must consider the cost of the communications as well [18]. Cellular Automata (CA) is an answer to this question that how to study complex systems. Cellular Automata can be a complex system in itself and yet provide appropriate methods to study complex systems like these - Complex systems – [19-20].

III. The Proposed Algorithm of Fuzzy Load Distribution using Cellular Automata (Fuzzy Load Balancing Cellular Automata)

The main idea of this project is using a cell of Cellular Automata to show a computational node in which the state of the cell shows the status of that node. In this method, a global load balancing solution can only be produced just using local load balancing. This method of load distribution is in the form of a wave motion.

All parameters that each processor considers during the proposed load balancing algorithm are described below:

M: Number of heterogeneous computing nodes in the system (P_1, P_2, ..., P_M)

x: Number of jobs executed in the system (J_1, J_2, ..., J_x).

T_s: Information exchange time.

T_e: The estimated time period.

N_i: Buddy set of node P_i.

$S_i(T_n)$: State of node 'I' at time T_n.

m: Number of migration of a job.

Q_i (t): The number of jobs waiting in the execution queue at the node P_i at time t.

W_i: Processing power at P_i.

Z (J_x): Size of job(x).

$TET_{i,t}$: Total waiting time for execution of waiting job at P_i queue.

$RET_{i,t}$: The remaining execution time of the job being processed at the P_i.

$LD_{i,t}$: Load of P_i at time t, comes from (1).

$$LD_{i,t} = TET_{i,t} + RET_{i,t} \quad (1)$$

$NLD_{i,t}$: Normalized average load in the buddy set of node P_i at time t.

BW_{ij}: Bandwidth communication between processors i and j

ArrTime(J_x): Arrival time of J_x job.

endTime(J_x): End time of J_x job.

ETC (J_x, P_i): Estimated execution time of J_x at P_i, comes from (2).

$$\text{ETC}(J_x, P_i) = \frac{\text{ETC}(J_x, P_{std})}{W_i} \quad (2)$$

$T_{com}(J_x, P_i, P_j, t)$: The time required for transfer J_x job from P_i to P_j at time t.

$EFC(J_x, P_i, P_j, t)$: Estimation of finish time of J_x job when transfer from P_i to P_j at time t.

if $T_{com}(J_x, S_i, S_j, t) \geq LD_{j,t}$

$$EFC(J_x, S_i, S_j, t) = T_{com}(J_x, S_i, S_j, t) + ETC(J_x, S_j)$$

ELSE

$$EFC(J_x, S_i, S_j, t) = LD_{j,t} + ETC(J_x, S_j) \quad (3)$$

$B_x(P_i, P_j)$: Benefit of execution of the J_x job at P_j compared to execution at P_i.

$$B_x = EFC(J_x, P_i, P_i, t) - EFC(J_x, P_i, P_j, t) \quad (4)$$

The general procedure of the proposed Load balancing algorithm is in the way when a new task enters the computational node, that node will decide based on cell's conditions should carry out this task itself or migrate it to another node.

This algorithm consists of several main procedure:

- Determining the state of nodes
- Making decision to migrate the task
- Selecting the best node to carry out the task

A. *Determining the state of nodes*

The overall basis for all decisions is the state of each node. In fact, the essential criterion in deciding to send a job is the state of the node, and the main criterion for selecting a node to perform the job is also the state of the node. Thus, determining the state of each node is crucial in load balance in the whole system.

To determine the state of each node and its neighbors for each execution and migration, the information is needed to determine the status of nodes. There will be a huge communication overhead in the system if all nodes exchange their status information. Thus, regular intervals are used to determine the state of nodes which are called information exchange periods (T_s). T_s which is greater than the period of time for running and migration of jobs is performed between nodes which estimate the state of nodes between these time periods. T_e tries to reduce communication overhead and more accurate decisions (Fig. 1).

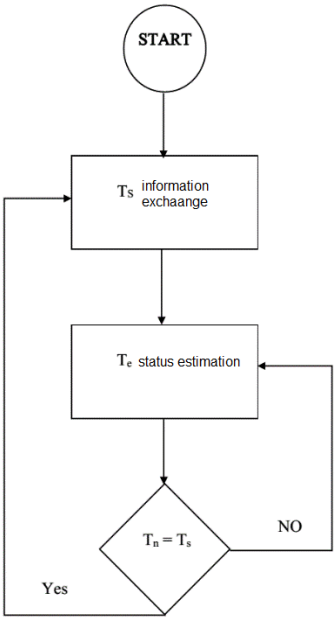

Fig. 1. Determining the state of nodes in T_n

Determining the state of the node is done in two methods:

- Information Exchange (T_s)
- State estimation (T_e)

1) Determining the state of nodes through the information exchange

It contains three parts of sending load information, calculating the average load and determining the status of nodes using Fuzzy Logic (Fig. 2).

a) Sending load information

The nodes send their status information to their buddy set in regular time periods of T_s which are called information exchange periods and receive their information. Then each node records the received information from each neighboring node in the neighboring table.

b) Calculate the average load

Each node calculates its average load and the load related to neighboring collection using (5) and considers the obtained index as normalized load average.

$$NLD_{i,t} = \frac{\sum_{j \in N_i} LD_{j,t}}{N \times \sum_{i \in N_i} w_i} \quad (5)$$

c) Determining the state of nodes using Fuzzy Logic

The normalized load average is considered as the point of balance and map the state of each node to one of Very Light, Light, Normal, Heavy, Very Heavy forms using Fuzzy Logic. This step is called mapping input values to the fuzzy mode and the state of each node will be recorded in the neighboring table.

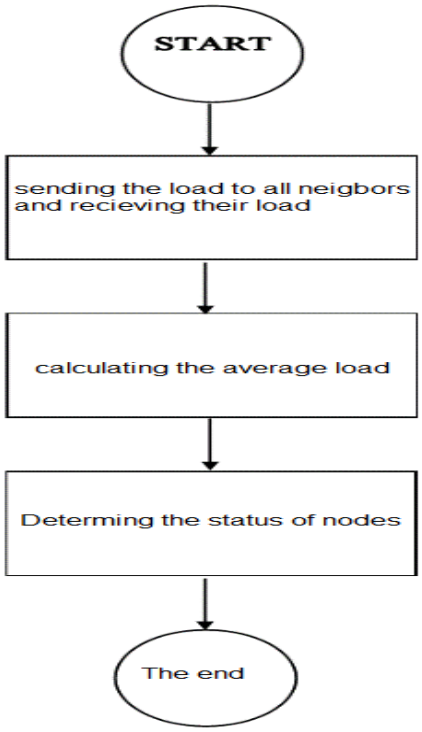

Fig. 2. Information exchange

2) Determining the state of node using estimation

If this data transfer is done in intervals with a short distance, there will be a high communication overhead imposed on the system. Thus, these intervals must be increased and estimated T_e state in information exchange periods using fuzzy rules.

To discover these laws, the algorithm runs without estimation periods and records the data on each node. The related fuzzy rules are extracted by using Matlab software. To reduce communication overhead caused by data exchange, exchange periods will be increased, and the status of each node using obtained rules to reduce errors in decision making will be estimated.

B. The decision to send task

When job J_x enters node, and that node is in one of VL, L, VN states, it will be queued for processing and will be waiting to run according to respective priority, and the higher rate of migration (ɱ) leads to increased running priority.

Otherwise if J_x enters node, and that node is in one of H, VH states, and it migrated from another node, it will not be accepted, on the other hand if the job belongs to that node, if the node is in VH state then it calculates its own load and the normal load, and if it has a neighbor with VL state, it sends ½ of its extra load and if there is a neighbor with L state, it sends ¼ of its extra load. If its state is H, it will calculate the difference between its own load and the normal load, and if there is a neighbor with VL state, it will select ¼ of its extra load for migration, and selects 1/8 if there is a neighbor with L state.

C. Selecting the most appropriate node to execute the job

At the beginning, nodes in the buddy set are marked with the weight of 1 for L nodes, and 2 for VL nodes. Then the execution time of the job J_x is estimated on specified nodes and based on B_x (P_i, P_j) benefit which is the difference between execution time in node P_j compared to P_i node, if this benefit is positive and bigger than the threshold, a weight is given to each of them. In this way, the higher execution benefit leads to higher score. Finally, considering the weight of each node and the weight of execution benefit in P_j node, the decision to transfer job to P_j is taken. the higher weight will lead to a higher possibility of sending the task to that node (Fig. 3).

If the execution time of a job is equal in the source and destination nodes, which means running the job in the source node or migrating it to another node may result in similar end time, and the benefit will be nil or smaller than threshold. In this case, the job is not allowed to be migrated to that node, because it imposes communication overhead on the system and the bandwidth between resources is engaged even for small time.

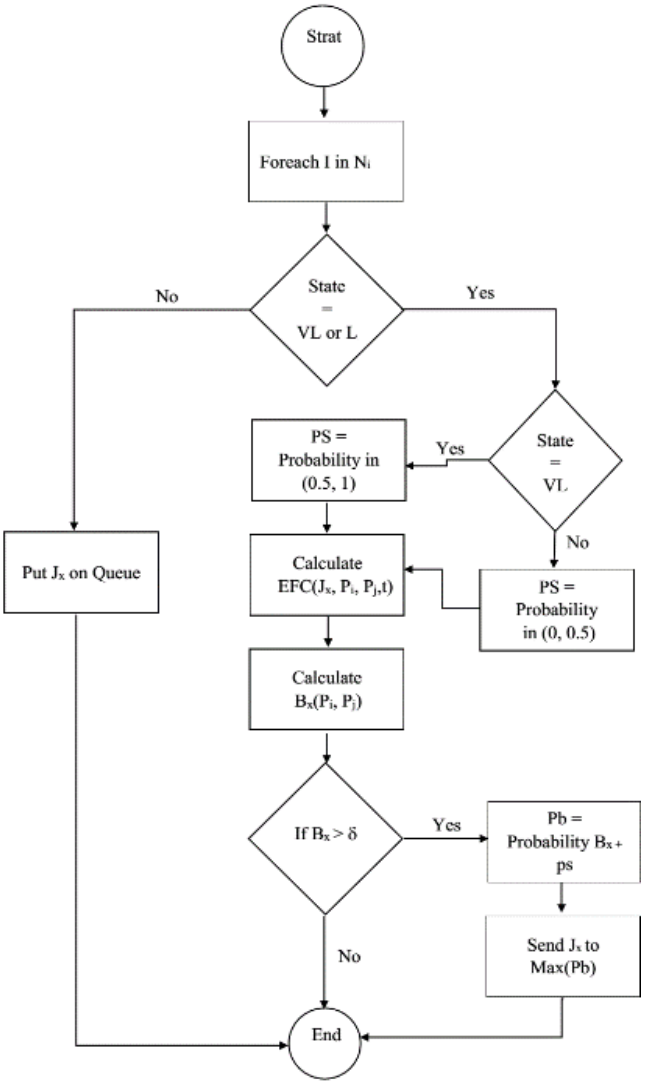

Fig. 3. Selecting the most appropriate node to execute the job

IV. Simulation and Results

Simulation is an imitation of the real world. Simulators enable programmers, developers and Grid developers who need to test their programs, tools and services to ensure the proper function of their program before final production and using them in a real world.

The proposed FLBCA algorithm has been simulated using C# programming language in Visual Studio 2010 environment, and its algorithm has been compared with MELISA and ELISA algorithms.

A. Performance metrics

Following two parameters have been used to evaluate the performance of the proposed algorithm [21-23]:

1) Average Response Time (ART):

The average response time is the average amount of time a job must wait before execution.

2) Average time of Resourses Used (ARU):

The ratio of working time to the total time of a system:

$$U_i = \frac{Busy_i}{Busy_i + Idle_i} \quad (6)$$

$$ARU = \frac{\sum_{i=1}^{M} U_i}{M} \quad (7)$$

B. Simulation Model

This simulation is formed by 30 heterogeneous computing nodes which their processing power follows random distribution in the range of [1, 10] and the relation between two nodes has been formed by a heterogeneous communication network in a way that their communication bandwidth is variable from 1 Mbps to 10 Mbps.

10,000 independent tasks have been used in this simulation in a way that running time of each task has been generated randomly in the range of [1, 100]. These tasks enter the system based on Poisson distribution with the rate of [1, 4], and the volume of each task follows a normal distribution with the mean of 5 MB and standard deviation of 1 MB.

The time for information exchange (T_s) is assumed to be 20 units and the estimation time of status is considered to be five units.

C. Simulation results

This algorithm has been evaluated regarding performance metric and under the effect of parameters such as the number of tasks, time and period of service transition and estimation interval.

1) The effect of the jobs entered in the homogeneous environment

In a homogeneous environment where processing power of each CPU is 1, and the communication bandwidth between any two nodes is constant and equal to 10 Mbps, the number of tasks has been added from 0,000 to 50,000 to measure these factors. In these conditions where the number of tasks is 10000. The average response time is about the same among all three algorithms although after increasing jobs, the efficiency of ELISA algorithm is better than MELISA and the proposed algorithm (FLBCA), and the efficiency of FLBCA algorithm is slightly better than MELISA algorithm (Fig. 4). The total run time is about the same in all three algorithms (Fig. 5).

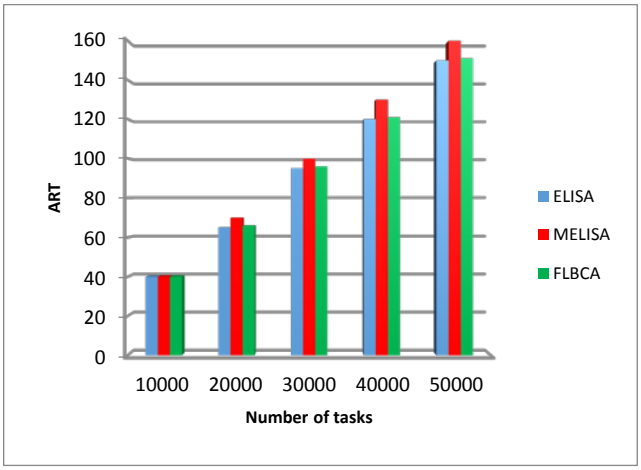

Fig. 4. The average response time in case of homogeneous

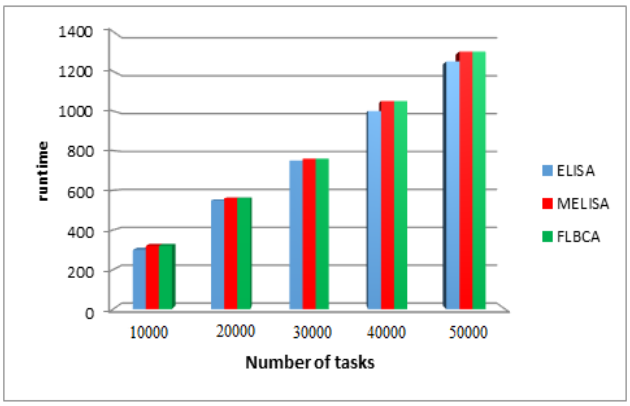

Fig. 5. The total runtime in the homogeneous environment

2) The effect of jobs entered into the heterogeneous environment

Since this algorithm has been designed for a heterogeneous environments, it is compared in a heterogeneous environment with different processing power in the range of [1, 10] and various communication bandwidths between any two nodes in the range of 1 Mbps to 10 Mbps with ELISA and MELISA algorithms. To measure the effectiveness of the jobs, the number of jobs has been increased from 10,000 to 50,000. The average response time is much better in FLBCA and MELISA algorithms than the ELISA algorithm. The average response time is initially about the same in FLBCA and MELISA algorithms, but is gets better with increasing number of tasks in FLBCA algorithm, and it shows better and faster decision making in Fuzzy Logic (Fig. 6).

The total runtime is similar to all three algorithms. The runtime is a function of the rate of entering jobs into the system, and the runtime is about the same due to using same data (Fig. 7).

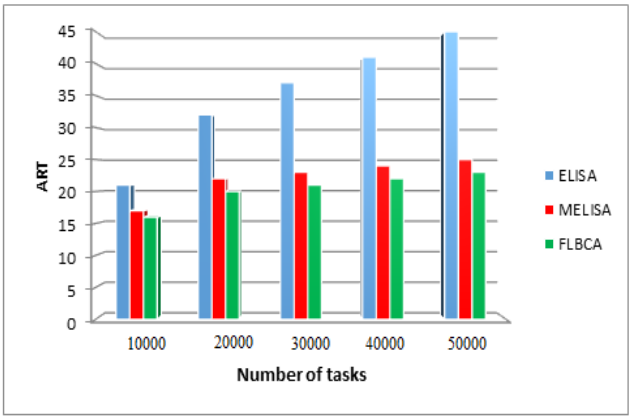

Fig. 6. Comparing the average response time in the heterogeneous environment

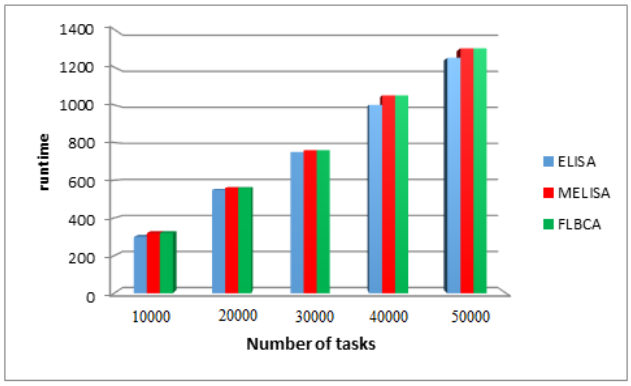

Fig. 7. Comparing the total runtime in the heterogeneous environment

3) The effect of job size

This section tries to evaluate the effect of changing tasks volume from 5 MB to 50 MB on the average response time in the proposed algorithm. The number of migration reduces and the average response time increases (Fig. 8).

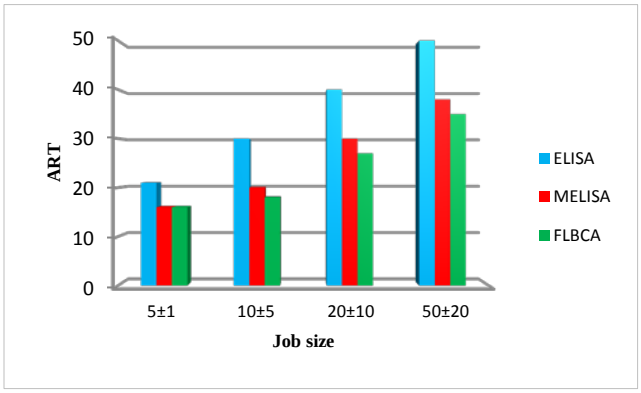

Fig. 8. Average response time for different job sizes

4) The effect of job runtime

This section tries to evaluate the effect of changing the average runtime of tasks from 10 to 150 units on the efficiency of the algorithm. The average response time increases with large rate by increasing the runtime (Fig. 9).

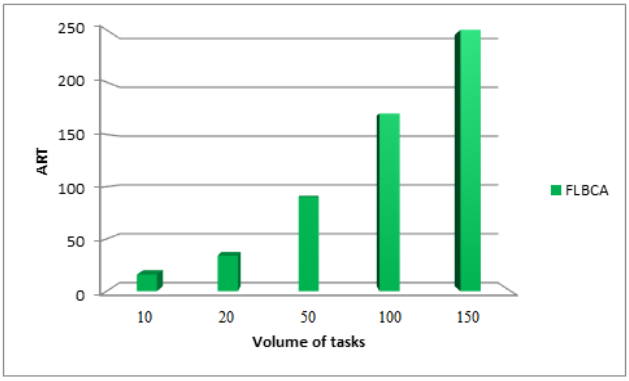

Fig. 9. Average response time for various run times

5) The effect of information exchanges time

This algorithm was tested with different times in range of 2 units to 40 units to find a best time for information exchanges which is suitable in perspective of efficiency criteria (Fig. 10).

The accuracy of information reduces with increasing time of the information exchange. The distribute the load between nodes is done with less precision and average response time increases for this purpose.

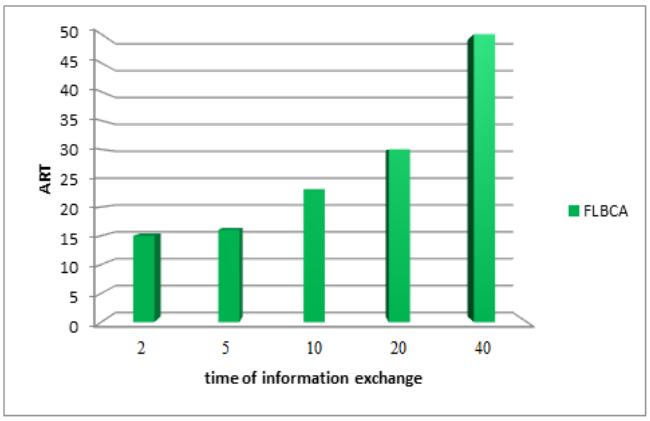

Fig. 10. Average response time for different times of exchanging information

6) The effect of state estimation time

To access a proper state estimation time from the point of efficiency criteria, the information exchange time must be considered to be 20 units by default. Then by changing the estimation time in different intervals from 2 units to 10 units, the most effective time will be found (Fig. 11).

The average response time increases by reducing estimation time due to increased computational overhead and the most optimal time is reached at times of 4 and 5 and the average response time increases again by increasing this time due to reduced accuracy of data.

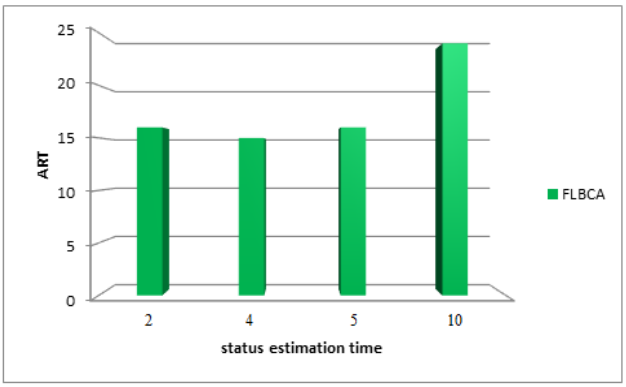

Fig. 11. Comparing the average response time for different times

V. Conclusion

A load balancing model has been provided in this research for Grid computing environment which is called FLBCA. This algorithm tries to meet needs and characteristic required for a load balancing algorithm such as scalability, adaptability, stability, application transparency, and minimal communication overhead as much as possible. Because the Fuzzy Logic can deal with imprecision and uncertainty information, and increase accuracy in decision making, the algorithm based on Fuzzy Logic has been proposed for dynamic load balancing in the computing Grid. The major purpose of using this algorithm is increasing efficiency and productivity of Grid system which will lead to reducing organization's costs, and increasing productivity and saving energy, and using energy efficiently saves it which is the top priority in our life today and helps to protect the environment and nature.

In the following, it seems hierarchical Cellular Automata is an appropriate structure for designing these types of algorithms, which can provide a better view of the whole conditions of the Grid System. In addition, the usage of Fuzzy Logic which leads to accuracy of decision-making in uncertain environments can be used to improve the efficiency of parallel algorithms. The combination of Fuzzy Logic and Cellular Automata can be a good technique for a lot of parallel algorithms.

References

[1] M. Fathi, F. Mehryari., "the principles and concepts of grid computing technology and its applications in various fields", Noorpardazan, .10, 2009 - Tehran, pp.

[2] M. Amini Salehi, H. Deldari, load balancing in the grid resources using agent-based resource management, Master's thesis, Department of Computer Engineering, Ferdowsi University of Mashhad, 2005.

[3] S. Ghanbari, balance and self-organization in the grid computing using learning automata, Master's thesis, Department of Computer Engineering, Amirkabir University of Technology, 2004.

[4] R. Tlili, Y. Slimani, A Hierarchical Dynamic Load Balancing Strategy for Distributed Data Mining, International Journal of Advanced Science and Technology, Vol. 39, February. 2012

[5] S. Adabi, reducing power consumption in wireless sensor networks based on cellular automata, Master's thesis, Department of Computer Engineering, Islamic Azad University of Science and Research Branch, 2010.

[6] L. Anand, D. Ghose, V. Mani, ELISA: An Estimated Load Information Scheduling Algorithm for Distributed Computing Systems, An International computers & mathematics with applications, 1999.

[7] L. Rostami, A. Rahmani, An adaptive Load Balancing Algorithm with use of cellular Automata for Computational Grid Systems, Euro-Par 2011 Parallel Processing, Lecture Notes in Computer Science Volume 6852, 2011, pp 419-430.

[8] A. Karimi, F. Zarafshan, A. Jantan, Anew Fuzzy Approach for Dynamic Load Balancing Algorithm, International Journal of Computer Science and Information Security, Vol. 6, No. 1, 2009.

[9] M. Marinov, Intuitinistic Fuzzy Load balancing in cloud computing, 8th Int. Workshop on IFSs, Ocy 2012.

[10] S. Mousavi Nejad, S. Mortazavi, B. Vosoughi Vahdat, Design and set optimal control and intelligent load balancing based on fuzzy logic in distributed systems, first regional conference on new approaches in computer engineering, 2011.

[11] I. Foster, and C. Kesselman, "The Grid: Blueprint for a New Computing Infrastructure" Morgan Kaufmann and Elsevier, Second Edition, USA, ISBN: 1-55860-933-4, 2004.

[12] I. FOSTER, C. KESSELMAN, M. NICK J, S. TUECKE, "Grid services for distributed system integration,« vol. 35, 6, 2002.

[13] C. J., K. E.,. L. M. Anderson D P., "SETI @ home: an experiment in public-resource computing," vol. 45 (11).

[14] S. Graupner, J. Pruyne, S. Singhal, "Making the Utility Data Center a Power Station for the Enterprise Grid," 2003.

[15] J. Liu, X. Jin, and Y. Wang, Agent-Based Load Balancing on Homogeneous Minigrids: Macroscopic Modeling and Characterization, IEEE TRANS.IONS ON PARALLEL AND DISTRIBUTED SYSTEMS, VOL. 16 (7) 2005.

[16] L.SUN CHEUNG, A fuzzy approach to load balancing in a distributed object computing network, In Proc. Of 6th Int IEEE Conf. HPDC, 2000.

[17] T. L. Casavants and J. G. Kuhl, A taxonomy of scheduling in general-purpose distributed computing systems, IEEE Trans. Software Eng., Vol. SE-14 (2), pp. 141-154, 1988.

[18] H. Kameda, J. Li, C. Kim, and Y. Zhang, Optimal Load Balancing in Distributed Computer Systems. London, U.K .: Springer-Verlag, 1997.

[19] Z. Zeng and B. Veeravalli, Rate-Based and Queue-Based Dynamic Load Balancing Algorithms in Distributed Systems, 10th Int. Conference on Parallel and Distributed Systems, IEEE 2000.

[20] Abubakar, Haroon Rashid and Usman Aftab, Evaluation of Load Balancing Strategies, National Conference on Emerging Technologies 2004.

[21] J.Cao, Daniel P. Spooner, Agent-Based Grid Load Balancing Using Performance-Driven Task Scheduling, In Proc. of 17th IEEE Int. Parallel & Distributed Processing Symposium (IPDPS 2003), Nice, France, April 2003.

[22] A. Shaout and P. McAuliffe, Job scheduling using fuzzy load balancing in distributed system, in Proc. of 6st conf ICPAD, 1998.

[23] J. Liu ,, X. Jin, and Y. Wang, Agent-Based Load Balancing on Homogeneous Minigrids: Macroscopic Modeling and Characterization, IEEE TRANS.IONS ON PARALLEL AND DISTRIBUTED SYSTEMS, VOL. 16 (7), 2005.