Challenges of Load Balancing to Support Distributed Exascale Computing Environment

Ehsan Mousavi Khaneghah[1], Faezeh Mollasalehi[1], Araz R. Aliev[2], Nigar Ismayilova[2], Ulphat Bakhishoff[2]

[1]Department of Computer Engineering, Faculty of Engineering, Shahed University, Tehran, Iran

[2]Department of General and Applied Mathematics, Azerbaijan State Oil and Industry University, Baku, Azerbaijan

Abstract - *Distributed Exascale computing systems are considered as a specific type of system that is capable of running dynamic and interactive applications. The existence of a dynamic and interactive nature of computational processes that require these types of systems creates situations that were not considered during system design. This paper examines the implications of the dynamic and interactive concept of the load distribution management by examining the concept of the dynamic and interactive nature of computational processes requiring distributed Exascale systems. This study extracts the effects of the dynamic and interactive nature on the load distribution by presenting a mathematical definition for the load distribution management. In this paper, it has been determined that if being able to redefine the load distribution based on the concept of time and considering the sensitivity to the system state at the time of load redistribution, the load balancing management can support distributed Exascale systems.*

Keywords: Distributed Exascale Computing, Load Balancing, Dynamic and Interactive Nature, System Status**.**

1. Introduction

The load distribution has the task of managing the computational processes in the system, is the most pivotal element of computing system management, in such a way that each process is responded as quickly as possible, and on the other hand, the computational resources available in the system are in the majority of the system's operating time [1,2,3]. To achieve this goal, this element uses two concepts, the requirements of computational processes and the property of resources in the system [4, 5, 6, 7]. The load distribution management should be able to handle system-level events or events that lead to a change in of these two concepts, by re-distributing and allocating resources to the computational processes; it is possible to overlook the two objectives mentioned above [8, 9, 10]. In distributed Exascale systems, the nature of the two sets of process requirements and resource characteristics is such that can be changed at any moment, the duration of the activity in the system [8, 9, 10, 11, 12]. At each change of either of these two sets, load distribution management should be able to handle the two main objectives of the computing system by managing these two sets and allocating resources to the processes So that the system responds to requests in the shortest possible time [13, 14]. In traditional HPC and processing power systems, the set of system resource features does not change over the life of the system [15]. On the other hand, the nature of the scientific and applied program running on this type of computing system is such that the nature of the program requirements does not change during the execution of the program [16, 17, 18]. This causes the load distribution management to be based on information received from the user as well as information about the implementation of the program extracted from the core data structure of the operating system, Decide on the nature of the requirements of processes [19]. The existence of accurate information about the nature of the process requirements and the lack of variability of the computational resources available in the system makes it possible, in the sense of applying the request to the requestor, a proper mapping between the process request and the resource attribute.

By changing the nature of scientific, engineering programs requiring high computing, and processing power systems, there is a need for distributed Exascale computing systems that can implement them based on the specific features of this kind of scientific and engineering programs. [20, 21, 22]. The most important feature of the scientific program is the need for distributed Exascale systems is the existence of computational processes of a dynamic and interactive nature [12, 13, 14]. The dynamic and interactive nature of this type of process makes the nature of the requirements of the processes change over the process execution time in the computing system [10, 13, 14]. On the other hand, being distributed of these types of computational systems causes the state and number of computing elements to have a change frequency [23].

The change in these two sets makes it unlikely that, unlike traditional computing systems, the load distribution management manages two dynamic sets. On the other hand, the nature of the frequency of events that results in a change in the status of each of the two sets of Distributed Exascale systems is higher than in traditional computing systems. The nature of science and engineering programs requires Distributed Exascale systems, and the platform of this type of computational system is the main reason for the high frequency of changes in the two sets mentioned [24].

In general, can be considered two ways to support the concept of distributed Exascale systems by the load distribution management. The first is to create the distribution management based on the features of distributed Exascale systems. The nature of computational systems is such that the concept of compatibility with the past has priority and importance in these types of systems [25]. Therefore, distributed Exascale computing management, especially the load distribution management, should have the highest degree of adaptability to implement traditional science and engineering programs. Therefore, the way is not appropriate to create a load distribution management based on the specific features of Distributed Exascale systems. The second way is to examine and analyze the basic concepts of distributed Exascale systems and its impact on the traditional load distribution and provide a proper way for managing them by the load distribution.

In this paper, based on the functional and operational generalizations of the load distribution in traditional computing systems, what does the concept of the dynamic and interactive nature of the load management effect, and the element of distributed load management in traditional computing systems faces challenges regarding supporting the concept of distributed Exascale systems?

2. Traditional Load Balancing

In traditional computing systems such as clustering systems, the load distribution management complies with the pattern shown in Formula 1, because of the processes Irreversibility also the resources in the system.

$$Load\ Distribution: [Process_{requirment}] \rightarrow [Resource_{Space}]\ Therefore\ Best_{Load-Distribution} = \begin{bmatrix} [\nexists \text{Process} \in \text{HPC}_{\text{Proces}} \overset{So\ Means}{\therefore} \ni HPC_{Process-Scheduling}] \\ and \\ [Resource_{Activity} = 100\%] \end{bmatrix} \quad (1)$$

As seen in Formula 1, the load distribution management creates a mapping between the process request space and the resources in the system. In traditional computing systems, there are no changes in the two processing space, as well as resources space in the system, before the load distribution activity begins or at the time of its operation [15]. This causes the load distribution to redistribute according to one of these two policies: return to the initial distribution state or go to the next state of equilibrium [26]. On the other hand, the load distribution management can use either the user's initial information or process data building information to determine the status of the system [19].

The load distribution performs a mapping operation between the two processing space and resource properties based on Part II of Formula 1. With this in mind, the load distribution in traditional computing systems can be expressed according to Formula 2.

$$Load\ Balancing_{Space}: (Resource_{Attribute}Set, Process_{Requirement}Set, < Metrics\ Bench,\ Time_{Space}, Distribution\ Pattern, Error, < Action >>) \quad (2)$$

As shown in formula 2, the load distribution is defined on the basis of the two spaces: the resource attributes and the process requests. This element has two main variables, two activities, and one operation. The Metrics Bench and $Time_{Space}$ variables are the two main variables of the load distribution, which the Metrics Bench variable represents is the concept that the load distribution uses to check the status of the system. Concepts such as the state of the central processing unit, memory status, and system bandwidth status are among the most important concepts used by the load distribution element. $Time_{Space}$ Variable represents all constraints, limitations and time-stamina that governs the load distribution. Deciding on the waiting time of the processes, the free time of the central processor, Time needed to redistribute the load, and most importantly, the time to collect information about the system status is one of the most important constraints, which is discussed in different load balancing algorithms.

$Distribution\ Pattern$ Activity represents a pattern of load distribution uses by the distributed load

management, with different algorithms. $Error$ Activity represents a pattern of load distribution collisions in the event of the inability to distribute or redistribute the load. The Action element is a set of tasks that performs the load distribution management when the load distribution status is compromised in the computing system. The action element is in fact the mechanism of collision of the load distribution with the disruption of the system load condition. The most important feature of Action concept is to stop operation. In a computational system, the activation of the load distribution is based on one of the two concepts of event and time [9]. In an event mode, an event occurs at the level of the computational system, which leads to the activation of the load distribution [27]. In time mode, the load distribution is activated at certain times or due to occurrence of certain time events [27]. The load distribution that is activated for any reason follows the same system as the system described in Formula 3.

$$LB\ System: (System_{State}, Process_{Request}, Resource_{Condition}) \rightarrow (New\ System_{State}, Resource_{condition}, Process\ Reload) \quad (3)$$

As stated in Formula 3, if the load distribution is a system, the input of this system is the current state of the computing system, the request for processes in the area of access to the central processor and the status of resources in the system. The state of the system is meant to be a factor in the process of implementing the process, the formation of the application process and the way of communication and interactions between resources. As seen in Formula 3, when the load distribution is activated, the load distribution receives information about the status of the system as input [28]. The definition of a system depends on the implementation of the burden distribution management [27]. If the central distribution management is implemented, then the system's meaning is the whole of the computational element and if the distribution is implemented in a distributed manner, the system implies a local machine and machines that interact with the local machine in the implementation of one (or more than one) global activity [27]. Formula 3 implies that when the load distribution is activated, it follows the time function as in figure 1.

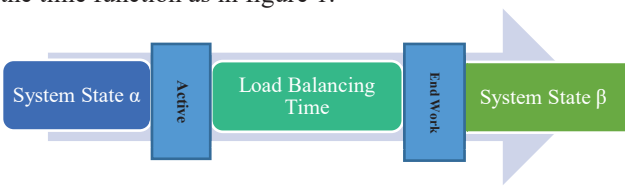

Figure 1: Timeline Activity of load distribution

As shown in Fig. 1, the load distribution is activated as a result of an event or time occurring [9]. The load distribution based on formula 3, based on the status of the system α, as well as the request for processes that, A) caused the occurrence of event B) or during the period when the load distribution was not active, require redistribution and As well the status of resources in the system, it attempts to re-distribute the load in the Load Balancing Time period. During the above period, if a request is arising from either of (A) and (B), until the next active activation of the load distribution, this request is suspended. The load distribution is completed, at the time of End Work. After completion of the load distribution activity in End Work, the load distribution runs the second part of the Formula 3. During the Load Balancing Time, the system load distribution is always equal to the previous state.

As seen in Formula 3, the definition of the load distribution management is such that it considers the $System_{State}$ that it was until it has activated and it does not get information about the status of the system during the Load Balancing period.

3. What Means Dynamic and Interactive in Load Balancing Terminology

The nature of scientific and applied programs running on traditional computing systems is the use of high computational power systems to examine the rules governing a natural occurrence for a particular instance at the least time of response. [28]. This will make it clear computational processing, and interactions in the implementation process, because, in this situation, the parameters associated with the natural event that is using the high-power computing system is known [28]. Determining they define the parameters and the activity means the definite processes of the computational system [28]. On the other hand, communication between the internal parameters of the system is known as well as parameters within the system and the system environment [28]. In general, can say that the purpose of using the traditional computing system is to examine a natural event for specific quantities in the shortest possible time.

While if look at the nature of the scientific and engineering programs needed for Exascale systems will find that the purpose of this type of scientific and engineering program is to discover the rules governing a natural event [20]. In this type of computational system, the purpose of using the computational system is the discovery of unknowns, the extraction of the governing parameters of the natural event and the identifiable activities on them, and the recognition of the connections and interactions between the in-system

parameters as well as the environment and system [22]. In this type of computing system, requirements for computational processes cannot be analyzed and extracted at system design time. Therefore, the computational system is created taking into account the initial demands of computational processes (equivalent to the concept of the basic rules of the particular knowledge domain). Also, the elements of system management, especially the load distribution management, can have a mechanism to deal with when a request occurs that results in a dynamic and interactive state of the system [13, 29]. The dynamic and interactive nature of the computational processes of its Exascale systems results from the occurrence of one (or more than one) of the following three conditions [29]: a) There is a process in the system that attempts to create a process that the process was not defined at the time of designing the computing system. This is due to the formation of a concept in the scientific and engineering program that the computing system is implementing.

b) Process establishing communications and interactions with another process that the connection is not defined at the time of designing the computing system. The issue stems from the formation of a new link in the scientific and engineering program that the computing system is running.

c) The process with another process outside of the computational system establishes communications and interactions that the connection is not defined at the time of designing the computing system. This is due to the formation of the definition of a new variable in the scientific and engineering program that the computing system is implementing.

In the Exascale computing system, the occurrence of any of the three situations results in a change in the state of the computational system [28].

4. Related Works

In general, workload management can be done in three ways: Centralized, hierarchical, or distributed ways [23]. So far, management systems of various HPC programs, such as [30] Condor, Slurm [31] and PBS [32] have allocated workload in a centralized manner. Centralized distributors face constraints like scalability and reliability [23]. Although large-scale distributed equilibrium is well supported for expanding and reliability, it has many challenges that have to be considered. Some fully distributed load balancing strategies have been in resource [23]. One of the major challenges of HPC programs is Load Balancing, which uses millions of computational nodes in Exascale systems to accomplish this. Many of the new science programs have a dynamic workload for some reason, such as time and space adaptation [33]. For this reason, it is necessary to balance the dynamic load to carry out these programs to react to changes in the program and to implement the program with high efficiency [33].

The traditional cluster systems for implementing the new generation of programs face two challenges that are the dynamic nature of the programs and the heterogeneity of the platforms [12, 13]. Exascale computing systems are now an excellent solution to support this type of scientific program [10, 13].

So far, six Real-world HPC applications are being explored to simulate and run by Exascale systems. It can be said that in the absence of Exascale systems or even higher computational power, it would not be possible to simulate these programs [21, 34]. For example, one of these programs is the simulation of the brain, which makes a great jump to better understand the inner workings of the human brain [24]. Climate simulation, considering that the importance of recognizing evolution and global climate change in the 21st century is another HPC program that applies only to Exascale systems [22].

With the design and development of Exascale architecture, this new architecture introduces a group of new computing challenges [20]. Exascale computing has some key challenges that software developers can ignore, Such as the timing and robustness of algorithms and their implementation on millions of processors, data storage and I/O for parallelization, fault tolerance, and power consumption reduction. To solve these challenges, new algorithms are needed that these algorithms must specifically design to solve these challenges to guarantee efficiency and flexibility [20].

In recent years, many dynamic load-balancing algorithms have been proposed. For scientific applications, there is a need for a particular type of load balancing, such as that expressed in CHARM ++ [35]. These load balancers use the migration of work and related data over a lifetime of work [35].

In [13], a distributed dynamic equilibrium load mechanism for Exascale computing systems is presented. The presented method covers the dynamic behavior of new issues. The proposed mechanism for estimating the additional load of nodes uses many parameters such as load transfer and communication latency.

In [33], a load balancing method called the "diffusive" load balancing for a large-scale environment is proposed, which is the minimum load transfer and only requires communication between the neighboring nodes. Distribution load balancing methods have the most effective time and have significant advantages if the load balancing overload such as computational time or the time it takes to transfer workload from source to another is very low.

5. Argument

To understand the effects of the dynamic and interactive events described in Section 3 on the element of load distribution, it needs to consider the situation In which, for any reason, one (or more than one) of the situations listed in Section 3 occurs in the computing system. If the occurrence of an event leads to a dynamic and interactive nature, was before activating the load distribution in Figure 1, In this case, from the system management, occurrence occurs as a) Creation of a new process b) Creating an interconnected and in-system communication or outsource. In either case, the system management element, through the load distribution or resource discovery, attempts to create an accountability structure. This accountability structure does not differ from the perspective of the burden distribution management with the structures of accountability to traditional computing processes.

If an event results in a dynamic and interactive nature is in the Load Balancing Time period shown in Figure 1, then there will be a functional inconsistency between the system state from the perspective of the load distribution management and the system management. From the perspective of the load distribution management, the system status is equivalent to the state of the system now of the Active time, while viewed from the perspective of the system state management system, follows a pattern similar to Figure 2.

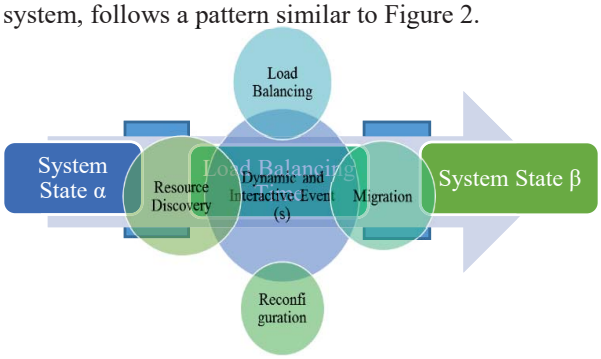

Figure 2. Status of the system from the perspective of the system management in the occurrence of a dynamic and interactive event

As seen in Figure 2, the system has two different situations at a given moment in view of load distribution management and system management. The load distribution management based on the state of the system now of Active, initiate redistribution. However while due to the occurrence of an event leading to an interactive and dynamic situation in the computing system, the system management requires the use of Resource Discovery, Migration, Reconfiguration and Load Balancing management to respond to the dynamic and interactive occurrence of the system.

The reason for this is the functional nature of the load distribution. The load distribution for redistributing loads requires two sets of process status and resource status [10, 11, 12, 13]. The nature of the two sets is such that they must be analyzed in a given moment. If take a look again at Formula 2, will realize that the load distribution formed in traditional computing systems cannot deal with the duality of the system. Two sets of load distribution generators lack the ability to consider the concept of time. $Resource_{Attribute}$Set and $Process_{Requirment}Set$ sets in traditional computing systems at a given time. In this kind of computational system, the nature of these two sets is such that the element of time cannot be considered. In these two sets, the concept of sensitivity to the state of the system cannot be defined because the concept of time is not taken into account. Failure to determine the sensitivity of the system changes causes the two sets to have a static nature. The load distribution management should be based on local information or on the information in the central servicing machine at the moment t = Alpha to set the two sets to be able to define itself according to them.

The Metrics Bench variable in the definition of the load distribution management in traditional systems is discussed in the event of the occurrence or at a given time point. The nature of this variable is such that it cannot be measured continuously. If a computing system wants to measure the Metrics Bench variable continuously, then it will need to spend a lot of computational resources for this variable. On the other hand, the Metrics Bench variable is one of the main variables of the load distribution management for sensitivity to the system state.

$Time_{Space}$ Variable in traditional computing systems, from the perspective of the load distribution management, encompasses all constraints, limitations, and timing abilities associated with the burden distribution management, and its related activities. The nature of this variable is such that it depends only on the load distribution management and does not consider the relationship between the load distribution management and other elements in the system management. This causes the load distribution management to be active at the time of load distribution or redistribution activities in an abstract environment relative to the system. $Time_{Space}$ Variable, however, has the ability to take into account the time challenges associated with the load distribution management, but all the factors and parameters that are discussed in this variable are based on the demands of the processor and the use of the central processor. The events that led to the creation of a dynamic and interactive nature are events that based on the focus on creating a new concept in the system [28]. The two Distribution Pattern

and Error variables depend on the function of the computing system and are defined based on which programs run on this system. The most important challenge for these two variables is nature of their nature of executive. When the load-distribution management at the Active moment begins to initiate redistributive activities, the two variables are initialized and do not change during the processes of redistribution.
The nature of the constituent elements of the definition of the load distribution management in Formula 2 is such that there are no possibilities for the two concepts of time-based definition, as well as sensitivity to events created at the system level during Load Balancing Time.

6. Conclusion

The load distribution is considered as the most pivotal element among the set of elements of computational system management. The nature of this element is such that in the event of a specific occurrence or in certain periods of time, it is attempted to redistribute the load at the system level. The nature of this element is such that when the event triggers its activation in the system, Depending on the two sets of resource status and process requests and also based on its general policy, changing the system states to maximize the use of central processing resources to respond the processes as quickly as possible. The definition of the load distribution in the traditional systems is such that the load distribution only considers all activities between the two periods of its activation time in terms of activities that have the right to operate on them, It does not take into account all the activities performed during the time interval for the distribution of load at the system level. On the other hand, the nature of the programs required by Exascale systems is such that the occurrence of any event leads to a dynamic and interactive state of change. When one (or more than one) events occur leading to a dynamic and interactive state of the system at the time of the load distribution activity, In this case, there is a difference between the actual situation of the system and the situation in which the load distribution is redistributed. The difference is due to the lack of definition of the elements of the load distribution to the time and the lack of consideration of the sensitivity to system state changes. If the element of traditional load distribution management wants to be used in Exascale systems, In this case, it should be possible to make two changes to the definition of the load distribution management element.

References

[1] Valentini, Giorgio Luigi, et al. "An overview of energy efficiency techniques in cluster computing systems." *Cluster Computing* (2013): 1-13.

[2] Sreenivas, Velagapudi, M. Prathap, and Mohammed Kemal. "Load balancing techniques: Major challenge in Cloud Computing-a systematic review." *Electronics and Communication Systems (ICECS), 2014 International Conference on*. IEEE, 2014.

[3] Rahman, Mazedur, Samira Iqbal, and Jerry Gao. "Load balancer as a service in cloud computing." *Service Oriented System Engineering (SOSE), 2014 IEEE 8th International Symposium on*. IEEE, 2014.

[4] Khaneghah, Ehsan Mousavi. "PMamut: runtime flexible resource management framework in scalable distributed system based on nature of request, demand and supply and federalism." U.S. Patent No. 9,613,312. 4 Apr. 2017.

[5] Sharifi, Mohsen, Seyedeh Leili Mirtaheri, and Ehsan Mousavi Khaneghah. "A dynamic framework for integrated management of all types of resources in P2P systems." (2009).

[6] Domanal, Shridhar G., and G. Ram Mohana Reddy. "Optimal load balancing in cloud computing by efficient utilization of virtual machines." *Communication Systems and Networks (COMSNETS), 2014 Sixth International Conference on*. IEEE, 2014.

[7] Papagianni, Chrysa, et al. "On the optimal allocation of virtual resources in cloud computing networks." *IEEE Transactions on Computers* 62.6 (2013): 1060-1071.

[8] Khaneghah, Ehsan Mousavi, Nosratollah Shadnoush, and Amir Hossein Ghobakhlou. "A mathematical model to calculate real cost/performance in software distributed shared memory on computing environments." *The Journal of Supercomputing* 74.4 (2018): 1715-1764.

[9] Singh, Priyanka, Palak Baaga, and Saurabh Gupta. "Assorted Load Balancing Algorithms in Cloud Computing: A Survey." *International Journal of Computer Applications* 143.7 (2016): 34-40.

[10] Mousavi, Seyedmajid, Amir Mosavi, and Annamária R. Varkonyi-Koczy. "A load balancing algorithm for resource allocation in cloud computing." International Conference on Global Research and Education. Springer, Cham, 2017.

[11] Khaneghah, Ehsan Mousavi, Amirhosein Reyhani ShowkatAbad, and Reyhaneh Noorabad Ghahroodi. "Challenges of Process Migration to Support Distributed Exascale Computing Environment." *Proceedings of the 2018 7th International Conference on Software and Computer Applications*. ACM, 2018.

[12] Khaneghah, Ehsan Mousavi, Reyhaneh Noorabad Ghahroodi, and Amirhosein Reyhani ShowkatAbad. "A mathematical multi-dimensional mechanism to improve process migration efficiency in peer-to-peer computing environments." *Cogent Engineering* 5.1 (2018): 1458434.

[13] Mirtaheri, Seyedeh Leili, and Lucio Grandinetti. "Dynamic load balancing in distributed exascale computing systems." *Cluster Computing* (2017): 1-13.

[14] Barak, Amnon, et al. "Resilient gossip algorithms for collecting online management information in exascale clusters." *Concurrency and Computation: Practice and Experience* 27.17 (2015): 4797-4818.

[15] Hussain, Hameed, et al. "A survey on resource allocation in high performance distributed computing systems." *Parallel Computing* 39.11 (2013): 709-736.

[16] Zhou, Amelie Chi, Bingsheng He, and Shadi Ibrahim. "A Taxonomy and Survey of Scientific Computing in the Cloud." (2016).

[17] Deelman, Ewa, et al. "Pegasus, a workflow management system for science automation." *Future Generation Computer Systems* 46 (2015): 17-35.

[18] Ogrizović, Dario, Zlatan Car, and Božidar Kovačić. "Scientific Applications in Cloud Computing." *The IPSI BgD Transactions on Advanced Research* 10.1 (2014): 27-33.

[19] Choudhury, Anamitra R., et al. "Method for improving the performance of high performance computing applications on cloud using integrated load balancing." U.S. Patent No. 9,021,477. 28 Apr. 2015.

[20] Alowayyed, Saad, et al. "Multiscale Computing in the Exascale Era." *arXiv preprint arXiv:1612.02467* (2016).

[21] Reylé, C., et al. "Perspectives In Numerical Astrophysics: Towards An Exciting Future In The Exascale Era."

[22] Innocenti, Maria Elena, et al. "Progress towards physics-based space weather forecasting with exascale computing." *Advances in Engineering Software* (2016).

[23] Wang, Ke, Anupam Rajendran, and Ioan Raicu. "MATRIX: MAny-Task computing execution fabRIc at eXascale." *Tech Report, IIT* (2013).

[24] Amunts, Katrin, et al. "The Human Brain Project: creating a European research infrastructure to decode the human brain." *Neuron* 92.3 (2016): 574-581.

[25] Lian, Jiunn-Woei, David C. Yen, and Yen-Ting Wang. "An exploratory study to understand the critical factors affecting the decision to adopt cloud computing in Taiwan hospital." *International Journal of Information Management* 34.1 (2014): 28-36.

[26] Choudhury, Anamitra R., et al. "System for improving the performance of high performance computing applications on cloud using integrated load balancing." U.S. Patent No. 9,015,708. 21 Apr. 2015.

[27] Katyal, Mayanka, and Atul Mishra. "A comparative study of load balancing algorithms in cloud computing environment." *arXiv preprint arXiv:1403.6918* (2014).

[28] Khaneghah, Ehsan Mousavi, and Mohsen Sharifi. "AMRC: an algebraic model for reconfiguration of high performance cluster computing systems at runtime." *The Journal of Supercomputing* 67.1 (2014): 1-30.

[29] Mirtaheri, Seyedeh Leili, et al. "A mathematical model for empowerment of Beowulf clusters for exascale computing." *High Performance Computing and Simulation (HPCS), 2013 International Conference on*. IEEE, 2013.

[30] Alnoshan, Abdullah. *Microeconomic Approach to P2P Resource Allocation*. Diss. The George Washington University, 2014.

[31] Buczkowski, Steven. "Using MySQL for load balancing and job control under slurm." *Linux Journal* 2015.256 (2015): 3.

[32] Wang, Ke, et al. "Next generation job management systems for extreme-scale ensemble computing." *Proceedings of the 23rd international symposium on High-performance parallel and distributed computing*. ACM, 2014.

[33] Lieber, Matthias, Kerstin Gößner, and Wolfgang E. Nagel. "The Potential of Diffusive Load Balancing at Large Scale." *Proceedings of the 23rd European MPI Users' Group Meeting*. ACM, 2016.

[34] www.deep-project.eu (last seen at May 2018)

[35] Jeannot, Emmanuel, Guillaume Mercier, and François Tessier. "Topology and affinity aware hierarchical and distributed load-balancing in Charm++." *Communication Optimizations in HPC (COMHPC), International Workshop on*. IEEE, 2016.