Challenges of Execution Trend in Distributed Exascale System

Hamid Gharb*[1], Ehsan Mousavi Khaneghah[1], Faezeh Mollasalehi[1], Shirin Shahrabi[1]

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

تاریخچه مقاله:

تاریخ ارسال: ۹۷/۳/۱

تاریخ اصلاحات:۹۷/۱۲/۱

تاریخ پذیرش:۹۸/۱/۳۰

تاریخ انتشار:۹۸/۵/۱۵

Keywords:

Distributed Exascale-Computing System
Execution Trend
Dynamic and-Interactive Nature
Configuration
Load balancing

Abstract

In computing systems, the execution trend pattern for the application indicates the executing trend of the application based on the application features, as well as the ability of the system elements to reduce the response time. In traditional computing systems, due to the clear nature of the application, system designer is able to determine its implementation pattern when designing the system and to establish this execution pattern at runtime by changing system manager’s constituent elements. In distributed Exascale systems, the occurrence of a dynamic and interactive nature changes the processes and system elements. While investigating the effect of the dynamic and interactive nature of distributed Exascale systems on determining the execution pattern, this paper also discusses the challenges that exist to establish this concept in distributed Exascale systems. It also states why the determinant element of execution pattern is required in system manager and suggests some strategies to manage these challenges in distributed Exascale systems.

روش ارجاع به مقاله: H.Gharb, E.Mousavi Khaneghah, F.Mollasalehi, S.Shahrabi, Challenges of Execution Trend in Distributed Exascale System, Journal of Distributed Computing and Systems (JDCS) Vol.2, No.1, PP. 140-151, 2019

Email: H.Gharb@Shahed.ac.ir

1. Introduction

In high performance computing systems, system manager must be able to manage the system based on a model and in such a way to answer the application in the shortest possible time [1, 2, 3]. This can be achieved when system manager controls and manages the processes and resources based on a specific pattern [4, 5]. The execution pattern in traditional computing systems should be designed and executed based on the application features as well as the computing system capabilities [6]. The designer of the computing and processing system extracts this pattern and the constituent elements of the computing system's manger should be able to establish it at runtime [6]. Each of element in system manager has to establish the pattern defined by the system designer through managing the functionality and behavior of existing processes or resources in the computing system [7, 8, 9].

The execution pattern in computing systems can be designed based on one of the two policies of purpose or application and system [6]. If designing the execution pattern is based on the purpose policy, the designer of the computing system will design a model based on the purpose of executing the application in the computing system. In application and system policy, system designer will design a model based on the application features as well as the system's capabilities. In either case, each system must have an execution pattern at any moment [6]. Also, at any moment of running the system, an element (or elements) must be defined in the system that is (are) in charge of establishing the execution pattern [6]. The execution pattern can only be designed when the system is being designed [6]. Traditional computing systems use this method [6, 10]. The designer of the computing system must have a clear vision of the purpose or features of the application and system capabilities in order to be able to use the concept of designing the execution pattern when designing the system. In the absence of a clear vision of the above, the system designer will not be able to use this pattern. In this situation, an element must be defined in the system manager whose task is to define the system pattern based on one of the two policies.

One of the most important concepts influencing the execution pattern is the diversity or unity of requests in the system [11]. The nature of the execution pattern must be such that it can ultimately meet the requests in the system [6]. Regardless of whether the execution pattern is designed at the time of designing the system or during running the system operations, there must be an element (or elements) in the system that can maintain the executive pattern in the system. Typically, the elements forming the system manager are responsible for this task [3, 6]. This element (or elements) must be able to answer requests based on the request pattern. In traditional computing systems, typically the requests in the system are to gain access to the computing resource in order to reduce the response time [3]. As a result, all the elements forming the system manager and the element (or elements) establishing the concept of execution pattern will be designed to answer the requests with the nature of gaining access to the computing resources aimed at reducing the response time [6].

Another important element that influences the implementation pattern is the resource diversity or unity [12]. The nature of the execution pattern is such that it must be able to ultimately meet the objective of implementing the activity by managing the resources in the system [6]. Regardless of which policy determines the execution pattern, it should be able to perceive the concept of the resource, and more importantly, the concept of using the resource. The execution pattern should define a set of resource descriptor indicators and also the state of using the resource, so that it can accordingly manage the execution trend of the activity [6]. In traditional computing systems, the only resource that the application needs, and consequently, can be managed by the implementation pattern, is the computing resource [6, 12, 13]. Defining the activities of system manager elements is also based on the computing resource. The indicators describing the resource state are also based on the extent of using the resource by computing processes [6].

In distributed Exascale computing systems, due to the occurrence of a dynamic and interactive nature, both the application and the constituent elements of the computing system are changing during the runtime of the application [14, 15, 16]. This changes the executing pattern of the application during its runtime [6]. Changing the implementation pattern requires defining a new element in the architecture of system manager called the execution pattern determinant. The concepts used by execution pattern determinant in distributed Exascale computing systems are dependent on the concept of executive pattern in these types of computing system. While investigating the concept of dynamic and interactive nature and its impact on the concept of execution pattern, this paper also investigates the challenges of the determinant element of execution pattern in these computing systems.

2. Related work

This section examines the works carried out in the field of the execution pattern determinant or configuration of computing systems at runtime.

In [6] proposes a definition of configuration. According to the definition, configuration is the result of cooperation between software and hardware elites and the problem itself. With the help of configuration, fixed values, constraints, and features of a problem are generated [6]. Then, with the help of these generated values, HPC system is implemented on the computing node in order to obtain the optimum computing power [6]. Traditional HPC systems have the best problem-solving efficiency, and the reason is that they can be configured before starting to solve the problem [6]. Configuration in traditional computing systems is static [6].

Paper [17] refers to the configurable calculations from the operating system perspective. The general scenarios for configurable logics introduced in this paper include [17]:

1. Application, as an accelerator, supports a general-purpose processor by implementing computing functions.
2. Application, as a parallel collaborative processing element, cooperates with its neighbors.
3. Application, as an additional and independent processing element, not only executes computing functions, but also performs its own applications.

In paper [17], the operating system services are stated as the key to support configurable computations in the operating system and their relationship with configurable computations have been separately examined.

Applications are typically developed in distributed environments such as Grid or Cloud Infrastructure, where computing resources are flexible, and more importantly, the elements of different applications can be implemented in a parallel form [18]. Cloud-based applications must be dynamically reconfigured to adapt quickly to new conditions and support service quality requirement and avoid long downtime [18].

In [18], a hierarchical component-based framework is provided for configurable structures and simplifying distributed applications in behavior automatic programming. This framework is a distributed client-server application with high scalability. Applications are designed based on components to be reconfigurable and compatible [18].

The purpose of component-based systems, such as Fractal and GCM, is to facilitate repeatability and are especially designed to have less error-prone reconfiguration [18].

In [6], a method for dynamic reconfiguration is presented which is derived from the runtime programming model. This paper shows that by combining these elements, a task can expand itself in a way to use the ideal computing nodes [6].

In [19], the author has presented a model for dynamic reconfiguration of task scheduler parameters, through which the scheduler obtains information on the actual condition of the physical machine on which the calculations are performed. The paper combines online route allocation scheduling with a reconfiguration technique, which allows an element almost freely balance its own resources and scales its almost free tasks [19]. The reconfiguration performed in this paper has increased the overall system efficiency [19].

In the next generation of HPCs, applications face more complexities and more unpredictable requests and they will require more powerful systems with greater scales [15, 19, 20, 21]. The purpose of using traditional computing systems was to examine a natural event for certain values in the shortest possible time [11]. However, if we look at the nature of scientific and engineering applications that require Exascale systems, we will realize that the purpose of running these applications is to discover the rules governing a natural event [11]. One of the strategies available for HPC systems to support applications with a dynamic and interactive nature is to increase their scalability during runtime [22]. Paper [11] addresses the issue that how multiscale computing systems can be optimally used in existing petascale systems as well as HPC Exascale systems by increasing their capacity.

Due to the dynamic nature of running applications in an Exascale computing system environment, the initial configuration cannot answer dynamic changes of the problem and a mismatch occurs in the configuration [6]. To resolve this mismatch, a reconfiguration will be required to re-adapt the computing system to the requirements of the running process [6].

Generally, two strategies have been proposed to solve this problem [6]:

A. Using dynamic load balancing: The importance of using a dynamic LB, versus the static one, is that with the help of dynamic LB, it is possible to rebalance the computing load on the nodes when the ratio of resources to the computing process requirements has been unbalanced. Using this mechanism, dynamic property is included in HPC systems, so that it will show a dynamic behavior.
B. Using Configurable Platforms: With these configurable platforms, the configuration of HPC systems can be altered at runtime in such a way that changes can be supported by the platform on which the HPC system is implementing. Using these platforms, the dynamic property will be added to the HPC system. These platforms are important because they help to use traditional high performance systems to solve problems with a dynamic structure.

3. What do execution trends mean?

In HPC systems, the execution pattern refers to the execution trend of the application [6]. This pattern determines how to implement the constituent processes of the application and how to manage resources in the system to meet the objectives of the computing system [2, 6, 23, 24]. In addition to the general purpose of the computing system, the execution pattern should be able to take into account the minor goals of running processes [6, 23, 25]. This pattern is typically considered as a time function whose independent variable is the system runtime and its dependent variables are the behavior and functionality of the system's basic elements. In computing systems, execution pattern can be described and defined as Eq.1.

$$F(Execution\ Trends): \left(Manage(Basic\ Elements) \overset{HPC}{\overset{\frown}{\rightarrow}} Do\ Operation \right)$$

Eq.1

As shown in Eq.1, the execution pattern indicates how basic elements are managed to achieve the ultimate goal of the system which is being implemented. In computing systems, the ultimate goal of performing activities related to the application and the execution conditions is carry out the activities in the shortest possible time [2, 26, 27]. The system's basic elements are dependent on the system execution pattern. Independent variable, in the function shown in Eq.1, is typically time-based, however, other independent variables can be considered for this purpose. The dependent variable of Eq.1 is the system's basic element. In HPC systems, two elements of process and resource are conventionally considered as basic elements [4, 5, 6]. The basic element in computing systems is to a large extent dependent on the nature of the application.

As can be seen in Eq.1, the concept of execution pattern implies that during runtime, from the moment the computing system starts operating until it completes the activity, based on what pattern the application must be implemented. This pattern determines which elements should be defined in system manager [6]. This pattern also suggest what tasks should the elements of system manager be in charge of [7, 8, 9]. Given Eq.1 and defining the constituent elements of system manager, the generating space of the execution pattern element can be stated as Eq.2 and based on the execution pattern.

$$Execution\ Trends < Basic\ Elements, Target, < HPC\ Manager >> \qquad \text{Eq.2}$$

As shown in Eq.2, the execution pattern space is based on basic elements, with Target feature, and uses the elements of system manager. In Eq.2, the concept of applications and its effects on the execution pattern are implicitly considered in two concepts of Basic Elements and Target. The nature of the application, for both Basic Elements and Target concept is considered as the beneficiary element and influences the two concepts. Fig.1 shows a schema of creating the execution pattern space in traditional computing systems.

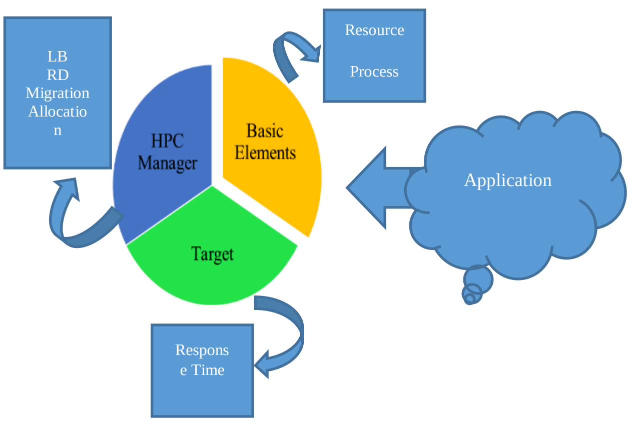

Figure.1. the space of creating the execution pattern in traditional computing systems

As can be seen in Fig.1, in traditional computing systems, basic elements are the process and source, and the goal is to run the program in the shortest possible time, and the system manager is quadruple, consisting of resource discovery, load balancing, migration, and load allocation [3, 26, 27, 28, 29, 30]. In this type of computing system, the procedure of allocating resources to processes should reduce response time [3, 26, 27]. LB manager is selected based on the nature of the application as well as the execution pattern [6, 31].

Closed traditional computing systems, such as cluster computing systems, due to not altering the nature of the application and not altering the computing elements of the system, enables the computing system designer to, through optimally adapting the process requirements to resource properties, obtain the optimal execution pattern when designing the system, and accordingly implement the application in cluster computing system [6, 10, 32].Due to the lack of changes in both basic elements of the computing system, the execution pattern does not change during the runtime of the application [6, 10, 32, 33]. Therefore, the system designer, based on the application execution pattern, selects a mechanism for LB manager that can establish the execution pattern in the system during the application runtime [6, 12, 34]. LB manager is considered as the basic element of system manager and other elements are designed according to the functionality and nature of the algorithm and the mechanism used by LB manager [7, 35, 36].

In open traditional computing systems, such as peer-to-peer and grid computing systems, the nature of the application does not change over the course of its runtime [34, 37]. In these types of computing systems, the system designer does not consider all the resources required to run the application when designing the computing system and during the runtime, appropriate to the requirements of the computing processes, adds or removes computing elements to or from the system during the implementation of the application [38]. Thus, system's computing elements alter during runtime [34, 37]. Changes in system's computing elements in this type of computing system will not cause change in the execution pattern extracted at the time of system designing. As a result of not having changes in the application nature and the execution pattern due to changes in computing elements, system manager will establish the execution pattern at runtime based on LB and RD [12]. The reason for lack of changes in the execution pattern due to changes in the computing elements is that system designer has considered a computing element appropriate to the computing process requests when designing the execution pattern, and based on the mechanism specified for RD manager at runtime, will provide the processes with the process required resources [6, 12]. At system runtime, the computing resource that has not been considered when designing the execution pattern, will not be added to the computing system [34, 37].

The most important challenge in open computing systems, such as Grid and peer-to-peer, is the ability to simultaneously run multiple applications [39, 40]. In these types of computing systems, the system manager has to be able to simultaneously run several applications. To this end, system manager will operate based on one of two policies of community space or total match. In community space policy, the computing system manager determines the mechanisms and algorithms used by elements of the system manger. System manager's criterion is that the mechanisms and algorithms need to be able to support the execution patterns for all applications that run on the computing system. If this policy is used, typically, each constituent element of system manager will follow a specific mechanism. This mechanism is able to support a specific range of the execution pattern. In total match policy, for each type of application with a specific execution pattern, an element which performs a specific activity in system manager is defined. In case of using this policy, each element of system manager will be composed of a number of sub-elements that are designed to perform the related activities based on a specific execution pattern.

4. Dynamic and Interactive influence on Execution Pattern

In distributed Exascale systems, the nature of the application is such that they try to discover the dominant rules in a natural event [11]. In these types of computing systems, the application is formed based on a set of assumptions, and during its execution trend, examines and analyzes the rules governing the event [11]. Because of this, system designer lacks a precise vision of the application purpose. On the other hand, system designer lacks a precise vision of the application features [15, 19, 20, 21, 41]. In distributed Exascale systems, there is a concept called dynamic and interactive nature [6, 21, 41]. The occurrence of a dynamic and interactive nature creates new requirements and requests in the system that may not be taken into account when designing the initial structure of the execution pattern [21, 41]. Thus, at the time of designing, system designer does not have a detailed view on the functional nature of system elements. Consequently, in these computing systems it is not possible to use the policy of designing the execution pattern at the time of system designing. Therefore, there must be an element in the structure of system manager that can create an execution pattern based on the policy of application and system. It is not possible to use the target policy because the application cannot consider the time prior to runtime [11]. Therefore, this element should be able to form the execution pattern structure by analyzing the application features and the capabilities of the current computing system.

In these types of computing systems, the concept of dynamic and interactive nature alters the nature of process requests during the application implementation [6, 11, 21, 41, 42]. In these systems, due to the application nature, as well as the dynamic and interactive nature, process requests do not solely focus on the computing resource and the process can create other requests at any moment to continue its activity [6, 12]. These requests may have a nature different from processing requests. The most important challenge in this regard is that the process may have a request that is not considered in the execution pattern. This request has to be considered as a dynamic and interactive request from system manager's perspective [21, 41, 42]. Two general policies can be defined for managing these types of requests. In the first policy, called empowerment policy, system manager's element (or elements) must be modified so that they can implement and manage dynamic and interactive requests [6, 12, 14]. In this case, each element of the constituent elements of system manager creates mechanisms to support the concept of dynamic and interactive nature and to respond the requests formed based on a dynamic and interactive nature. In the second policy, referred to as changing the execution pattern, based on a set of indicators, system manager must be able to describe the functional and behavioral state of the application as the cause of the request. These indicators need to be able to provide a detailed view on the application state at a specific moment for the execution pattern determinant. In this case, in accordance with the indicators describing the application, the execution pattern determinant can either use vectors algebra and return the application to its initial state or use graph algebra and determine the schemas or use the correlation pattern of the indicators and find the closest application that matches to the new indicators' state of the application in order to manage and determine the execution pattern. The execution pattern determinant unit has to define the set of indicators in such a way that it can make decisions based on them when changing the application's execution pattern and determining the new state.

In these types of computing systems, due to the nature of the application as well as the dynamic and interactive nature, processes may require resources other than computing resources to continue their operations [12, 21, 41, 42]. For this reason, system manger in these systems needs to manage a variety of resources, provide classifications for resources, and consequently, manage requests for gaining access to different resources [12]. In these systems, requests from resources may be such that the process may request access to a resource that cannot be answered by the system [15, 21]. In this case, system manager must be able to answer this request by changing system's computing elements and limitations. This will change the system state from system manager's perspective. When the system state changes, it may be necessary to change the execution pattern. Similarly, in situations where a dynamic and interactive nature occurs and the process requirements change, either policies of system manager empowerment in managing resources or changing the execution pattern can also be used. In either case, the execution pattern determinant unit must have a set of indicators that on its basis, decisions can be made on the resources' state and their changes that alter the execution pattern. The concept of system's resource change has also been proposed as system's scalability in traditional computing systems [12, 43]. Therefore, the execution pattern determinant unit must be able to define indicators in a way that scalability due to the change of the execution pattern differs from the standard scalability.

5. Argument

The occurrence of a dynamic and interactive nature in the computing processes running in distributed Exascale systems, causes changes in the request nature and consequently, the state of the requested resources [12, 21, 41]. In traditional computing systems, the computing system is defined based on two basic elements of process and resource [4, 5, 6]. As a result: a) need to be defined independent and separate indicators to examine the resource and process state and their impact on the execution pattern. These indicators, while describing the state of these two basic elements, should also be able to describe the execution pattern change that results from two basic element changes. b) When the execution pattern determinant unit examines the system at t = Alpha, it analyzes the two basic elements regardless of their previous state. Because of this, their execution trend as well as the activities carried out by them are not considered in determining the execution pattern. Thus, to solve these challenges, the concept of global activity for two basic elements of resource and process is used in distributed Exascale systems [12]. The concept of global activity includes the history of performed operations and activities, as well as inter-system and intra-system interactions and communications of the activity [12, 42]. When the execution pattern determinant unit examines the global activity state to change the execution pattern or to check that this pattern has changed, the examination includes all the stated items. The execution pattern determinant unit in these systems should be able to use a pattern for resource management. The existence of a wide range of process's requested resources, necessitates using a pattern for resource management. In [12], based on the classification pattern of the operating system resources, and considering that each computing element in distributed systems must have its independent operating system, a four-category operating system has been used. As a result, by creating a concept called regions, system manager can create computing elements with higher capabilities and higher relative advantage in responding to requests of a particular resource [12]. Using the concept of relative advantage will change the response region for creating a new pattern if the execution pattern changes due to requiring resources [12].

Also, the execution pattern determinant unit should be able to classify the requests of forming the global activity based on a concept similar to the one stated in [12]. Using the classification pattern of requests of global activity processes allows the execution pattern determinant unit to use the concept of the execution pattern similarity to determine the execution regions.

6. Conclusion

In distributed Exascale systems, the occurrence of a dynamic and interactive nature introduces the concept of requiring the execution pattern determinant unit. Execution patterns are defined based on application features and system capabilities. In distributed Exascale systems, both concepts of application features and system capabilities and also the constituent elements of the system are changing during runtime. Execution patterns in these types of computing systems should change at runtime and when the state of each of these two factors is changing. While examining the effect of dynamic and interactive nature on determining the executive patterns, this paper also indicated that these systems require the execution pattern determinant unit, which this element needs to change the factors affecting the space of definition the execution pattern in order to examine the need for changing the execution pattern and create a new pattern. The need for concepts such as global activity, execution regions of resources, as well as classification of resources and requests are the most important solutions used by this management element to determine the execution pattern.

7. References

[1] Wallace, Sean. *Power Profiling, Analysis, Learning, and Management for High-Performance Computing*. Illinois Institute of Technology, 2017.

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

[3] Khaneghah, Ehsan Mousavi, et al. "A Dynamic replication mechanism to reduce response-time of I/O operations in high performance computing clusters." *Social Computing (SocialCom), 2013 International Conference on*. IEEE, 2013.

[4] Kołodziej, Joanna, et al. "Security, energy, and performance-aware resource allocation mechanisms for computational grids." *Future Generation Computer Systems* 31 (2014): 77-92.

[5] Qureshi, Muhammad Bilal, et al. "Survey on grid resource allocation mechanisms." *Journal of Grid Computing* 12.2 (2014): 399-441.

[6] 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.

[7] Amelina, Natalia, et al. "Approximate consensus in stochastic networks with application to load balancing." *IEEE Transactions on Information Theory* 61.4 (2015): 1739-1752.

[8] 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.

[9] 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.

[10] 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.

[11] Alowayyed, Saad, et al. "Multiscale computing in the exascale era." *Journal of Computational Science* 22 (2017): 15-25.

[12] 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.

[13] Monsalve, Jose, Aaron Landwehr, and Michela Taufer. "Dynamic CPU resource allocation in containerized cloud environments." *Cluster Computing (CLUSTER), 2015 IEEE International Conference on*. IEEE, 2015.

[14] 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.

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

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

[17] Eckert, Marcel, et al. "Operating system concepts for reconfigurable computing: review and survey." *International Journal of Reconfigurable Computing* 2016 (2016).

[18] Ibáñez, Matías, et al. "Reconfigurable Applications Using GCMScript." *IEEE cloud computing* 3.3 (2016): 30-39.

[19] Khorandi, Sina Mahmoodi, Siavash Ghiasvand, and Mohsen Sharifi. "Reducing Load Imbalance of Virtual Clusters via Reconfiguration and Adaptive Job Scheduling." *Proceedings of the 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*. IEEE Press, 2017.

[20] Mirtaheri, Seyedeh Leili, Seyed Arman Fatemi, and Lucio Grandinetti. "Adaptive Load Balancing Dashboard in Dynamic Distributed Systems." *Supercomputing Frontiers and Innovations* 4.4 (2017): 34-49.

[21] Wang, Ke, et al. "Load- balanced and locality- aware scheduling for data-intensive workloads at extreme scales." *Concurrency and Computation: Practice and Experience* 28.1 (2016): 70-94.

[22] Abraham, Erika, et al. "Preparing HPC applications for exascale: Challenges and recommendations." *Network-Based Information Systems (NBiS), 2015 18th International Conference on*. IEEE, 2015.

[23] 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.

[24] Rubab, Saddaf, et al. "Grid Computing in Light of Resource Management Systems: A Review." (2015).

[25] Khaneghah, Ehsan Mousavi, and Seyed Ali Ghoreishi. "CGUW: A system software for heterogeneous IPC mechanism in grid computing environments." *Engineering, Technology and Innovation (ICE/ITMC), 2017 International Conference on*. IEEE, 2017.

[26] Alawneh, Luay, Abdelwahab Hamou-Lhadj, and Jameleddine Hassine. "Segmenting large traces of inter-process communication with a focus on high performance computing systems." *Journal of Systems and Software* 120 (2016): 1-16.

[27] Khaneghah, Ehsan Mousavi, Nosratollah Shadnoush, and Amin Salem. "Artemis time: A mathematical model to calculate maximum acceptable waiting time in B2C e-commerce." *Cogent Business & Management* 4.1 (2017): 1405509.

[28] Rathore, Neeraj, and Inderveer Chana. "Load balancing and job migration techniques in grid: a survey of recent trends." *Wireless personal communications* 79.3 (2014): 2089-2125.

[29] Navimipour, Nima Jafari, et al. "Resource discovery mechanisms in grid systems: A survey." Journal of Network and Computer Applications 41 (2014): 389-410.

[30] Jiang, Yichuan. "A survey of task allocation and load balancing in distributed systems." *IEEE Transactions on Parallel and Distributed Systems* 27.2 (2016): 585-599.

[31] Soltani, Narjes, et al. "A dynamic popularity-aware load balancing algorithm for structured p2p systems." *IFIP International Conference on Network and Parallel Computing*. Springer, Berlin, Heidelberg, 2012.

[32] Yang, Chao-Tung, et al. "On improvement of cloud virtual machine availability with virtualization fault tolerance mechanism." *The Journal of Supercomputing* 69.3 (2014): 1103-1122.

[33] Soni, Ashish, Gagan Vishwakarma, and Yogendra Kumar Jain. "A bee colony based multi-objective load balancing technique for cloud computing environment." International Journal of Computer Applications 114.4 (2015).

[34] Ramezani, Fahimeh, Jie Lu, and Farookh Khadeer Hussain. "Task-based system load balancing in cloud computing using particle swarm optimization." *International journal of parallel programming* 42.5 (2014): 739-754.

[35] Mirtaheri, Seyedeh Leili, et al. "Four-dimensional model for describing the status of peers in peer-to-peer distributed systems." *Turkish Journal of Electrical Engineering & Computer Sciences* 21.6 (2013): 1646-1664.

[36] Silberschatz, Abraham, Peter Baer Galvin, and Greg Gagne. Operating system concepts essentials. John Wiley & Sons, Inc., 2014.

[37] Liu, Ji, et al. "A survey of data-intensive scientific workflow management." *Journal of Grid Computing* 13.4 (2015): 457-493.

[38] Navaz, K., and Kannan Balasubramanian. "Multicast due date round-robin scheduling algorithm for input-queued switches." International Journal of Computer Network and Information Security 8.2 (2016): 56.

[39] Biswas, Tarun, Pratyay Kuila, and Anjan Kumar Ray. "Multi-level queue for task scheduling in heterogeneous distributed computing system." *Advanced Computing and Communication Systems (ICACCS), 2017 4th International Conference on*. IEEE, 2017.

[40] Cesario, Eugenio, Carlo Mastroianni, and Domenico Talia. "Distributed volunteer computing for solving ensemble learning problems." *Future Generation Computer Systems* 54 (2016): 68-78.

[41] http://www.deep-er.eu (last seen August 4)

[42] 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.

[43] Kaur, Kiranjot, and Anjandeep Kaur Rai. "A comparative analysis: Grid, cluster and cloud computing." International Journal of Advanced Research in Computer and Communication Engineering 3.3 (2014): 5730-5734.