Challenges of Process Migration to Support Distributed Exascale Computing Environment

Ehsan Mousavi Khaneghah
Department of Computer Engineering, Faculty of Engineering, Shahed University,
(+98) 21-51212046
EMousavi@Shahed.ac.ir

Amirhosein Reyhani ShowkatAbad
Department of Computer Engineering, Faculty of Engineering, Shahed University,
(+98) 21-51212046
Amirhoseinreyhani75@gmail.com

Reyhaneh Noorabad Ghahroodi
Department of Computer Engineering, Faculty of Engineering, Shahed University,
(+98) 21-51212046
RNoorabad@hotmail.com

ABSTRACT
New scientific and engineering programs, due to the complexity of the structure and the existence of dynamic and interactive requests in nature of the program, require distributed Exascale computing systems. New scientific and engineering programs by providing new parameters during the process execution are demanded distributed Exascale computing systems. The dynamic and interactive nature of the process makes the system management element a dynamic model for managing computing activity. In this paper by analyzing the reasons for the dynamic and interactive requirements, a new model for process migration is represented. Adding <Viability, Dependency> parameters to the process migration definition makes the process migration be able to be used in the distributed Exascale computing system.

CCS Concepts
• **Computing methodologies→ Distributed computing methodologies**

Keywords
Distributed Exascale Computing; Process Migration; Viability, Dependency,

1. INTRODUCTION
Programs that are demanded to Distributed Exascale computing systems (DECS) require different computing systems rather than traditional scientific programs [1, 2, 3]. Due to the nature of computing processes, which are used in the explained programs, managing and controlling them require much complex managing structures. The complex structure of programs requiring DECS makes it possible to witness the emergence of dynamic and interactive applications. The nature of the dynamic and interactive requests is such that it can only be made to create accountability structures after the occurrence of the related events [4]. The system manager must be able to implement and manage these types of scientific and engineering programs by supporting flexible and dynamic accountability structures.

In traditional scientific programs, due to non-having interactive and dynamic requirements, the computing system's process migration can begin to manage the system, based on a specific and deterministic set of rules [5]. Specific and deterministic set of rules are used mechanisms, by the system manager, to response the process requirements when it occurs an event [5]. The deterministic nature of scientific and engineering programs in these types of systems is such that it is possible to construct responsive structures based on patterns at the start time of the executing computing program or by the process information [5, 6]. The discussed issue lets the managing components such as load distribution element and the process migration be able to control and manage the execution of the program based on the deterministic and specific patterns [5].

In DECS, because of interactive and dynamic requirements, the concept, effects, and influences of these requirements should be studied. This issue will illustrate that whether available traditional mechanisms in the system can execute the scientific and engineering programs or not? The main effect of dynamic and interactive nature on processes is the enhancement of the time of interaction among processes and the system manager [7, 8]. This problem will cause two fundamental elements: the load distribution element and the process migration, which most effects cause on them.

It is clear that a scientific and engineering program has a long execution time in computing systems [9]. If the system manager was able to extract the governing pattern of the computing processes, based on user information or the process execution's trend, it could use a specific and deterministic set of rules for managing the system. The existence of specific mechanisms for the process migration is a clear indication of this issue. The main challenge of the computing process during the transmission time is the inability to respond to the process communication mechanisms [10]. For this reason, the available process migration mechanisms are largely base on reducing the transmission time.

When a process is selected based on load distribution mechanisms for any reason, during the process transmission duration, no occurrence leads to breaking the main aim of the load distribution element [5, 9]. The load distribution element begins to migrate a process when one of these two reasons happen: A) using the maximum power of the central processing unit B) Failure to execute the process for a specified period in the computing system. To reach this goal, the load distribution element calls for the process migration [10]. If the process does not have a dynamic and interactive nature, then during the process transmission trend, the process will not have requests

ICSCA 2018, February 8–10, 2018, Kuantan, Malaysia

ACM ISBN 978-1-4503-5414-1/18/02…$15.00
https://doi.org/10.1145/3185089.3185098

that conflict with the purpose of the load distribution element. Dynamic and interactive request creation, at the time of process migration trend, will make the process migration manager cope with a new set of challenges, apart from traditional challenges (such as reducing transmission time, reducing costs, reducing dependencies). This paper is studying the challenges of traditional process migration mechanism used in the DECS.

2. RELATED WORK

To understand challenges in the DECS, process migration requires a detailed analysis of the traditional process migration mechanisms to support programs requiring DECS. This analysis allows us to decide on which patterns the process migration uses in the system, and this pattern will support which processes? [10, 11, 12, 13, 14, 15] On the other hand, gathering information about programs that are demanded to DECS to analyze the events that lead to the creation of a dynamic and interactive nature is needed [16].

The activities in the field of process migration mechanisms are classified into two general categories.

The first class of this classification is all the mechanisms that the process is not able to receive any messages from other processes and rejects all messages sent to the suspended process, during the process transmission [10, 11]. These types of mechanisms have two different categories:

The first group is operating systems that use several process migration mechanisms to transmit the process [10, 11]. In this kind of systems, the process migration manager tries to decide on which process migration mechanism is appropriate to the current state based on the information of processes data structures [11]. In this type of systems, the nature of the process requests does not change during the process transmission. To clarify and to understand this issue, consider a process is requesting time allocation from the central processing unit. If the local system does not answer the request, will be passed to the load distribution element. The load distribution element selects the destination machine and calls the process migration to transmit the process. The process migration decides to use which process migration mechanism only based on the information of the process data structures that are related to the time [10]. The concepts of allocated time by the central processing unit to the processes and the nature of interprocess communications and interactions are fixed and do not change during the process transmission.

The second group is represented mechanisms, which extracts a mechanism by combining other available strategies or based on the characteristics of the four basic strategies [12, 13, 14, 15]. In this type of system, the goal is to provide a mechanism that performs the process migration with the least cost and in the least amount of time [13, 15]. When a process starts the process migration trend based on the set of parameters, which contains the interprocess communication type, time, and cost of migration, the set of parameters is fixed and does not change the process migration trend [12].

The second class is all mechanisms that let the process be alive during the transmission time [17, 18, 19]. Being alive means that the suspended process will be able to receive messages from other processes [19, 20]. In this kind of systems, the main purpose of the process migration is to transfer the process by maintaining the possibility of inter-process communication [17, 19]. During the process migration, only requests for inter-process communications and interactions that were defined before the process migration trend are answered [18]. The nature of these mechanisms is such that if a new process is created that did not exist before the process migration started the request for communications and interactions between the new process and the transmitting process would be rejected.

Another kind of live transmission is discussed in Cloud systems [17, 18]. In such systems, each process is somehow a virtual machine in which a separate operating system is running on it. One of the most challenging issues of this kind of systems is the size of the process. The size of the process is an important parameter in determining the feasibility of each process migration trend [18, 20]. The size of the process has a strong correlation with the concept of the migration time.

Regarding the stated issues, it can be concluded that the process migration manager, based on four parameters, time, size, cost, and type of inter-process communication, proceeds to migrate, and these four parameters are fixed and do not change During the process migration trend [19, 21]. The reason for this issue is that the process does not carry out any action that the result of that, applied any changes in the data structures of these four basic parameters, during the process migration trend [19, 21].

To examine the traditional mechanisms in DECS, analyzing more precisely the programs that are faced with different events during the run-time, is needed. The named events may create dynamic and interactive nature in the computing processes of the programs. One of the applications of DECS is the real-world simulations [1, 21, 22]. Traditional systems have also implemented part of the simulation and predictive programs in the past, but today due to changes in the type of simulation and the existence of multiple effective parameters and precise analyzes, it is necessary to run this simulation in Exascale systems [1, 22].

The human brain simulation [23, 24] is one of the new technologies that nowadays is considered and spotlighted by medical and engineering science. The purpose of these programs is to simulate the entire human brain. In line to achieve this goal, the Exascale system should be able to simulate different levels of the brain and be able to switch between levels when it is necessary. A change and mutation from the cell surface of the brain to its molecular level is one of the critical parameters of this kind of simulations, and this objective will not be possible except with the powerful Exascale systems [24].

To achieve proper analysis and accurate understanding of other scientific and engineering programs such as those mentioned above require DECS. The result of this issue has a strong and direct relation to the validity of the results of each simulation that depends on the number of components used in the simulation process [21, 22, 23, 25, 26]. In this kind of programs due to the existence of processes, which have dynamic and interactive nature, the different inter-process communication is seen. This matter is due to the difference in the type of communication or the formation of an unknown new relationship during the process simulation [27]. On the other hand, the existence of more processes to have fine and tune results will increase the code of the program. Program codes' complexity will affect time, inter-process communication and the cost of managing the system [25, 27]. The process migration as an element of the system manager should be able to analyze the conditions to apply to the transmitting mechanisms to the processes that are having dynamic and interactive nature.

3. PROPOSITION

In cluster computing systems, the main purpose of mechanisms is to prepare the possibility of continuing the process execution by changing the location of execution in the shortest time of transmission [7, 11]. The reason for defining the process migration in computing systems is that there is a process in the machine that either the response time exceeds the acceptable time or the local machine cannot execute the process [28]. In such a situation, the load distribution element attempts to specify the <Source, Destination, Process> and by calling the process migration, tries to continue the execution trend by transferring the process. It is important to consider that the process is not defined in abstract mode and has a set of inter-process communications and interactions. In many cases, the failure of the inter-process communication and interaction may fail the process execution trend [19, 28].

Transferring the process in the shortest time is one of the main goals of the process migration [19]. With this approach, it can be stated that the process migration in traditional systems is in the form of a pair of <Time, Running Ability>. In this pair, the Time parameter contains all concepts that are related to the time. Concepts such as transmission time, selection time, and the process selection time intervals are the concepts that can be defined in the space-time domain. The purpose of Running Ability are reasons that the process is selected for the process transmission. When the process is selected for transmission, the aim of the transferring is to continue the process execution trend [29]. Regardless of how the process ultimately runs, the process migration should ultimately lead a situation in which the process continues the execution [6, 29].

In the process migration, due to the inability of communication among the suspended process and other processes, the duration of this operation has direct effects on the performance of the system [10,11]. In distributed computing systems, in which the concept of global activity [29] is used, this issue is more tangibly is debatable. If the computing processes that are being transmitted were not able to communicate with other computing processes of another member of the global activity, the failure of related processes and consequently, the failure of global activity may also occur [28, 29].

In traditional systems, the nature of the mechanisms, which are defined, for the process migration is such that the process migration manager does not expect to occur any uncertain event in the computing activity [12, 13, 19]. As stated in the second section, the uncertainties are meant any changes to each of the four components of <Size, IPC, Cost, Time> during the process migration trend.

In DECS, due to the dynamic and interactive nature of running processes [24,29,30,31], it is possible to occur an event, which it was not defined in the process migration trend. The dynamic and interactive nature of the computing processes in DECS is results of the occurrence of one (or more than one) of the following situations:

- A process in the system begins to create a process, which was not predicted at the time of system design. This is due to the formation of a new concept or parameter in the scientific and engineering program that the computing system is executing it. From the process migration's point of view, in the process migration trend, while the process is in the transition, the process begins to call the fork instruction to create a process for any reason. This issue happens while in the pair component definition of the process migration manager, there is no implication for this situation [24, 29, 30].
- A process begins an inter-process communication or interaction with another process, which was not defined at the time of system design. This is due to the formation of a new link or a new communication in the scientific and engineering program that the computing system is executing it. From the process migration manager's point of view, in the process migration trend, while the process is in the transition, the process begins to initiate an inter-process communication or transaction with another process, which the nature of the creation of this communication is unknown for the process migration manager [24, 29,30].
- A process begins to create a communication or transaction with another process from outside the computing system, which was not predicted at the time of system design. This is due to the formation of defining a new variable in the scientific and engineering program, which is running in the computing system. From the process migration's point of view, during the process migration trend, for any reason or in any way the process can be expanded, and system needs to be scalable. In the definition of the pair, that process migration uses, no concept refers to the mentioned situation [24, 29, 30].

4. ARGUMENT

In DECS, due to the dynamic and interactive nature of the computing processes, flexible management units to adopt appropriate policies against the conditions created during the computing program execution are needed. In this type of systems, the process migration manager should be able to decide, based on structures of dynamic processes, about which mechanism to use for the process transmission. This implies that the process migration manager should be designed flexibly and dynamically itself. The flexibility of the process migration means the capability of detecting the state of the DECS at the time of the transfer operation, to prevent the failure of the migration trend against a random event.

From the process migration manager's point of view, the occurrence of each of three situations mentioned above implies the requirement of a review of the process migration trend. When the process migration, transmits the process, it is possible that the transmitting process would create one of the three situations mentioned above. By happening each case, the process migration manager should be able to analyze the situation newly created and decide on whether the process can continue execution or not. Moreover, if the process migration manager decided on stop the continuation of the process, be able to decide another machine in the system as the destination of the process execution trend. For this purpose, after the analyzing the situation, if the change in status leads to stop the continuation of the process transmission or to the change of destination of the process migration trend, the process migration manager is required to transfer the information of the DECS's status to the load distribution element.

The process migration uses general tow methods for analyzing the situation that occurred dynamic and interactive requirement

for three reasons. The first method, which named as Reduction method, is based on analyzing the migrated process and the event, which had occurred the dynamic and interactive requirement, by the process migration. As the analysis finished, the process migration tries to represent a solution for the occurred situation. Although this method is highly accurate and after the analyzing the situation by the process migration, accurate information is obtained about the cause of the dynamic and interactive requirement's status, but it has two important disadvantages:

- It requires high execution time. So that the time of analyzing the situation may be longer than the time of the process migration trend.
- The process migration should have an independent mechanism for analyzing the situation for each process.

The second method, which named as the Systematic analysis, is based on the developing the definition of the process migration. The definition must be developed in a way that includes general parameters describing the status of process migration trend after the occurrence of dynamic and interactive requests. For this reason, by analyzing the parameters governing process migration trend, after the occurrence of three-situation mention above, should be examined.

When a process calls the systematic fork operation, during the computing execution, the new process creates new dependencies in the DECS. In the traditional definition of the process migration, nothing has been mentioned about the concept of establishing dependencies at the time of the process transmission. This is because in traditional systems when the load distribution element invokes the process migration, it chooses a process that has the least dependencies to the origin machine. It can be used to relocate or transfer its dependencies or to manage any in a way that no failure occurs. Without going into details, it can be explained that the process migration should set a solution for the concept of dependencies before embarking on the process migration trend. In these systems, the concept of creating dependencies, during the process migration trend, is not defined.

In DECS, the execution of the fork operation makes it possible to establish a dependency that has not been solved before the process migration trend has been started. When the transmitting process calls the fork operation, the result of this operation can be a new process creation in the source or destination or any machines in the system. Creating a process means defining a dependency that did not exist before the start of the process migration trend, so the process migration did not anticipate any decision to control this situation. Therefore, in the first step to support the DECS, the consideration of the concept of dependencies, at the time of execution is needed.

When a process during the execution of computing process attempts to define a new IPC with other processes that have not been seen in the previous IPC patterns, defines the dependency that has not been solved before the process migration trend has been started. The definition of a new IPC can take place between two processes or a process and the source owner process. In the first case, when an IPC created between two processes that do not own the resources, this communication and interaction will require constructing new data structures for this communication. The process migration should be managed in such a way that if the process transmission is done in any way, then this connection be accessible to both processes. This implies that the process migration should somehow migrate the process to provide the Viability feature. This concept involves the required concepts to create and use communicational data structures. On the other hand, if this feature is violated, the process migration, through the load distribution element or information about previous process migrations, should decide on the process migration trend.

When a process interacts or communicates with processes outside of the computing system, this leads the manager of the high performance computing system, being expanded to join the computing system outside, to the origin system. This is because the migrated process has requested during the process migration trend, that the local computing system was not able to respond it. Scalability is a concept that is not considered in the definition of the process migration, in traditional computing systems. Now, if this scalability is in a way that results in the definition of a new dependency between the machines that just been a member in the system and one of the membered machines of the system, then a specific dependency called "definition dependency" is created. "Definition dependency" is not considered in the traditional computing systems.

As mentioned above, parameters such as Dependency, Viability, Scalability, Define Dependency and instruction of request, are new parameters that should be considered in the definition of process migration manager. The traditional definition of process migration management is like a pair of <Time, Running-ability>. In this situation, if the consideration of effective parameters in case of the happening of the three mentioned situations, the definition of process migration manager should be in the form of (Process Request Type, <Time, Running-ability, Dependency, Viability, Scalability or Define Dependency>). In this form, Process Request Type defines the process migration manager, and it should contain some operators in it to implement operations related to Time, Running-ability, Viability, Scalability and Define Dependency.

5. CONCLUSION

Computing processes that require DECS create dynamic and interactive applications, in three situations. If the occurrence of each of these three situations happened, before the start of process migration trend, for the process migration the occurred changes should be discussed by the load distribution element. However, if the time of occurrence of each of the three situations during the process migration trend, then the process migration should specify a mechanism for dealing with the situation so that it can continue the process migration trend. From the process migration's point of view, the occurrence of each three situations mentioned in this paper means the formation of a new set of effective parameters in the process migration trend. This issue makes the requirement of developing the definition of the process migration. In traditional systems, a process migration is a tool for the load distribution element, which only decides on the process migration trend based on the two parameters. However, if the generated parameters affecting process migration in the situations listed, it requires a process migration that can analyze parameters during the process migration trend, and also, it is necessary to communicate with the load distribution element to change the process migration trend. On the other hand, the nature of processes that create dynamic and interactive requests is one of the main concepts that should be considered by this element. The integration of these factors leads to redefining the definition of this element based on the new computing environment.

6.REFRENCES

[1] Reed, Daniel A., and Jack Dongarra. "Exascale computing and big data." *Communications of the ACM* 58.7 (2015): 56-68.

[2] Gibbon, Paul, et al. *Towards Exascale Computing-the Jülich Supercomputing Centre Approach*. No. FZJ-2014-05538. Jülich Supercomputing Center, 2014.

[3] Zingale, M., et al. "Modeling Astrophysical Explosions with Sustained Exascale Computing." *arXiv preprint arXiv:1510.05061* (2015).

[4] Germann, Timothy. "Exascale computing and what it means for shock physics." *APS Shock Compression of Condensed Matter Meeting Abstracts*. 2015.

[5] 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.

[6] Sharma, Deepti, and Vijay B. Aggarwal. "An Effective Mechanism for Improving Performance of Load Balancing System in Cluster Computing." *International Journal of Computer Applications* 115.7 (2015).

[7] D'Hollander, E. H., ed. *Transition of HPC towards Exascale computing*. Vol. 24. IOS Press, 2013.

[8] Magoulès, Frédéric, Mark Parsons, and Lorna Smith. "Advances in High Performance Computing: on the path to Exascale software." (2017): 1-2.

[9] Wright, Nicholas J., et al. "Cori: A Pre-Exascale Supercomputer for Big Data and HPC Applications." *Big Data and High Performance Computing* 26 (2015): 82.

[10] 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.

[11] Milojičić, Dejan. *Load distribution: Implementation for the Mach microkernel*. Springer-Verlag, 2013.

[12] Zarrabi, Amirreza. "A generic process migration algorithm." *International Journal of Distributed and Parallel Systems* 3.5 (2012): 29.

[13] Thakkar, Nirali, and Anand Pandya. "Process Migration in Heterogeneous Systems." *International Journal for Scientific Research & Development (2013)*.

[14] Bahena, Vıctor Rodrıguez, and PhD Maria Guadalupe Sanchez Cervantes. "Embedded Distributed Systems: A Case of Study." (2014).

[15] Zhongyuan, Shan, et al. "Use Pre-record Algorithm to Improve Process Migration Efficiency." *Distributed Computing and Applications for Business Engineering and Science (DCABES), 2015 14th International Symposium on*. IEEE, 2015.

[16] Schulte, Michael J., et al. "Achieving Exascale capabilities through heterogeneous computing." *IEEE Micro* 35.4 (2015): 26-36.

[17] Liu, Haikun, et al. "Performance and energy modeling for live migration of virtual machines." *Cluster computing* 16.2 (2013): 249-264.

[18] Jansen, Gerardus T. "Inter-cloud live migration of virtualization systems." U.S. Patent No. 9,104,460. 11 Aug. 2015.

[19] Khaneghah, Ehsan Mousavi, et al. "An efficient live process migration approach for high performance cluster computing systems." *Innovative Computing Technology* (2011): 362-373.

[20] Liu, Haikun, et al. "Performance and energy modeling for live migration of virtual machines." *Cluster computing* 16.2 (2013): 249-264.

[21] Eicker, Norbert, et al. "The DEEP Project An alternative approach to heterogeneous cluster-computing in the many-core era." *Concurrency and Computation: Practice and Experience* 28.8 (2016): 2394-2411.

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

[23] Brömmel, Dirk, et al. "Paving the road towards pre-Exascale supercomputing." *NIC Symposium 2014*. No. FZJ-2014-01327. Jülich Supercomputing Center, 2014.

[24] Alvarez Mallon, Damian, et al. "Programming model and application porting to the Dynamical Exascale Entry Platform (DEEP)." *Exascale Applications and Software Conference*. No. FZJ-2013-05879. Jülich Supercomputing Center, 2013.

[25] Reverdy, V. "Perspectives in numerical astrophysics." *SF2A-2016: Proceedings of the Annual meeting of the French Society of Astronomy and Astrophysics*. 2016.

[26] Innocenti, Maria Elena, et al. "Progress towards physics-based space weather forecasting with Exascale computing." *Advances in Engineering Software* 111 (2017): 3-17.

[27] Lucas, Robert, et al. *DOE advanced scientific computing advisory subcommittee (ASCAC) report: top ten Exascale research challenges*. USDOE Office of Science (SC)(United States), 2014.

[28] Egwutuoha, Ifeanyi P., et al. "A survey of fault tolerance mechanisms and checkpoint/restart implementations for high performance computing systems." *The Journal of Supercomputing*65.3 (2013): 1302-1326.

[29] 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.

[30] Lapenta, Giovanni, et al. "Space weather prediction and exascale computing." *Computing in Science & Engineering* 15.5 (2013): 68-76.

[31] 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): 1-3