International Journal of Innovation and Applied Studies
ISSN 2028-9324 Vol. 4 No. 3 Nov. 2013, pp. 560-567

http://www.issr-journals.org/ijias/

Analysis of Scheduling Algorithms in Grid Computing Environment

Farhad Soleimanian Gharehchopogh[1], Majid Ahadi[2], Isa Maleki[2], Ramin Habibpour[2], and Amin Kamalinia[2]

[1]Department of Computer Engineering,
Hacettepe University, Ankara, Turkey

[2]Department of Computer Engineering,
Science and Research Branch, Islamic Azad University, West Azerbaijan, Urmia, Iran

ABSTRACT: Grid Computing is the technology of dividing computer networks with different and heterogeneous resources based on distribution computing. Grid computing has no limitation due to its geographical domain and the type of undercover resources. Generally, a grid network can be considered as a series of several big branches, different kinds of microprocessors, thousands of PC computers and workstations in all over the world. The goal of grid computing is to apply available computing resources easily for complicated calculations vie sites which are distributed geographically. In another words, the least cost for many users is to support parallelism, minimize the time of task operation and so on in scientific, trade and industrial contexts. To reach the goal, it is necessary to use an efficient scheduling system as a vital part for grid environment. Generally, scheduling plays very important role in grid networks. So, selecting the type of scheduling algorithm has an important role in optimizing the reply and waiting time which involve as two important factors. As providing scheduling algorithms which can minimize tasks runtime and increase operational power has remarkable importance in these categories. In this paper, we discuss about scheduling algorithms which involve independent algorithms such as Minimum Execution Time, Minimum Completion Time, Min-min, Max-min and XSuffrage.

KEYWORDS: Grid Computing, Scheduling, Scheduling Algorithms, Network, Resource.

1 INTRODUCTION

Grid computing technology provides an opportunity for users to access different types of remote resources using communicational substructures of computer networks and distributed systems [1]. Nowadays, heterogeneous computing networks can be connected to each other using grid technology as it seems as a fully integrated machine. Then, very complicated application programs can be implemented which require high processing power and large amount of data [2]. For example, by using grid technology, several super and PC computers can be connected to each other. Grid computing has no limitation due to its geographical domain and the type of undercover resources. In general, a grid network can be considered as a series of several big branches, different kinds of microprocessors, thousands of PC computers and workstations in all over the world. Grid network makes connection between those heterogeneous computers which have no consistency among them. It also recognizes different types of resources and manages to remote reach to them and finally makes possible big and complicated processing to be implemented with huge amount of resources as distributed in a powerful context [3], [4].

In fact, by linking to one of the gridding networks, we can obtain very high computing to implement very big projects. At the other hand, grid network provides different organizations' association about common project fields an all over the world. In distributed environments, a group of users send their programs to a series of resources to implements. Grid computing scheduling system is responsible for program and resources management. The scheduling system must be able to assign appropriate resource to programs and also meets the efficient goals. The scheduling system is capable of providing parallel computational environments link to perform easier due to the same features of programs and resources. Briefly, it can be

Corresponding Author: Farhad Soleimanian Gharehchopogh (bonab.farhad@gmail.com)

said that grid computing is a software frame which collects resource status data, prepares appropriate resources, anticipates the efficiency of each resource and determines the best resource [5], [6] and [7]. In another word, grid resource management system is the responsible for controlling grid resources and its goal is to promote efficiency [8]. Grid data service is a part of grid resources management system which provides dynamic data of grid resources. Grid scheduling is another part of grid resources management system. It provides an assignment of tasks to resources using resource data which created by grid data service and their tasks data. It is called scheduling assignment. An efficient scheduling causes to decrease ending time and tasks implementation costs, more reliability and better error controlling. To reach these efficiencies, scheduling algorithms are created which are discussed in this paper.

We organized general structure of this paper as follows: in section 2, we discuss about the challenges of grid scheduling system; in section 3, grid computing is discussed; in section 4, it is presented various types of independent works of scheduling algorithms; in section 5, we will discuss about grid scheduling algorithms and finally in the section 6, conclusion and future works is presented.

2 GRID SYSTEM CHALLENGES

Although grid network is placed in distributed parallel computational systems, but its unique features which is scheduling makes the resources links difficult in grid environment. Grid scheduling system must overcome challenges which we note to be able to provide services with appropriate efficiency and perform grid potentials [9]. The main challenges of grid scheduling system are as follow:

2.1 HETEROGENIC OF RESOURCES

Grid computing consists of two resources: computational resources and communication (network) resources. There is heterogenic in these two resources. The networks are different in band width and communication protocols. The computational resources include different hardware (e.g. series of orders, architecture, number of processes, physical memory size, processing rate) and different software (e. g. operating system, file system, cluster management software). So, scheduling system must use these different computational powers as optimal [9], [10].

2.2 INDEPENDENT DOMAINS

It is possible that grid consists of several various management domain in which each one has its own particular security and management policies and usually allow the group of users to use resources. It means that it must be impossible to implement invalid users programs on that domain. Each site has an independent nature and its own particular scheduling policy. This makes the anticipation of task implementation impossible. At the other hand, the goal determination of unit general efficiency is also impossible because each site makes decision about scheduling due to its goals and independent from others [9, 11].

2.3 UNALLOCATED RESOURCES

Due to the available unallocated resources, there is a remarkable competition about using the resources. It means that a resource is connected to several grids and local users and other grids use that, simultaneously. One of the competition results is that resources don't allocate to all works. For example, in widespread networks which use internet protocol series, network features such as delay and band width are changed due to the simultaneous usage of users. In such an environment, designing a precise efficient model is a difficult task. By assessing the fraction of available resources dynamically, competition can be recognized. In grid network, resource management system facilitates resource efficiency anticipation by guaranteeing service quality and resource reservation [9], [12]. The simplest way is that the tasks are executed on several machines. The machine which is running normally may involve in computation abnormally due to facing with a lot of activities. So, the tasks must be executed on no-use machines.

2.4 VARIETY OF PROGRAMS

The increasing numbers of grid users have programs with their own particular needs. It is possible that the programs need sequential or concurrent implementation. The programs include dependent and/or independent tasks. It is also difficult to build an all-purpose scheduling system to be able to manage different programs [9], [13].

2.5 DYNAMIC BEHAVIOUR

In traditional parallel computational environments such as clusters, there are a mass of constant resources. But, there is also dynamic either in network or computational resources in grid network. Firstly, a network which is connected by many users couldn't be able to provide guaranteed band width. Secondly, resource accessibility and their capabilities are also dynamic. It means that new resources are added to the network or may be removed from accessibility due to network problems. Due to the available competition among machines to access the resource, the efficiency of them is also changed over time. Scheduling must be able to adapt such a dynamic behavior, recognize it automatically after connection to a new resource, consider it in the future decision making and finally guarantee its reliability (grid system) as a resource becomes out of reach due to ambiguous reasons by applying mechanisms such as checkout point and/or re-scheduling [9], [14].

Fig 1. shows a sample of grid scheduling for users.

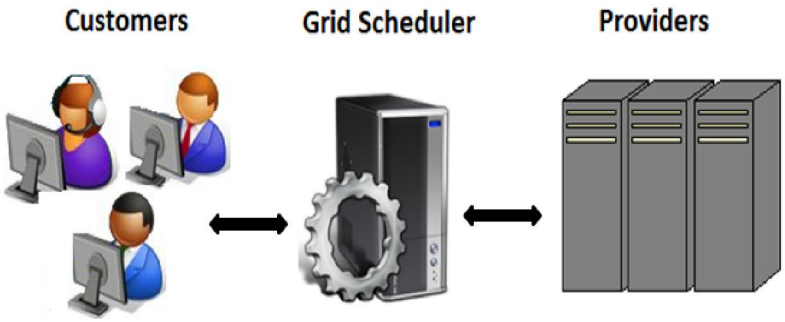

Fig. 1. A Sample of Grid Scheduling for Users

3 GRID SCHEDULING

In grid scheduling, there are several hierarchical scheduling which include algorithms. In this paper, we begin to discuss about hierarchical scheduling to get familiar with them. Independent tasks scheduling is considered as one of the hierarchical scheduling in which its algorithms are discussed in this paper.

3.1 LOCAL AND GLOBAL SCHEDULING

Local scheduling determines assignment and implementation methods in a processor. However, global scheduling policy uses system data to divide the tasks among several processors in such a way to optimize system efficiency [15]. In grid scheduling, it is used global scheduling to assign resources to the tasks.

3.2 STATIC AND DYNAMIC SCHEDULING

In static scheduling, it is assumed that data related to both all grid resources and tasks of an applied program while scheduling is available [16]. In dynamic scheduling, the main idea is that tasks assign is implemented while the applied program is running. This method is useful as it is impossible to determine running time, branching orientation and unknown number of loop repetition and real time of tasks. Grid computing can be performed in two types of static and dynamic [17].

In static scheduling, each task for implementation is assigned to a resource once. So, the location of applied program is constant and accurate estimation can be provided from computing costs during real implementation development. The main advantage of this method is the simplicity of programming from scheduling view. This model provides a global view about tasks and costs. However, cost estimation based on static data isn't in accordance with new positions (situations). For example, if a node which is assigned to perform a computing destroys or inaccessible due to network defaults and/or the response time becomes more than expected due to network traffic. Of course, to decrease such problems, it can be used special mechanisms such as re-scheduling to provide opportunities for tasks transmission which naturally impose additional extra load to system. Using these mechanisms makes the distance between static and dynamic scheduling less important. Dynamic scheduling is usually used when it is difficult to anticipate cost of applied program and/or tasks are dynamic and on the line [16]. Dynamic scheduling includes two parts: situation estimation (rather cost estimation in static scheduling) and decision making [17]. The system situation scheduling includes gathering the total data situation of grid network and building estimation. Based on noted estimations, it is decided tasks assign to which resource. As the cost of assignment isn't

estimated, the natural way to keep the system dynamic and secure is to balance the total load of system resource. The advantage of dynamic balancing to static scheduling is that system doesn't need to know about implementation time of applied programs before it [18]. This issue is particularly useful in the system that its goal is to maximize resource efficiency. If a lot of tasks are added to a resource, the balancing policy will determine whether it is necessary to send some tasks to the other resources and if it is needed which tasks must be sent.

3.3 Optimal Scheduling

If all data related to resources and tasks are known, optimal assigning will be performed based on criteria such as "minimizing spreading time" or "maximizing resources efficiency". However, due to NP-complete nature of scheduling algorithms, having logical assumptions and optimality proving of such algorithms would be difficult. For this reason, the solutions close to the optimal are investigated in current researches. Algorithms close to the optimal solutions are divided in to two groups: approximation and innovation [19]. In approximate algorithms, it is applied formal computing models but doesn't investigate the total reply space. However, when a good enough solution found, the search would be stopped. This method decreases the spent time to find scheduling. The other group of algorithm close to the optimal solution is innovative. These algorithms include assumptions based on facts about processing and system load characteristics. The solution it produces wouldn't be lead to the optimal reply but its resources and cost is logical. Assessing these kinds of solutions is performed in real and/or simulated environments. Innovative algorithms are considerably adjusted to grid environment [19], [20].

3.4 Distributed Scheduling and Central Scheduling

In dynamic scheduling, decision making is assigned to central scheduling about global scheduling or shared among several distributed scheduling [21]. Central scheduling implementation is a simple process but not expandable, can't bear error and it is possible to be just a data entry. In distributed scheduling, the resource assigning decision making process is performed in parallel with the real process of tasks to save processes expensive cycles. In distributed scheduling, the tasks are sent to task machines which involved resources in parallel. It acts well when all tasks have request of resource in a same site.

3.5 Cooperative Scheduling and Independent Scheduling

The other considerable point of distributed scheduling algorithms is that whether scheduling involved nodes do their tasks cooperatively or independently? In independent status, each scheduling decides as independent and based on its optimal goals without considering its decision effects on other systems. For example, this kind of scheduling is the scheduling of applied program level. In cooperative status, each scheduling is supposed to program and schedule the related tasks. However, all scheduling are going to achieve a systematic common goal and the common goal is to promote the efficiency of total system rather local and/or particular program efficiency [22].

3.6 Compile Time Scheduling and Runtime Scheduling

Scheduling which compute the time of applied series of tasks in compile time is so called compile time scheduling. It is supposed that data such as processing rate are known. It is favorite model because it is easy to program scheduling. In runtime scheduling, total time to perform tasks is determined by scheduling during runtime [23, 24].

4 Independent Tasks Scheduling Algorithms

Technology growth and various integrated tasks in a form of comprehensive and integrated system causes that time management process face with a lot of challenges in grid computing. As grid computing is a series of distributed heterogeneous computers and connects to each other through network and shares programs, data and heterogeneous computing resources. So, wide computing systems such as grid must involve algorithms which include high speed rate and efficiency. As a series of independent tasks enters a system, it will be used a common strategy to assign them to the resources based on resources load. The goal is to reach higher operational power [22]. The most important algorithms of independent tasks scheduling is as follow:

4.1 MINIMUM EXECUTION TIME ALGORITHM

Minimum execution time algorithm assigns the task to a resource which has the best anticipated executed time for it. It doesn't consider whether this resource is available or not. Its main goal is to assign each task to the best machine. It may cause extreme imbalance of load. So, it isn't appropriate for heterogeneous computing environments because it doesn't consider tasks and resource features [24], [25]. In this algorithm, it is assigned tasks to machine without considering workload of processing machine and causes it can't end up the task in the determined time [26]. Due to the time changeable nature of this algorithm, the resources aren't under the control of central management and can't be sure about tasks execution in the determined time [27].

4.2 MINIMUM COMPLETION TIME (MCT) ALGORITHM

MCT algorithm assigns tasks to the resource which has the smallest time of completion. It causes that some tasks are assigned to machines which don't contain the smallest running time. S, in this algorithm, the task is mapped to a resource which can end up the execution sooner than the other resources. Of course, if all possible resources have similar workload, the selected resource will certainly involve more capable resources and can end up execution sooner. The idea of this algorithm is to combine time and load balance advantages in MET and refuse imbalance circumstances in which MET acts weaker [17]. It tries to assign the task to the best machine as soon as possible and replies to requests as optimal in the least possible time [28]. In MCT algorithm, each task is assigned to the resource considering its time to be able to end up execution sooner [29].

4.3 MIN-MIN ALGORITHM

Min-min algorithm begins to execute using U series (the series of all un-mapped tasks). Then, it is created the minimum series of completion times for tasks according to Equation (1).

$$M = [\min_{0<j<\mu}(ct(t_i, m_i)), for \quad each \quad t_i \in U] \tag{1}$$

So, the task which has the shortest time of completion is selected from M series and then assigned to that resource. Then, another task is selected from U series and the noted stages are repeated. It continues as all members of the U series become mapped. Similar to MCT method, it acts according to completion time minimizing but in Min-min method, all un-mapped tasks are considered during decision making for mapping. While in MCT, only a task is considered each time [17], [24],[25]. This algorithm is use to manage single or several tasks. A central machine acts as the resource manager to schedule tasks in it and all processing machines in grid environment are under the control and management of this machine. The tasks are first assigned to central scheduling and then central scheduling transfer them to the other appropriate processing machines. The tasks which don't assign them resource will put in an array of central task to be placed in the processing later [27], [28].

4.4 MAX-MIN ALGORITHM

It is similar to Min-min algorithm except that after building M series, it is selected the task which has the longest completion time. Intuitively, Max-min algorithm tries to minimize the related problems of task execution with the longest execution time [18, 30]. For example, suppose that a program has a lot of tasks with short execution time as well as a task with long runtime. If long task is mapped to the best machine, it provides the opportunity to execute long task along with the short one concurrently. By using this method, Max-min algorithm will have better map, load balance and expanding time. In this way, Max-min map has better function than Min-min map. But, if it is executed all short tasks and then long ones, a lot of machines will remain useless in the site. So, in Max-min, all tasks are firstly arranged chronologically as descending (from highest to lowest. Then, the first task is selected among them and the required time assigned to execute it. The task is assigned to a resource which can end up the execution as soon as possible. This process is applicable for all tasks to finish all requests. Min and Max-min algorithms have good efficiency and can easily incorporate with other scheduling algorithms [30]. They also guarantee tasks service quality partly and also minimize the access time to resources [28].

4.5 XSUFFRAGE ALGORITHM

Suffrage is another algorithm to schedule independent tasks [7]. The logic of this algorithm is that each task is assigned to particular machine and can't be take resource from other machines. The efficiency and function of tasks scheduling in Suffrage algorithm is performed based on MCT. In Suffrage algorithm the tasks faces problems as there are input and output

data as well as clustering resources. In this way, logically, the task must be assigned to the free resources to prevent time waste. If the resources are clustered and have similar efficiency, Suffrage algorithm won't have good performance and the tasks may not be assigned to free resources with good efficiency. Consequently, the desired task is removed due to the lack of resources from requests array. To solve this problem, a group of researchers have been improved Suffrage algorithm and called it Xsuffrage. In this way, each task receives resources in cluster level chronologically.

5 DISCUSSION

Nowadays, an efficient and effective scheduling is required to increase efficiency in grid computing. In grid networks, the resources are heterogeneous and this causes the increasing complexity of applicable scheduling in these environments. Grid computing shares the distributed resources geographically to achieve common goals with each other and this performs using scheduling algorithm. So, one of the main parts of grid computing is scheduling. Scheduling is responsible to manage resources and divide the tasks among computing resources accurately. Error controlling is also considered as one of these responsibilities. Optimal scheduling leads to increasing service quality.

Scheduling algorithms in grid computing is divided to two groups of global and local. In local scheduling method, it is decided about processes assigning to a process and its execution. In global scheduling method, it is performed processes assigning to several processes to optimize the general efficiency of system using systems data. The next level in scheduling hierarchy is to select static and dynamic scheduling. In static scheduling, the required data is available about all grid resources during scheduling. So, accurate estimation can be performed from computing resources in real execution. In this method, assigning requests are confirmed and resource request estimation is simple. But resource estimation based on static data isn't compatible esp. in cases that one of the selected nodes are destroyed to do computing and/or amount of load is increased due to available requests which their reply time is longer than expected. But in dynamic algorithms, the main idea is to assign request in programs runtime. The increasing dynamism in grid computing causes that these algorithms have better function. So, dynamic scheduling acts well in most cases but the main problem occur as the scheduling task is delivered to the desired resource and placed in the expecting list. If it is executed small tasks by scheduling algorithms and the bigger tasks are distributed among the resources at the end, wasted time will be increased in the resources and consequently runtime difference will increase in the most applicable resource than the other resources. Because, it isn't easy to assign transporting conditions of a big task from most applicable resource to faster one due to time limit and this isn't appropriate for grid environment considering time. If bigger tasks are distributed at first and then the smaller ones, it can be provided an opportunity to minimize the difference between the most applicable resource and the other ones using the remaining time to distribute the small tasks among the resources. It is possible that the amount of tasks reach a point which causes increasing rate of resource requests in grid computing. If applied programs are applicable in grid network, it can be transported to useless machines to execute in these cases. In general, a grid can perform the computing load balance in widespread series of resources.

Grid computing provides a software and hardware infrastructure in which local resources of each machine for users can be used in a network environment of distributed and heterogeneous machines. In fact, grid network is associated with a series of resources in widespread scale which commonly used. There are a lot of obstacles to decrease efficiency and scalability of grid computing. The first obstacle is algorithms which are used to classify tasks among a lot of processes. If algorithm is divided to only a few numbers of time independent parts, it can be an obstacle against scalability. The second obstacle is that the executing parts don't be fully independent from each other which causes tasks interruption and limits scalability. For example, all tasks are commonly performed writing and reading a file or a database if necessary. In this case, access limitations to that file or database will be a limiting factor for tasks scalability. We, in this paper, propose to discuss about scheduling algorithms which involve independent algorithms such as MET, MCT, Min-min, Max-min and XSuffrage. Each algorithm tries to minimize runtime and resource heterogeneity and reaches to high operational power.

6 CONCLUSION AND FUTURE WORKS

Due to the developing rate of trade, industry and science world, scheduling is considered as one of the main discussions in grid environment. As providing scheduling algorithms which can minimize tasks runtime and increase operational power has remarkable importance in these categories. Along with, there are algorithms which meet the needs as far as possible which is noted to a few of them in this paper. Grid scheduling system and various types of challenging features in grid are discussed in this paper to get familiar with scheduling challenges and be selected an algorithm to remove these challenges. Then, grid computing hierarchy is discussed to get familiar with them. Independent tasks scheduling algorithms are also

explained. We hope to provide researches in future to solve some un-solved scheduling challenges using scheduling algorithms incorporating.

REFERENCES

[1] I. Gandotra, P. Abrol, P. Gupta, R. Uppa, and S. Singh, "Cloud Computing Over Cluster, Grid Computing: a Comparative Analysis", *Journal of Grid and Distributed Computing*, Vol. 1, no. 1, pp.1-4, 2011.

[2] I. Foster, Y. Zhao, I. Raicu, and S. Lu, "Cloud Computing and Grid Computing 360-Degree Compared", *IEEE Grid Computing Environments Workshop (GCE 2008),* Austin, TX, pp. 1-10, 2008.

[3] F. Berman, R. Wolski, H. Casanova, W. Cirne, H. Dail, M. Faerman, S. Figueira, J. Hayes, G. Obertelli, J. Schopf, G. Shao, S. Smallen, S. Spring, A. Su, and D. Zagorodnov, "Adaptive computing on the grid using apples", *IEEE Trans. On Parallel and Distributed Systems (TPDS),* Vol. 14, No. 4, pp. 369-382, 2003.

[4] J. Broberg, S. Venugopal, and R. Buyya, "Market-oriented Grid and utility computing: The state-of-the-art and future directions", *Journal of Grid Computing*, Vol. 3, No. 6, pp. 255-276, 2008.

[5] C. Banino, O. Beaumont, L. Carter, J. Ferrante, A. Legrand, and Y. Robert, "Scheduling strategies for master-slave tasking on heterogeneous processor platforms", *IEEE Trans. Parallel Distributed Systems*, Vol. 15, pp. 319-330, 2004.

[6] O. Beaumont, A. Legrand, and Y. Robert, "Scheduling divisible workloads on heterogeneous platforms", *Parallel Computing*, Vol. 29, pp. 1121-1152, 2003.

[7] H. Casanova, A. Legrand, D. Zagorodnov, and F. Berman, "Heuristics for Scheduling Parameter Sweep Applications in Grid Environments", *In Ninth Heterogeneous Computing Workshop, IEEE Computer Society Press*, pp. 349-363, 2000.

[8] E. Heymann, M. A. Senar, E. Luque, and M. Livny, "Adaptive scheduling for master-worker applications on the computational grid", Grid Computing - GRID 2000, Springer-Verlag LNCS, pp. 214-227, 2000.

[9] A. A. Khokhar, V. K. Prasanna, M. E. Shaaban, and C. L. Wang, "Heterogeneous computing: Challenges and opportunities", *IEEE Computer*, Vol. 26, No. 6, pp. 18-27, June 1993.

[10] J. P. Goux, S. Kulkarni, J. Linderoth, and M. Yoder, "An enabling framework for master worker applications on the computational grid", *In Ninth IEEE International Symposium on High Performance Distributed Computing (HPDC'00*), IEEE Computer Society Press, 2000.

[11] Z. Yang, W. Zhao, W. Zhang, and S. Huang, "Research on trust model in autonomous domain of campus grid", *International Conference on Computer Science and Service System (CSSS)*, Nanjing, pp. 1670-1672, 27-29 June 2011.

[12] I. Ahmad, S. Faheem, and G. Qasim, "MMOD: A General Resource Scheduling Algorithm for Computational Grid", *International Conference on Emerging Technologies*, Islamabad, pp. 141-144, 2007.

[13] E. Huedo, S.R. Montero, and I.M. Liorente, "Experiences on adaptive grid scheduling of parameter sweep applications", *12th Euromicro Conference on Parallel, Distributed and Network-Based Processing*, pp. 28-33, 11-13 Feb 2004.

[14] J. Kolodziej and F. Xhafa, "Modelling of User Requirements and Behaviors in Computational Grids", *International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC),* Fukuoka, pp. 548-553, 4-6 Nov 2010.

[15] T.l. Casavant and J.G. Kuhl, "A taxonomy of scheduling in general-purpose distributed computing systems", *IEEE Transactions on Software Engineering,* Vol. 14, No. 2, pp. 141-154, Feb 1988.

[16] T.D. Braun, H.J. Siegel, N. Beck, L.L. Boloni, M. Maheswaran, A.I. Reuther, J.P. Robertson, M. D. Theys, B. Yao, D. Hensgen, and R.F. Freund, "A Comparison Study of Eleven Static Heuristics for Mapping a Class of Independent Tasks onto Heterogeneous Distributed Computing Systems", *Technical Report TR-ECE-00-4, School of Electrical and Computer Engineering,* Purdue University, Mar. 2000.

[17] M. Maheswaran, S. Ali, H. J. Siegel, D. Hensgen, and R. F. Freund, "Dynamic mapping of a class of independent tasks onto heterogeneous computing systems", *J. Parallel Distrib. Comput*, Vol. 59, pp. 107-121, 1999.

[18] T.D. Braun, H.J. Siegel, N. Beck, L.L. Boloni, M. Maheswaran, A.I. Reuther, J. P. Robertson, Mitchell D. Theys, and Bin Yao, "A Comparison of Eleven Static Heuristics for Mapping a Class of Independent Tasks onto Heterogeneous Distributed Computing Systems", *Journal of Parallel and Distributed Computing*, pp. 810-837, 2001.

[19] T. Andronikos and N. Koziris, "Optimal scheduling for UET-UCT grids into fixed number of processors", *8th Euromicro Workshop on Parallel and Distributed Processing,* Rhodos, pp. 237-243, 2000.

[20] T. Andronikos, N. Koziris, G. Papakonstantinou, and P. Tsanakas, "Optimal Scheduling for UET-UCT Generalized n-Dimensional Grid Task Graphs", *Proceedings of the 11th IEEE International Parallel Processing Symposium (IPPS97)*, pp.146 -151, 1997.

[21] K. Christodoulopoulos, V. Sourlas, I. Mpakolas, and E. Varvarigos, "A comparison of centralized and distributed meta-scheduling architectures for computation and communication tasks in Grid networks", *Journal Computer Communications*, Vol. 29, pp. 1172-1184, 2009.

[22] O. Beaumont, A. Legrand, L. Marchal, and Y. Robert, "Independent and divisible tasks scheduling on heterogeneous star-shaped platforms with limited memory", *Proceedings of the Conference on Parallel, Distributed and Network-Based Processing (Euromicro-PDP'05),* pp. 179-186, 2005.

[23] G.C. Sih and E. A. Lee, "A compile-time scheduling heuristic for interconnection-constrained heterogeneous processor architectures", *IEEE Trans. Parallel Distrib*, pp. 175-186, 1993.

[24] R. Armstrong, D. Hensgen, and T. Kidd, "The relative performance of various mapping algorithms is independent of sizable variances in run-time predictions", in *7th IEEE Heterogeneous Computing Workshop (HCW '98),* pp. 79-87, 1998.

[25] R.F. Freund, M. Gherrity, S. Ambrosius, M. Campbell, M. Halderman, D. Hensgen, E. Keith, T. Kidd, M. Kussow, J. D. Lima, F. Mirabile, L. Moore, B. Rust, and H. J. Siegel, "Scheduling resources in multi-user, heterogeneous, computing environments with SmartNet", in *7th IEEE Heterogeneous Computing Workshop (HCW '98),* pp. 184-199, 1998.

[26] L.Y. Tseng, Y.H. Chin, and S.C. Wang, "The Anatomy Study of High Performance Task Scheduling Algorithm for Grid Computing System", *Computer Standards & Interfaces, Elsevier B.V,* Vol. 31, pp. 713-722, 2009.

[27] O.M. Elzeki, M.Z. Rashad, and M.A. Elsoud, "Overview of Scheduling Tasks in Distributed Computing Systems", *International Journal of Soft Computing and Engineering (IJSCE),* Vol. 2, No. 3, pp. 470-475, July 2012.

[28] F. Alharbi, "Simple Scheduling Algorithm with Load Balancing for Grid Computing*", Asian Transactions on Computers (ATC),* Vol. 2, No. 2, pp. 8-15, 2012.

[29] R. Kaur, T. Kaur, and H. Kaur, "Scheduling in Grid Computing Environment", *International Journal of Advanced Research in Computer Science and Software Engineering*, Vol. 3, No. 6, pp. 455-458, June 2013.

[30] O.H. Ibarra and C.E. Kim, "Heuristic algorithms for scheduling independent tasks on no identical processors", *J. Assoc. Comput.,* Vo. 24, No. 2, pp. 280-289, April 1977.