Journal of Engineering and Applied Sciences 12 (8): 1972-1978, 2017
ISSN: 1816-949X

Development of First Come First Serve-Ejecting Based Dynamic Scheduling (FCFS-EDS) Simulation Scheduling Method for MPI Job in a Grid System

Rusydi Umar and Ardi Pujiyanta
Department of Informatics Engineering, Faculty of Industrial Engineering,
University of Ahmad Dahlan, Prof Soepomo Street, Janturan, Yogyakarta, Indonesia

Astract: Scheduling MPI job in a grid system, based on reservation, First Come First Serve-Ejecting based Dynamic Scheduling (FCFS-EDS) comes into flexible advance reservation it is dynamic in a logical view. It will be difficult to learn an imagine how dynamic it is because it is in logical view. The objective of this research is to make learning FCFS-EDS scheduling for MPI job method easier, by developing a simulation software to simulate First Come First Serve-Ejecting based Dynamic Scheduling (FCFS-EDS) scheduling method for MPI job in a grid system. With this software the process of scheduling MPI job with FCFS-EDS method can be observed by user in logical view. The user can also spot where the job will be executed (physical view). The development of this software follows waterfall process model. From the testing this software is worth to be implemented.

Key words: Scheduling, MPI job, FCFS-EDS, grid system, imagine

INTRODUCTION

Parallel applications need a very big resources, these resources come from many parallel computer, execute the parallel application in the same time. Message Passing Interface (MPI) is API specification which allow the parallel application to communicate to each other by passing the message. Typically, MPI is used by parallel program executed in a cluster, super computer or in a grid system. This parallel application can be called as MPI application or MPI job. If MPI job needs n nodes of computation power to execute, then these n jobs must be executed in the same time. Another applications needs resources from many places is workflow application where jobs in workflow application must wait another job before it can be executed. In the workflow application it is called dependency.

In multimedia and real time application to guarantee the broadcasting of video and audio in a network flawlessly such as video conference, it needs a particular amount of bandwidth. With these three example we need a reservation for scheduling to guarantee that particular resources will be ready to use for an application as needed in a future time (Smith *et al*., 2000).

In most grid system with traditional scheduler, all accepted job is put in a queue if the resource needed is not available. Every grid system uses different scheduling method like First Come First Serve (FCFS) Shortest Job First (SJF) Earliest Deadline First (EDF) or Easy backfilling (Mu'alem and Feitelson, 2001). All of the scheduling method above execute the job based on three parameters, i.e., number of resources needed, starting time and duration time of the job. With all scheduling method above there is no guarantee about when these job will be executed (Sulistio and Buyya, 2004). Advance reservation is the process of requesting resources for use at specific times in the future (Smith *et al*., 2000). Common resources that can be reserved or requested are storage elements, network bandwidth, compute nodes or a combination of any of those.

First Come First Serve-Ejecting based Dynamic Scheduling (FCFS-EDS) comes into flexible advance reservation it is dynamic in a logical view (Umar *et al*., 2012). It will be difficult to learn an imagine, how dynamic it is because it is in logical view. The aim of this research is to make learning FCFS-EDS scheduling for MPI job method easier by developing a simulation software to simulate First Come First Serve-Ejecting based Dynamic Scheduling (FCFS-EDS) scheduling method for MPI job in a grid system. With this software the process of scheduling MPI job with FCFS-EDS method can be observed by user in logical view. The user can also spot where the job will be executed (physical view).

Corresponding Author: Rusydi Umar, Department of Informatics Engineering, Faculty of Industrial Engineering, University of Ahmad Dahlan, Prof Soepomo Street, Janturan, Yogyakarta, Indonesia

Literature review: The grid computing discipline includes connections of a possibly unlimited number of omnipresent computing devices within a network or grid and also includes the networking services. This new advanced methodology to computing has similarities to a enormously large power grid such as what delivers power to our homes and business daily. This innovative methodology led to increasing interest in pairing geographically distributed resources for answering compute intensive application and large-scale data intensive application. This methodology is leading to what is popularly called Peer-to-Peer (P2P) computing networks and the Grid (Buyya and Sulistio, 2008). The distribution of utility-based power, worldwide has become second nature to many of us. We know that by merely entering into a chamber and switching on the AC or any other electric devices, the power will be engaged to the proper devices of our choice at that time. In this same utility style, grid computing openly search for an unlimited number of computing devices into grid environment. Grid computing can add an unlimited number of computing devices into any grid environment.

Grid (Daniel, 2005; Foster and Kesselman, 1999) computing platforms allow the sharing, selection and combination of geographically distributed heterogeneous resources (data sources and computers) belonging to different managerial organizations (also called Virtual Organizations (VOs)) for answering large-scale problems in engineering, commerce and science.

MATERIALS AND METHODS

Resource management in a grid system: The key components of a grid infrastructure are resource management, security, information services and data management (Jacob *et al*., 2003). The word resource management in grid computing can be defined as processes that control the technique that grid capabilities are made accessible for other entities like applications, services and users (Foster and Kesselman, 2003). The objective of resource management is to guarantee efficient deployment of computer resources and for optimization performance of particular tasks (Isah and Safwana, 2012).

In worldwide grid environment, users communicate with a Grid Resource Broker (GRB) that hides the complexity of resource management and scheduling. For the operation of a computational grid, the grid resource broker sees properties of resources that the user can contact using Grid Information Services (GIS) negotiates with (grid-enabled) resources or their mediators using middleware services, maps tasks to resources (scheduling) stages the data and application for processing (deployment) and finally collects results (Buyya *et al*., 2000).

There are three types of resource management architecture style which are centralized, decentralized and hierarchical. Hierarchical and decentralized approaches are appropriate for grid resource and operational management, since it is impossible to express an adequate system-wide performance matrix and common fabric management policy (Buyya *et al*., 2000).

To handle the wide differences in the software applications and hardware used in grid environments above different forms of grid networks, a software known as middleware is used. The most significant components of the middleware is the resource manager which handles resource selection and scheduling (Foster *et al*., 2000; Sulistio and Buyya, 2004).

There are two varieties of resource management which are global resource management and local resource management. Local resource management deals with scheduling and managing resource at a specific site or resource provider. Global resource manager doesn't possess the resources at a site and therefore doesn't have control over them. The global resource manager has to make best-effort choices and then submit the job to the designated resources (Schopf, 2004).

Grid resource management has some different layers of schedulers. At the highest layer is global resource management that may have a more universal view of the resources but very far away from the resources where lastly the application will be performed. At the lowest layer is a local resource management that manages a particular resource or set of resources. Other layers may exist in between, for instance to handle several resources specific to a project (Sulistio and Buyya, 2004). In local resource management, resources are accessed, assigned and allocated satisfying to Quality of Service (QoS) principles such as advance reservation, deadline and price.

MPI job: The main purpose for making and using parallel computers is that parallelism is one of the best ways to overcome the speed blockage of a single processor (Hwang and Xu, 1998). There are three approaches for making parallel applications. The first approach is based on automatic parallelization. With this approach a programmer does not need to worry about parallelizing task. The second is based on the usage of parallel libraries. With this approach parallel code that is in common to some applications is captured into a parallel library. The third approach is major recoding or resembles the code from scratch in creating a parallel

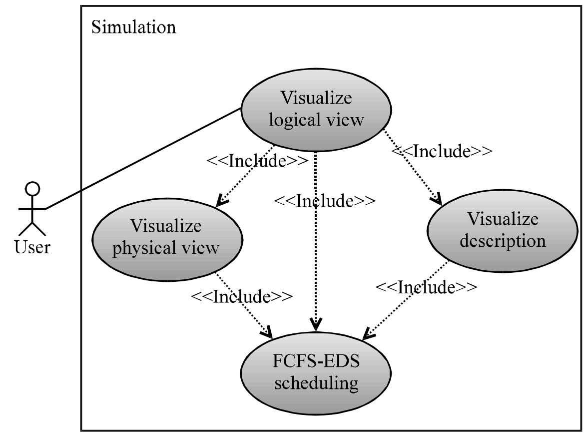

Fig. 1: Use case diagram

application. The programmer is free to select the language and the programming model used for creating a parallel application (Tiemeyer and Wong, 1998).

Message Passing Interface (MPI) is API specification which allow the parallel application to communicate to each other by passing the message. Typically, MPI is used by parallel program executed in a cluster, super computer or in a grid system. This parallel application can be called as MPI application or MPI job. If MPI job needs n nodes of computation power to execute, then these n jobs must be executed in the same time (Kan, 2001).

FCFS-EDS scheduling method: First Come First Serve-Ejecting based Dynamic Scheduling (FCFS-EDS) comes into flexible advance reservation it is dynamic in a logical view (Yang *et al*., 2007). This strategy takes an advantage of shifting earlier reservations made (subject to given flexible constraints) to make room for new incoming reservation request. However, the planning for reservation in their model provides a logical view as against the physical view reported in the literature. Therefore, they call their method as First Come First Serve-Ejecting based Dynamic Scheduling (FCFS-EDS) strategy.

Analysis and design: The requirement of this software simulation of scheduling MPI job in a grid system, based on reservation using First Come First Serve-Ejecting based Dynamic Scheduling (FCFS-EDS) method are: this software can visualize logical view as well as physical view of the scheduling. This software can also give the description of the process happening in the logical view and run the FCFS-EDS scheduling method.

Use case diagrams display business use cases, actors and the associations between them. The associations between actors and business use cases say that an actor can use a certain functionality of the software. Use case of this simulation software can be seen in Fig. 1. The package diagram shows how the various classes are grouped into packages. Package diagram of this simulation software can be seen in Fig. 2.

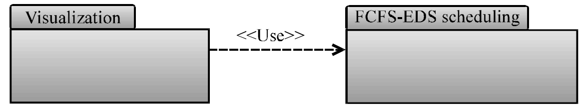

Fig. 2: Package diagram

RESULTS AND DISCUSSION

We have developed a simulation software to simulate First Come First Serve-Ejecting based Dynamic Scheduling (FCFS-EDS) scheduling method for MPI job in a grid system. This software then called as simulation of FCFS-EDS scheduling for MPI job. The initial display of the software can be seen in Fig. 3. From Fig. 3, we can see that there are two areas which are: area for visualization or simulation and area for menu.

Menu in simulation of FCFS-EDS scheduling for MPI job: Menu in simulation of FCFS-EDS scheduling for MPI job consist of two main menus, i.e., file menu and help menu. File menu is used for opening file (Open), i.e., input file for this software, the sequence of incoming MPI job to the local resource manager in each timeslot. The incoming sequence follows poisson distribution. Close menu is used for closing the input file that has been or not been simulated or visualize. This is done to change the input file of the incoming MPI job. File menu can be seen in Fig. 4.

If we click open menu in Fig. 4. We will be directed to the window for choosing file we want to open. This file contains the sequence of incoming MPI job to the local resource manager. The extension of this file is .txt. The window for opening input file

If we click the exit menu in Fig. 2. We will be directed to the dialog box asking whether we really want to close the program simulation of FCFS-EDS scheduling for MPI job. Dialog box exit can be seen in Fig. 6.

The next menu is close menu, this menu is used for closing input file. If we close the file, then we have to open another input file in order to run the simulation. Hel menu is containing about simulation of FCFS-EDS scheduling Fig. 7 depicts help menu. In Fig. 7, if we click help menu about simulation of FCFS-EDS scheduling we will be directed to the window containing the description of the the software simulation of FCFS-EDS scheduling for MPI job. The about window can be seen in Fig. 8.

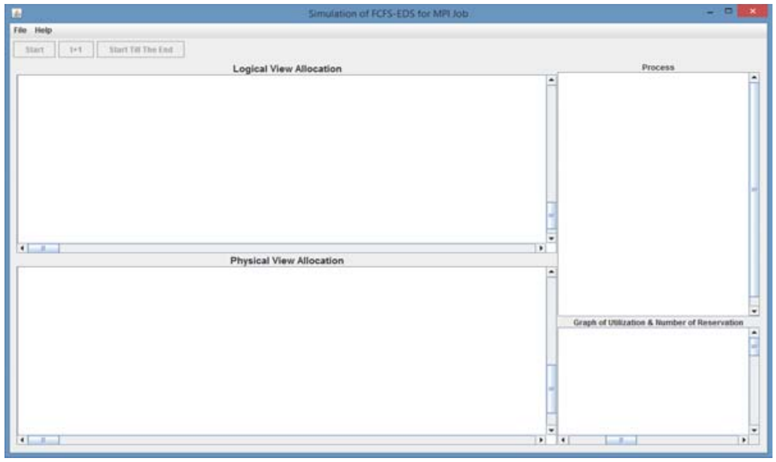

Fig. 3: Initial display of simulation of FCFS-EDS scheduling for MPI job

Fig. 4: File menu

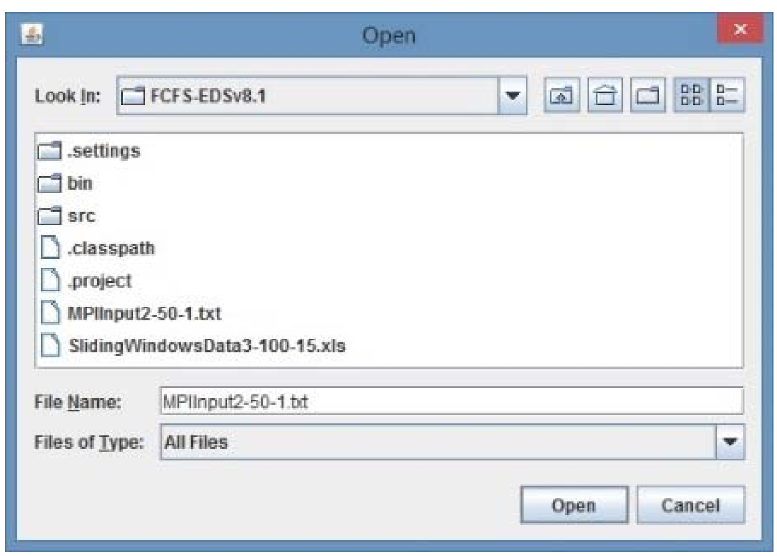

Fig. 5: The window for opening input file

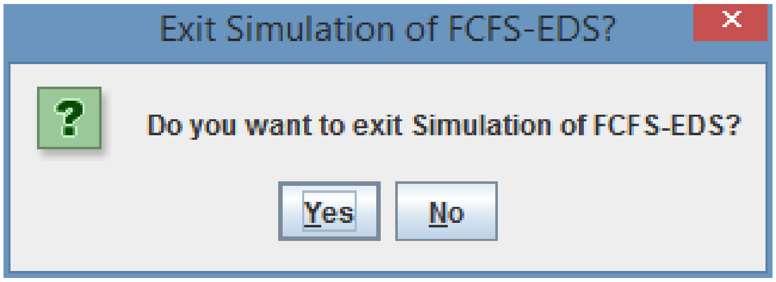

Fig. 6: Dialog box for exiting the software

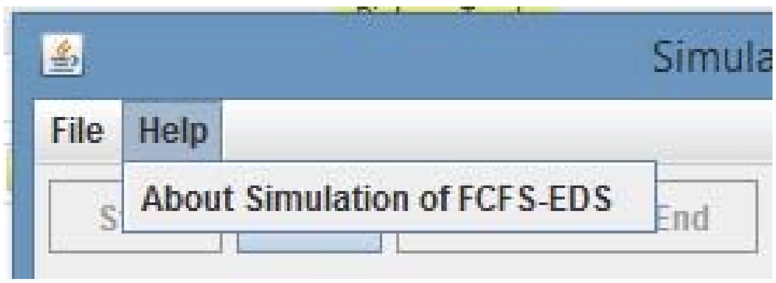

Fig. 7: Help menu

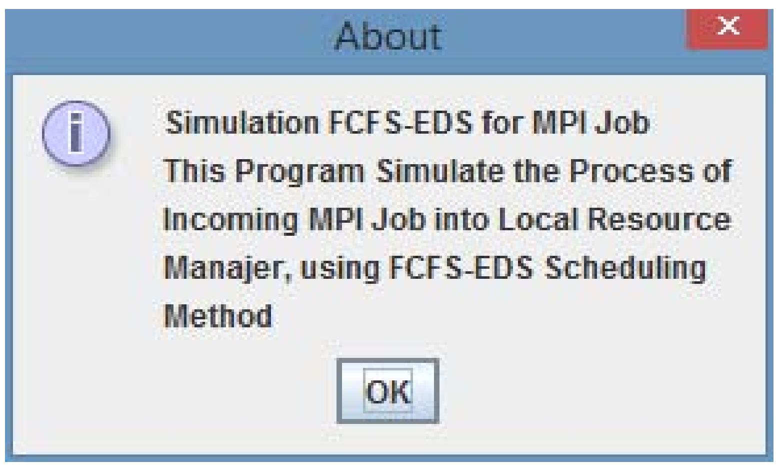

Fig. 8: About simulation of FCFS-EDS

The area of simulation for FCFS-EDS scheduling for MPI job: in this area consist of 4 compartments. The first compartment, used for visualize FCFS-EDS scheduling for MPI job in a logical view. The second compartment. Used for visualize FCFS-EDS scheduling for MPI job in a physical view. The third compartment is containing the running process for scheduling MPI job. In this area we show the incoming job in the particular time slot in the form of text. Also, it shows where the job is allocated. Visualization is in the compartment of logical and physical view. The last compartment show the efficiency graph in the real time mode. How to draw the graph will be done in the next research.

Above the logical view we have three buttons, i.e., start button, t+1 button and start till the end button as depicted in Fig. 9. Start button is enabled only there is an opened input file, otherwise it is disabled. If we have opened the input file, then start and start till the End buttons will be enabled. If we click the start button, then for the next step of simulation we have to click t+1 button but if we click the start Till the end button, then the simulation will run from the first time slot to the last time slot.

The example of the running simulation can be seen in Fig. 10. In Fig. 10, we can see that in the logical view part, there are MPI jobs has been scheduled. As we can see,

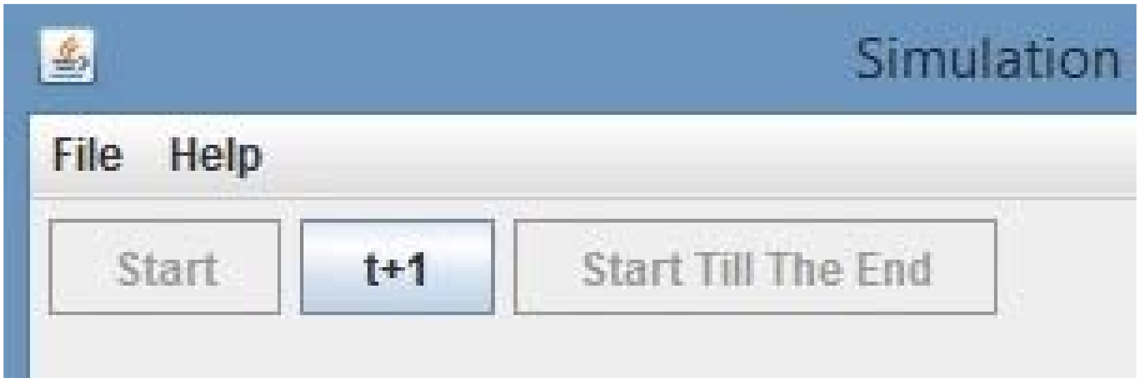

Fig. 9: Buttons in simulation of FCFS-EDS for MPI job

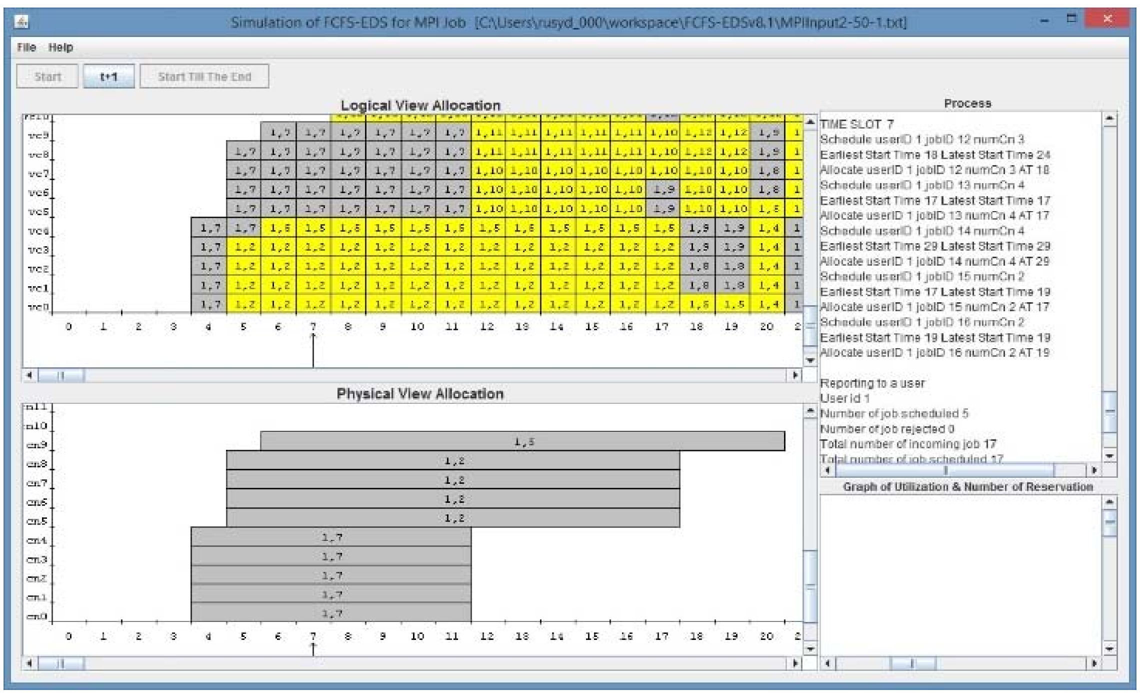

Fig. 10: Simulation of FCFS-EDS scheduling for MPI job

JobId 1.7 has been allocated in time slot number 4 in five virtual compute nodes (vc0 till vc4) and in time slot 5 has been allocated in five virtual compute nodes (vc4 till vc8). When we execute this job (JobId 1.7) it means that we are in time solt 4 (t = 4) then JobId 1.7 is executed in 5 compute nodes (cn0 till cn4) these compute nodes is dedicated for JobId 1.7 until time slot 11 where JobId 1.7 finished running on time slot 11.

CONCLUSION

After performing black box test we find that the software is running flawlessly. The conclusion that can be drawn from this research: we have developed the software for simulation FCFS-EDS scheduling for MPI job in the grid system, namely simulation of FCFS-EDS scheduling for MPI job. This software has the capability to visualize step by step the scheduling process for MPI job using FCFS-EDS method both in the logical view and physical view.

ACKNOWLEDGEMENTS

This research was supported by Directorate General of Higher Education (Dikti) of Indonesian Government. Contract number 011/HB-LIT/III/2016, 15th March 2016.

REFERENCES

Buyya, R. and A. Sulistio, 2008. Service and utility oriented distributed computing systems: Challenges and opportunities for modeling and simulation communities. Proceeding of the 4th Annual Symposium on Simulation ANSS 2008, April 13-16, 2008, IEEE, Melbourne, Australia, ISBN:978-0-7695-3143-4, pp: 68-81.

Buyya, R., D. Abramson and J. Giddy, 2000. Nimrod/G: An architecture for a resource management and scheduling system in a global computational grid. Proceedings of the 4th International Conference and Exhibition on High Performance Computing in Asia-Pacific Region, May 14-17, 2000, Beijing, China, pp: 1-7.

Daniel, M., 2005. A Networking Approach to Grid Computing. John Wiley & Sons, Hoboken, New Jersey,.

Foster, I. and C. Kesselman, 1999. The Grid: Blueprint for a New Computing Infrastructure. 1st Edn., Morgan Kaufmann Publisher, San Fransisco, ISBN-10: 1558604758, pp: 675.

Foster, I. and C. Kesselman, 2003. The Grid 2: Blueprint for a new Computing Infrastructure. Elsevier, Amsterdam, Netherlands,.

Foster, I., A. Roy and V. Sander, 2000. A quality of service architecture that combines resource reservation and application adaptation. Proceedings of the 8th International Workshop on Quality of Service, June 2000, Italy, pp: 12-20.

Hwang, K. and Z. Xu, 1998. Scalable Parallel Computing: Technology, Architecture, Programming. WCB/McGraw-Hill, Boston, Massachusetts,.

Isah, A.Z. and H. Safwana, 2012. Resource management in grid computing: A review. Greener J. Sci. Eng. Technol. Res., 2: 24-31.

Jacob, B., L. Ferreira, N. Bieberstein, C. Gilzean, J.Y. Girard, R. Strachowski and S. Yu, 2003. Enabling Applications for Grid Computing with Globus. IBM Crop, Riverton, New Jersey, USA.

Kan, G., 2001. Gnutella, Peer-to-Peer Harnessing the Power of Disruptive Technologies. O'Reilly Publications, UK., pp: 94-122.

Mu'alem, A.W. and D.G. Feitelson, 2001. Utilization, predictability, workloads and user runtime estimates in scheduling the IBM SP2 with backfilling. IEEE Trans. Parallel Distri. Syst., 12: 529-543.

Schopf, J.M., 2004. Ten Actions when Grid Scheduling: The User as a Grid Scheduler. In: Grid Resource Management: State of the Art and Future Trends, Nabrzyski, J., J.M. Schopf and J. Weglarz (Eds.). Chapter 2, Kluwer Academic Publisher, Norwell, MA., USA., ISBN-13: 9781402075759, pp: 15-24.

Schwiegelshohn, U. and R. Yahyapour, 2004. Attributes for Communication between Grid Scheduling Instances. In: Grid Resource Management, Nabrzyski, J., J.M. Schopf and J. Weglarz (Eds.). Springer, New York, USA., ISBN:978-1-4613-5112-2, pp: 41-52.

Smith, W., I. Foster and V. Taylor, 2000. Scheduling with advanced reservations. Proceeding of the 14th International Symposium on Parallel and Distributed Processing IPDPS 2000, May 1-5, 2000, IEEE, Illinois, USA., ISBN:0-7695-0574-0, pp: 127-132.

Sulistio, A. and R. Buyya, 2004. A grid simulation infrastructure supporting advance reservation. Proceeding of the 16th International Conference Parallel and Distributed Computer System, (INPDCS'04), Australia, pp: 1-7.

Tiemeyer, M.P. and J.S. Wong, 1998. A task migration algorithm for heterogeneous distributed computing systems. J. Syst. Software, 41: 175-188.

Umar, R., A. Agarwal and C.R. Rao, 2012. Advance planning and reservation for parametric (MPI) jobs in a cluster/grid system. J. Networking Technol., 3: 129-138.

Yang, J., Y. Bai and Y. Qiu, 2007. A decentralized resource allocation policy in minigrid. Future Gener. Comput. Syst., 23: 359-366.