Suresh Gyan Vihar University, Jaipur
International Journal of Converging Technologies and Management (IJCTM)
Volume 3, Issue 2, 2017
ISSN: 2455 - 7528

Load balancing on Computational Grid Using Advanced Reservation of Resources

Sophiya Sheikh
Department of Computer Science, Central University of Rajasthan, Kishangarh, Ajmer, India
sophiya.sheikh@gmail.com

Abstract- Grid computing refers as distributed computing that works upon the integration and collaboration of geographically-dispersed resources. Computational grid plays a vital role in resource sharing and load balancing among resources. To manage such a high intensive heterogeneous environment resource allocation should be not only scalable but also highly reliable. It is essential need of heterogeneous environment that all resources utilize optimally to balanced load among resources. Another requirement of heterogeneous computing is to minimized task waiting time in order to well regulated services to the users. In this paper we proposed Advanced Reservation of Resources Algorithm (ARRA) that commits advanced reservation of resources to users with minimized task waiting time. Simulation study reveals the motivation of algorithm with respect to load balancing. The performance of the algorithm is evaluated using performance criteria processing cost and task waiting time. A comparison of our proposed approach with a classical approach called dynamic task scheduling is provided. Experimental results show the success of proposed algorithm in terms of resource allocation.

Keywords—Computational Grid; Load Balancing; Waiting Time; Advanced Reservation;

I. INTRODUCTION

The usage of grid computing has been exponentially increasing in science, industries and in various others applications related to working in distributed environment. Grid system are classified in two categories compute grid and data grid. Compute grid focuses on compute intensive problem while data grid is used to manage data over a huge geographical network. In grid environment resource allocation should be in a manner so that we can fully utilize the capability of resources. A computational grid provides user a platform to execute the compute intensive jobs which cannot be executed at the user's end. It has become most common and widely adapted technique to solve computational problem occurred in scientific and technical areas [1]

In view of the proper usages of underutilized resources, allocation technique of resources must be efficient enough to process large no of computational jobs. To enhance performance of resources in decentralized architecture task scheduling needs to be done efficiently [3]. The basic authority of grid system is to manage all resources effectively. Grid is responsible for accepting user requests; allocate it to most suitable and available resources [2]. Job scheduling and load balancing are the key issues in grid computing. Allocating the task to the right resources at the right time is always a key issue with grid computing [4].

In this paper we present Advanced Reservation of Resources Algorithm (ARRA) that minimized task waiting time, equal load distribution, and enhanced processors capability. The main objective of this paper is effective resources allocation and utilization in terms of cost (MIPS). Grid is designed to work in geographically dispersed environment which is dynamic and heterogeneous. Millions of users are waiting to execute their tasks. So we need to achieve some objective.

1. Resources need to be fully utilized in order to process a large number of compute intensive jobs.
2. Waiting time should be minimized for the users to execute their tasks.
3. Balanced load among processors to properly utilized all heterogeneous resources.

ARRA is advanced commitment of resources in order to reduced waiting time for user to execute their task on time. Proposed approach also balanced load among resources and fully utilized resources with high execution rate given in MIPS(Million Instruction Per Seconds).

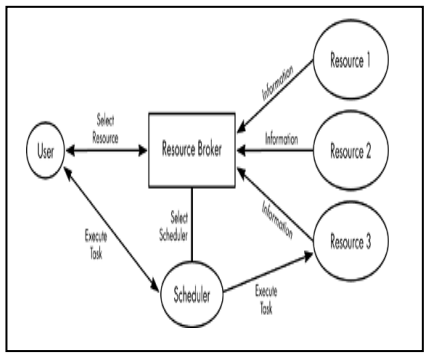

Figure-1 Resource Allocation in Computational Grid

Rest of the paper organized as follows. Section II defines some related work reported in literature. Section III defines proposed algorithm with description. Section IV describes experimental set up with simulation study and results. The paper ends with Section V that have conclusion of the paper with necessary remarks.

II. RELATED WORK

This section will refine various traditional and dynamic approaches related to load balancing, efficient task execution time and resource allocation. Scheduling in a heterogeneous and dynamic environment is proven to be NP complete. Henceforth many techniques for effective solutions for scheduling have been proposed. These heuristic can be either static or dynamic. Many Swarm intelligence and soft computing approaches have been used in larger scale for solving scheduling and resource allocation problem.

A. Nature Inspired Optimization and Artificial Intelligence Techniques

There are several nature inspired optimization techniques used in area of optimization and load balancing. Ant Based Heuristic Approach to scheduling & workload distribution (AHSWDG) for efficient load balancing among the available resources has been proposed [5]. This scheduling approach allocated task to the most optimal grid resources for faster task allocation. In [17] author proposed NSGA II with Fuzzy Adaptive Mutation Operator for task scheduling and load balancing. Proposed approach implemented to deal with implemented to deal with independent task assignments problems in parallel distributed computing systems. In [14] an artificial life technique using genetic algorithm and tabu search for load balancing has been introduced. Proposed approach is optimized and can handle complexity of heterogeneous grid environment. In [9] multi-objective genetic algorithm is known as NSGA-II using fuzzy operators has been improved for better quality and performance of task scheduling in heterogeneous grid environment.

B. Dynamic Load balancing

Dynamic load balancing techniques where scheduler allocating the jobs at the time of execution making it more complicated to achieve load balancing and minimize job execution time. In [8] author presented a scheduling algorithm for minimize completion time of jobs and maximize resource utilization. Proposed schedule_DLB to balance workload among different clusters presented in a grid system. Another load balancing scheme states that regional grids which are clusters around broker sites considered network transfer delay and organized in fully decentralized fashion[21]. In order to achieve static and dynamic load distribution and redistribution in efficient way author proposed hybrid policy for job scheduling.In [15] Shah and Veeravalli proposed a decentralized, scalable, adaptive and distributive approach for balancing load among resources. They proposed two load balancing algorithm modified ELISA and load balancing on arrival.

C. Classical Approaches

A Load balanced Min-Min algorithm for task scheduling firstly schedules the tasks then remaining task which have not executed yet will be rescheduled [20]. A comparison of eleven static scheduling heuristic for independent task has been proposed in heterogeneous environment [4]. availability of machine with balanced load. The proposed new approach called Minimum Completion Time (MCT) which is the combination of Opportunistic Load Balancing (OLB) algorithm and Minimum Execution Time (MET) algorithm. A new decentralized scheduling scheme has been introduced which is not only useful for local scheduling but also for queue balancing [18]. Main emphasis is on backfilling across multiple nodes. In [11] author proposed a hybrid approach to combine two class of load balancing strategy namely average-based and instantaneous approaches. Clients can send their request to central resource management unit(RMU). RMU perform scheduling for computing node. Each computational node (CN) has processor to execute tasks. Finally result return to RMU. In [10] scheduling algorithm for hierarchical load balancing has been presented for dividing computational load in efficient way. The algorithm addresses two important aspects, one is Multiple Parameters based Load Balancing (MPLB) and another is job scheduling at Logical Hierarchical Levels (LHL). Proposed load balancing technique implemented over a tree based grid model. Results show that Hierarchical Job Scheduling (HJS) approach is more efficient as compared to Flat Structure Job Scheduling (FJS) [7].

C. Deadline based approaches

Some algorithms are designed to control task execution time within given deadline and budget. In this paper [13] author proposed a load balancing approach deals with both system and individual optimal objectives. Some problems exist with EGDC like gridlet executes below the lower range of normal loaded range of resources and no precisely description that how gridlet scheduling will take place henceforth an improved Enhanced gridsim with Deadline Control proposed [10]. In [12] author proposed approach for QOS based resource allocation for grid infrastructure with constraints like budget and deadline.

III. PROPOSED ALGORITHM

This section describes ARRA with motivation of implementation. In this research we are suggesting a new allocation technique called advanced reservation of resources dynamically. In the previous approaches resources are allocate dynamically to tasks. Proposed strategy works on the basis of advanced commitment of available resources to users. We are simulating the experiment in gridsim. In this approach we first simulate resources then users and their jobs. Then

scheduler committed available resources in advanced to jobs dynamically to minimize waiting time, increased processors utilization with balanced load. Simulation is proposed in two sections initializing entities and advanced reservation.

```
ARRA()
{
Start Execution; Initialize Entities
getTotalUsers() ; getSimulationTime()
getStartTime() ; getEndTime()
If(SimulationTime>StartTime && SimulationTime>EndTime)
createResource()
{
        For k=0 to resource_size-1 do
                Set Resource name, Number of machine, MIPS rating; Set time zone and processing elements
                for each resource; Set scheduling policy as space shared
                k++
        End For
}
End If
Create User()
{
        For i=0 to User_Size-1 do
                If (userid%2==0)
                        TimeZone = 8.0 //Time zone to GMT+8; TotalJob = 5
                Else
                        TimeZone=-3.0 //Time zone to GMT-3; TotalJob = 4
                End If
        Create user entity ; Create Gridlet for each user ; Store user entity in user list
        i++
        End For
}
Start Simulation
Advanced Reservation ()
{
For i=0 to user_size-1 do
        For i=0 to resource_size-1 do
                //Checking for advanced Reservation of resources for first user
                if (QueryResult = “AdRes_STATUS_NOT_COMMITTED”)
                /*Check reservation for other user*/
                if(CommitResult= “AdRes _COMMIT_SUCCESS”)
                /*Check for query result*/
                if(QueryResult= “AdRes _STATUS_NOT_STARTED”)
                /*Check another resources for availability*/
                if(QueryResult= “AdRes_STATUS_ACTIVE”)
                Allocate resources to user  ; Get allocation cost in MIPS
                End if ; End if ; End if ; End if; End for
End for
}
PrintGridlet()
For j=0 to user_size-1
        For k=0 to gridlet-1
                        Print Gridlet id, status; Print ResourceId, ProcessingCost in MIPS
        End for
End for
}
```

Suresh Gyan Vihar University, Jaipur
International Journal of Converging Technologies and Management (IJCTM)
Volume 3, Issue 2, 2017
ISSN: 2455 - 7528

A. Initializing Entities

Prior to reservation whole environment need to be created. Firstly we need to get total number of users and current time of simulation. Current simulation time will be compare with start time and end time of reservation of resources. If current simulation time is greater then start time and end time of reservation then we will start resource creation. Each resource has multiple processing elements and time zone. We need to assign MIPS rating (Millions instruction per second) to every resource. MIPS rating will indicate the processing capability of resource. In our experiment each resource contains single machine but we can also include multiple machines if needed. Our next task is to create user entities. Each user has some jobs that are called gridlets. We have resources from different time zone. Even number of users has their time zone to GMT+8 and total gridlets 5 and Odd number of users has their time zone to GMT-3 and total gridlets 4. This is how we create first user entity with its task and store it to user list.

A. Advanced Reservation

Our next step is to allocating task to resources in advanced manner based upon space shared allocation policy. Every user has some task or gridlets to execute. In proposed approach we will commit resources with some allocation cost before task execution started. For the first user we will check for the availability of first resource. If resource is not available to commit in advanced then we will check for the next user that also has some gridlets to execute. The checking process is dynamic for all gridlets and resources. If resource committed successfully for a user then we will check results for internal query that whether reservation has started or not. If resource has committed to task in advanced, ready for execution and status is active then resource will be allocated to the gridlet. Similar procedure will run till all users get resources in advanced for the execution of their gridlets.

IV. EXPERIMENTAL SETUP AND RESULTS

A. Gridsim Simulator

The experiments were conducted using gridsim 5.2 on i7 processors having seven cores and 8GB RAM. The GridSim toolkit allows modelling and simulation of entities in parallel and distributed computing (PDC), peer to peer computing (P2P) and multi cluster computing for design and implementation of task scheduling algorithms. In grid system resources are heterogeneous in nature that can be aggregated using resource brokers. A resource consist one or more processing elements with shared or distributed memory [6]. Resources are managed by scheduler that could be either time shared or space shared. The processing nodes within a resource can be heterogeneous in terms of processing capability, configuration, and availability. The resource brokers uses different task scheduling algorithms or models for mapping task to resources to achieve both system and user objective [16]. In experiment we are simulating 5 users and resources. Each resource consist single machine. Each user has some time zone and gridlets. The total number of gridlets would be either 4 or 5. Length of gridlets can be vary from 0 to 50,000 MI. Resources have different processing elements, time zone and MIPS ratings for task. Various parameters used in simulation given from Table-1 to Table-3.

Table I

Gridlet Parameters	
Length	0 to 50,000 MI
File Size	Varies within this range 100 + (10% to 40%)
Output Size	Varies within this range 100 + (10% to 50%)

Table II

Simulation Parameters	
No of Machines	1
No of Resources	5
No of Users	5

Resources Specifications				
Resource Name	Total PE	No of machine	Time Zone	MIPS
Resource 0	4	1	10.0	515
Resource 1	13	1	10.0	684
Resource 2	16	1	1.0	410
Resource 3	6	1	1.0	410
Resource 4	8	1	-6.0	377

No of Gridlet	4 to 5

Table III

B. Results

We are simulating five resources and each resource consist single machine. Each resource is scheduled by time shared allocation policy. Each resource has its own task execution rate given in MIPS and processing elements. Proposed ARRA compared with Dynamic Task Scheduling (DTS). In figure-2 we present the simulation results of dynamic task scheduling allocation policy while in figure-3 we present the simulation results of ARRA. In figure-2 we can see that task execution rate is less with more waiting time. Load is also not distributed equally. While in figure-3 task execution time is high with minimum waiting time and equal load distribution.

Results show that when we simulating dynamic task scheduling approach we can get some hundred MIPS speed of task execution while when we combined dynamic and advanced resource allocation policy we can get many thousands MIPS speed of task execution.

The motivation behind using ARRA is resources are committed in advance so that task waiting time could be minimized which is essential need of geographically dispersed environment. Another advantage is that as we reserved resources in advanced so that we can check that which resource is over utilized and which is underutilized. Therefore we can make proper utilization of resources with balanced load.

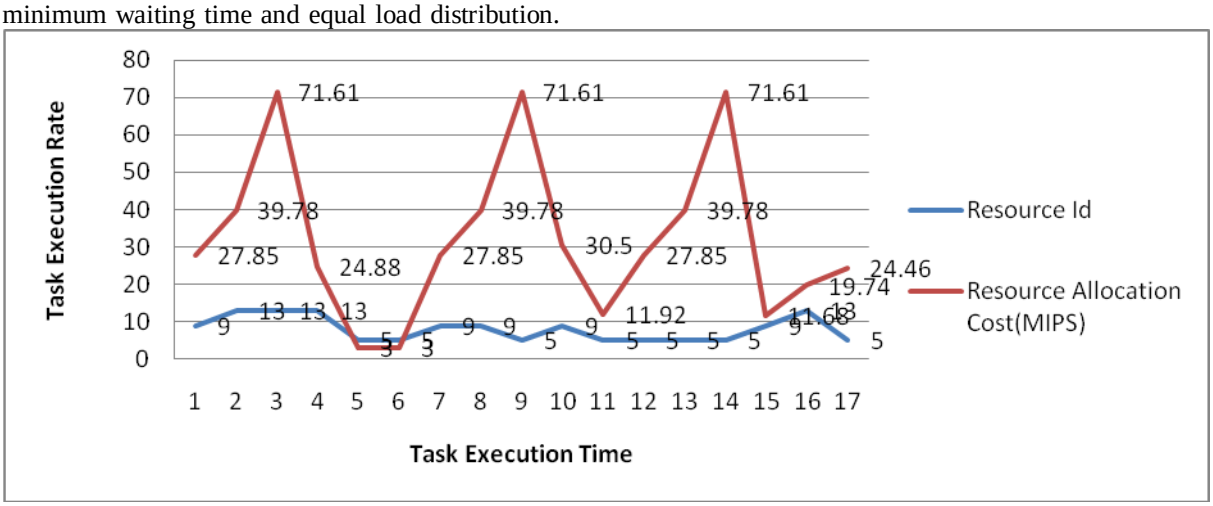

Fig-2: Dynamic Task Allocation Policy

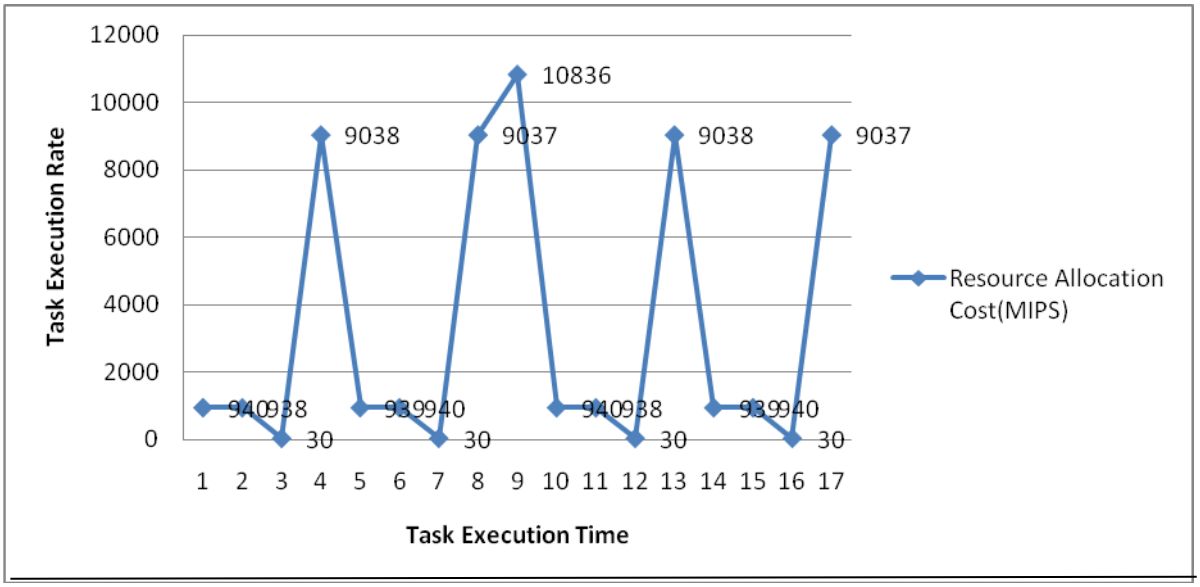

Fig-3: Advance Reservation Allocation Policy

Performance optimization criteria are processing cost, waiting time and equal processors utilization.

1) Enhanced Task Execution Rate

Results show that task execution rate increased exponentially in terms of MIPS. Task execution rate is negligible for DTS as compared to ARRA.

2) Minimized Task Waiting Time

Advanced Reservation of resources is advanced and dynamic allocation of resources. In proposed approach resources are committed in advanced it will minimized task waiting time for resources.

3) Efficient Load Balancing

Due to advanced commitment of resources we can check the load among various processors, so that we can distribute load equally among processors.

C. Observations

- We observed that we can get equal utilization from all resources by ARRA. We have tested it for a group of five resources, and we find that all resources are utilized equally so that in such a huge and geographically distributed architecture it is also possible that by ARRA we ensure proper resource utilization.
- It is also an ideal load distribution technique. ARRA is a dynamic scheduling algorithm and we observed that it optimized resources in efficient way; in a trial we get the equal load distribution among resources. We have tested it for five resources although the order of resource allocation could be changes as it is dynamic but the resources allocation cost always remain constant in some cases.
- Advance reservation of resources reduces waiting time and increases processors capability exponentially.

V. CONCLUSION

The key objective of grid environment is efficient utilization of resources. The heterogeneous system is only useful if we properly exploit its all resources. In this research we have investigated the effectiveness of advanced reservation of resources in grid environment. In traditional dynamic task scheduling task execution rate is some hundred millions instruction per second while in proposed ARRA task execution rate is thousand million instructions per seconds. Results show that with proposed ARRA processing capability of resources increased exponentially. The proposed strategy is quite suitable for applications where minimum task waiting time is essential requirement. The task will be never in task waiting queue as resources are allocated in advanced before execution started. The advanced commitment of resources properly utilizes resources with equal load distribution. Performance evaluation criteria are task waiting time, resource utilization, and processing cost. Simulation study reveals that ARRA is designed to work well in heterogeneous and dynamic grid environment.

REFERENCES

[1] Wu, Ming, and Xian-He Sun. "Memory conscious task partition and scheduling in grid environments." Proceedings of the 5th IEEE/ACM International Workshop on Grid Computing. IEEE Computer Society, 2004.

[2] Xhafa, Fatos, and Ajith Abraham. "Computational models and heuristic methods for Grid scheduling problems." Future generation computer systems, vol. 26, no. 4, pp. 608-621, 2010.

[3] Foster, Ian, Carl Kesselman, and Steven Tuecke. "The anatomy of the grid: Enabling scalable virtual organizations." International journal of high performance computing applications, vol. 15, no .3, pp. 200-222, 2001.

[4] Braun, Tracy D., Howard Jay Siegel, Noah Beck, Ladislau L. Bölöni, Muthucumaru Maheswaran, Albert I. Reuther, James P. Robertson et al. "A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems." Journal of Parallel and Distributed computing, vol. 61, no. 6, pp. 810-837, 2001.

[5] Saxena, Rohit, Ankur Kumar, Anuj Kumar, and Shailesh Saxena. "AHSWDG: An Ant Based Heuristic Approach to Scheduling and Workload Distribution in Computational Grids." In Computational Intelligence & Communication Technology (CICT), 2015 IEEE International Conference on, pp. 569-574. IEEE, 2015.

[6] Krauter, Klaus, Rajkumar Buyya, and Muthucumaru Maheswaran. "A taxonomy and survey of grid resource management systems for distributed computing." Software: Practice and Experience, vol. 32, no. 2, pp. 135-164, 2002.

[7] Reddy, K. Hemant Kumar, and Diptendu Shina Roy. "A hierarchical load balancing algorithm for efficient job scheduling in a computational grid testbed." In Recent Advances in Information Technology (RAIT), 2012 1st International Conference on, pp. 363-368. IEEE, 2012.

[8] Lu, Kai, and Albert Y. Zomaya. "A hybrid policy for job scheduling and load balancing in heterogeneous computational grids." Parallel and Distributed Computing, 2007. ISPDC'07. Sixth International Symposium on. IEEE, 2007.

[9] Salimi, Reza, Homayun Motameni, and Hesam Omranpour. "Task scheduling using NSGA II with fuzzy adaptive operators for computational grids."Journal of Parallel and Distributed Computing, vol. 74, no. 5, pp. 2333-2350, 2014.

[10] Patel, Deepak Kumar, and Chitaranjan Tripathy. "An improved approach for load balancing among heterogeneous resources in computational grids."Engineering with Computers, vol. 31, no. 4, pp. 825-839, 2015.

[11] Li, Yajun, Yuhang Yang, and Rongbo Zhu. "A hybrid load balancing strategy of sequential tasks for computational grids." Networking and Digital Society,

2009. ICNDS'09. International Conference on. Vol. 1. IEEE, 2009.
[12] Xiao, Peng, and Zhigang Hu. "A novel QoS-based co-allocation model in computational grid." IEEE GLOBECOM 2008-2008 IEEE Global Telecommunications Conference. IEEE, 2008.
[13] Chatrapati, K. Shahu, J. Ujwala Rekha, and A. Vinaya Babu. "Competitive equilibrium approach for load balancing a computational grid with communication delays." Journal of theoretical and applied Information Technology, vol. 19, no. 2, pp. 126-133, 2010.
[14] Subrata, Riky, Albert Y. Zomaya, and Bjorn Landfeldt. "Artificial life techniques for load balancing in computational grids." Journal of Computer and System Sciences, vol. 73, no. 8 pp. 1176-1190, 2007.
[15] Shah, Ruchir, Bhardwaj Veeravalli, and Manoj Misra. "On the design of adaptive and decentralized load balancing algorithms with load estimation for computational grid environments." IEEE Transactions on parallel and distributed systems, vol. 18, no. 12, pp. 1675-1686, 2007.
[16] Simone A. Ludwig and Azin Moallem. "Swarm intelligence approaches for grid load balancing." Journal of Grid Computing, vol. 9, no. 3, pp. 279-301, 2011.
[17] Salimi, Reza, Homayun Motameni, and Hesam Omranpour. "Task scheduling with Load balancing for computational grid using NSGA II with fuzzy mutation." In Parallel Distributed and Grid Computing (PDGC), 2012 2nd IEEE International Conference on, pp. 79-84. IEEE, 2012.
[18] Abraham, Ajith, Rajkumar Buyya, and Baikunth Nath. "Nature's heuristics for scheduling jobs on computational grids." In The 8th IEEE international conference on advanced computing and communications (ADCOM 2000), pp. 45-52. 2000.
[19] Lee, Jaehwan, Pete Keleher, and Alan Sussman. "Decentralized dynamic scheduling across heterogeneous multi-core desktop grids." In Parallel & Distributed Processing, Workshops and Phd Forum (IPDPSW), 2010 IEEE International Symposium on, pp. 1-9. IEEE, 2010.
[20] Kokilavani, T. and Amalarethinam, D.D.G., 2011. Load balanced min-min algorithm for static meta-task scheduling in grid computing. International Journal of Computer Applications, vol. 20. no. 2, pp.43-49.
[21] Rajavel, Rajkumar. "De-centralized load balancing for the computational grid environment." In Communication and Computational Intelligence (INCOCCI), 2010 International Conference on, pp. 419-424. IEEE, 2010.