INTERNATIONAL JOURNAL OF SCIENTIFIC PROGRESS AND RESEARCH (IJSPR) ISSN: 2349-4689
Volume 30, Number 02, 2016

An Effective Load Balancing with Conservative Backfilling using Random Search in Grid Computing

Anurag Shrivastava
Associate Professor
NRI Institute of Research and Technology Bhopal (M.P.) INDIA

Vaibhav Patel
Assistant Professor
NRI Institute of Research and Technology Bhopal (M.P.) INDIA

Praneet Khare
M.Tech Scholar
NRI Institute of Research and Technology Bhopal (M.P.) INDIA

Abstract - Grid computing, one of the most contemporary phrase used in IT, is appearing enormously distributed computational paradigm. Despite a number of profits in grid computing, still resource allotment is a challenging task in the grid. This work defines to increase the benefits by analyzing how the tasks are allocated to grid resources effectively according to quality of service parameter and gratifying user requisition. A method of conservative backfilling load balancing with random search algorithm has introduced to answer the above raised question about the resource allocation problem based on grid user requisition. The result of proposed method of CNBF-RS algorithm ameliorates the grid resource allocation.

Index Terms: Grid Computing, Heterogeneous Resource, Conservative Backfilling Load Balancing, Random Search, Grid Resource Allocation.

1. INTRODUCTION

Grids are distributed systems that admit users to connect with resources owned by different organizations. Grid scheduling, that is, the allocation of distributed computational resources to user function, is the most challenging and difficult work in Grid computing. Nowadays, several are the real-life applications in which Grids are involved; some practical fields are protein folding, weather modeling, and satellite image processing. In grid architecture, users submit requests for task execution from any one of a number of sites. At each site, besides the local distributed system, the system miniature is consists by three components: an External Scheduler (ES), which is liable for finding a particular site where a submitted work can be executed; a Local Scheduler (LS), is liable for resolving the order in which tasks are executed at that particular site; a Dataset Scheduler (DS), it is liable for determining if and when to replicate data and/or erased local files. Resource site consist, in general assorted computing resources interconnect by vendor independent networks. In general, on receipt of a task request, the ES examine the LSs by double check whether the task can be performed on the available resources and meet the user stated due date. If this is the case, stated site in which executing that task is chosen. Otherwise, the ES attempt to locate LS of a site, controlled by another ES that can meet the task processing requirements, through search mechanisms. If a LS cannot be placed in between a preset number of search steps, the task request is either dropped or move to another LS that can minimize the due date failure lay on a task request parameter. When a satisfactory site is located, the task request is move from the ES to this site and managed by the correlated LS.

In Following Chapter 2 gives literature reviews of the different method of load balancing, this entire scheme entitled with their authors name and respective title. Chapter 3 describes the process of proposed method with algorithm and corresponding diagrams. Chapter 4 describes the implementation details and information about used data set. The implementation details are also explained out in terms of the algorithm process. Chapter 5 describes the result and analysis of the proposed work in terms of performance metrics. Chapter 6 describes the conclusion of the proposed work and also describes the future work. Chapter 7 specifies the references of research paper, which use the corresponding details in our work.

2. LITERATURE SURVEY

Gnanasekaran et. al[1], Grid Computing emerged as a vast scale distributed system to suggested dynamic coordinated resources sharing and high performance computing. Grid coordinates the resources that are not related to centralized control. It applies standard open, general purpose protocols and its interfaces. Grid sends non-trivial qualities of service such as the response time, throughput, availability and security.

Caramia et al. [2], Grid scheduling stands for the allocation of distributed computational resources to user applications. It’s the most challenging and complex task in Grid computing. The problem of allotment of resources in Grid scheduling requires the definition of a model that permits local and external schedulers to communicate and achieve an efficient management of the resources themselves. To achieve this aim, some economic/market-based models have been introduced

Keerthika et. al[3], This research paper propose a new scheduling methodology for computational grids that reduces load balancing, fault tolerance and user satisfaction deals with grid architecture, resource heterogeneity, resource availability and task characteristics such as user deadline. This algorithm minimizes the makespan of the job along with user satisfaction and also balanced the load.

3. PROPOSED METHODOLOGY

The basic algorithm of proposed work CNBF-RS is as follows:

[M'] = CNBF-RS (Q, M)
// Q is the queue for incoming jobs
// M is a map between jobs and resource nodes
// M' is the updated allocation map
Step 1: Initialize the status of all nodes (N).
Step 2: Initial status=.Previous
Step 3: While jobs=N and N>0 do
Step 4: if Current state is ready to change then
Step 5: Current = Get change state (); //Computation stage
Step 6: Threshold = generate threshold (upper bound, lower bound); //Load Balancing
Step 7: Now Random Search Optimization technique apply on Q.
7.1 Initialize random search parameter θ_0 on the basis of current state and threshold value, initial point $X_0 \subset Q$ and iteration index k=0.
7.2 Generate a collection of candidate points $V_{k+1} \subset Q$ according to a specific generator and associated sampling distribution.
7.3 Update X_{k+1} based on the candidate points V_{k+1}, previous iterates and algorithmic parameters. Also updates algorithm parameters θ_{k+1}.
7.4 If a stopping criterion is met then stop, otherwise increment k and return to step 7.2.
Step 8: Through Random Search, find the optimized job set in queue Q.
Step 9: Get first job j from Q.
Step 10: while $j \neq null$
Step 11: $N_j \leftarrow$ the number of nodes required by j;
Step 12: $N_{idle} \leftarrow$ the number of idle nodes;
Step 13: if $N_j \leq N_{idle}$ then remove j from Q and dispatch it to any N_j idle nodes; Updates M accordingly;
if j is not at the head of Q then
insert j into $Q_{backfill}$;
else
$N_{backfill}$<- the number of nodes running jobs arriving later than j;
if $N_j \leq (N_{backfill} + N_{idle})$ then
Suspend jobs in $Q_{backfill}$ that arrive later than j and move then back to Q According to descending order of their arrival time until the number of Idle node is greater than N_j;
Remove j from Q and dispatch it to N_j idle nodes;
Update M;
Step 14: j<- get the next job from Q and goto step 10.
Step 15: Go to step 3 with decrement list of nodes N.
Step 16: M' = M;
Step 17: Finally obtain the updated jobs and resource node allocation map with required load balancing.

4. EXPERIMENTAL STEP:

Load Balancing components have been developed which executes in simulated grid environment. This application has been developed using Java and Netbeans. GridSim simulator is an event-based modular, consists of independent entities which use the desired simulation functionality (see Figure 1). It consists of the centralized scheduler; the grid resource(s) with the local job allocation strategy, the job, the machine, failure loader and various classes are responsible for the simulation setup, the visualization and the generation of simulation output. By now, the Grid nodes are not directly simulated but the job loader entity used as a main class for the future operation of the Grid user. Simulator's behavior is driven by the event-based message passing protocol. For each simulated event, such as the job arrival or the job completion, one message defining this event is created. It contains the identifier of the message recipient, the type of the event, the time when the event will occur and the message data.

Fig 1: Main Parts of the GridSim 5.0.2 Simulator

The JobLoader class of GridSim supports several trace formats including the Grid Workloads Format (GWF) of the Grid Workloads Archive and the Standard Workloads Format (SWF) of the Parallel Workloads Archive.

5. RESULT ANALYSIS

The analysis of the existing work (FCFS, EASY-Backfilling) and the proposed work (CNBF-RS) on different parameters are given in following tables. We can compare different load balancing technique with our proposed technique. On the basis of load balancing with scheduling method, FCFS (First Come First Serve) and EASY Backfilling used as a existing technique and Conservative Backfilling with Random Search (CNBF-RS) used as a proposed method. There are four parameters are here for compare results of different approaches.

(1) Average Machine Usage

(2) Job Cluster Uses/Day

(3) Number of Requested and Used CPU

(4) Number of Waiting and Running Jobs

Table 1: Average Machine Usage (in Percentage) with Different Load Balancing Strategies

DAYS	FCFS	EASY BACK FILLING	PROPOSED METHOD(CNBF-RS)
0-5	0.6	0.8	0.6
5-10	40	40	40
10-15	1.8	1.8	2
15-20	35.6	31.6	32
20-25	35.2	36.8	34.5
25-30	14.6	15	10.5
30-35	39.8	35.8	35
35-40	12.2	12.6	9.6
40-45	21.4	21.6	20.8
45-50	42.6	44.4	44
50-55	29.8	31	30
55-60	23.8	28.2	23.4

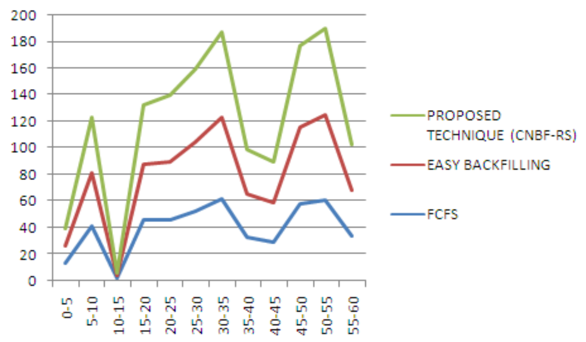

Fig 2: Average Machine Usage (in Percentage) with FCFS, EASY Backfilling and Conservative Backfilling with Random Search Strategies

As per above graph, proposed technique conservative backfilling with random search has more utilize machine resources like CPU and other equipments as compare than FCFS and EASY Backfilling method.

Table 2: Job Cluster Uses per Day (in Percentage) with Different Load Balancing Strategies

DAYS	FCFS	EASY BACKFILLING	PROPOSED TECHNIQUE (CNBF-RS)
0-5	13.2	12.8	13.2
5-10	40.6	40.4	42.4
10-15	2.2	1.8	2.2
15-20	45.5	41.6	45
20-25	45.6	44	50
25-30	52	52.4	54.8
30-35	61.4	61	65
35-40	32.4	33	34
40-45	28.6	30	30.8
45-50	57.4	58	62
50-55	60.4	64.2	65.4
55-60	34	34.2	35

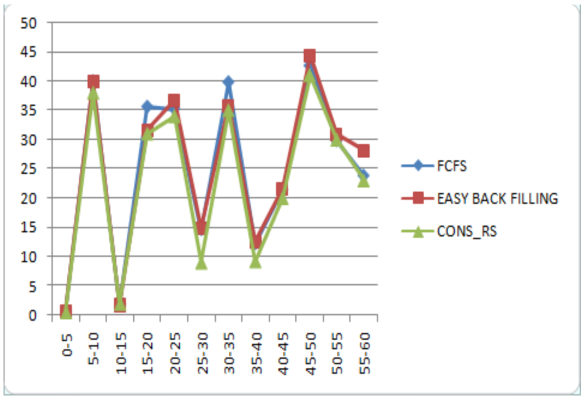

Fig: 3 Job Cluster Uses per Day (in Percentage) with FCFS, EASY Backfilling and Conservative Backfilling with Random Search Strategies.

As per above graph, proposed technique conservative backfilling with random search has more uses job clusters as compare than FCFS and EASY Backfilling method.

Table3: Number of Requested and Used CPU's

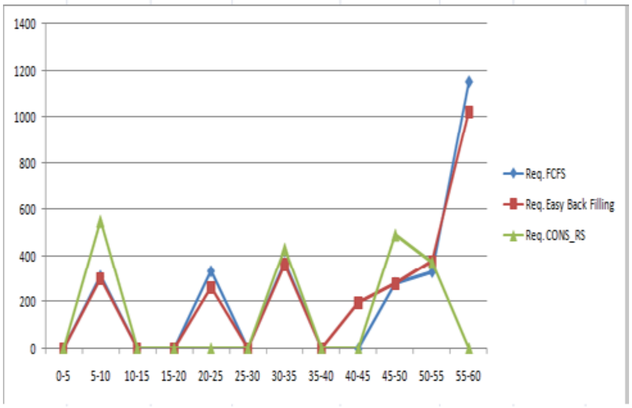

Fig: 4 Number of Requested CPU with FCFS, EASY Backfilling and Conservative Backfilling with Random Search Strategies

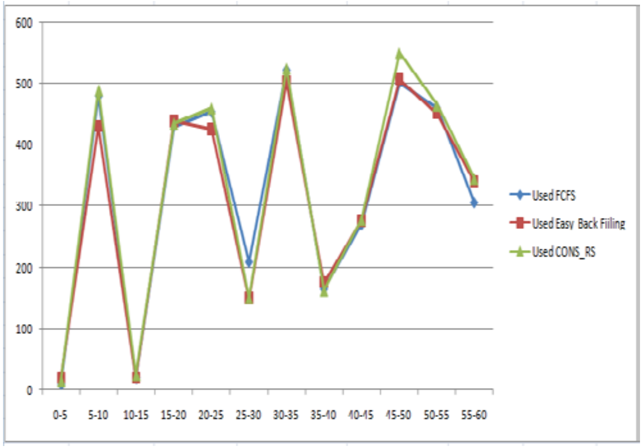

Fig: 5 Number of Used CPU with FCFS, EASY Backfilling and Conservative Backfilling with Random Search Strategies

Table4: Number of waiting and running job with different load balancing strategy.

DAYS	FCFS		EASY BACKFILLING		CNBF+RS (PROPOSED METHOD)	
	Waiting	Running	Waiting	Running	Waiting	Running
0-5	0	2	0	2	0	2
5-10	5	21	5	21	5	21
10-15	0	1	0	1	0	1
15-20	0	26	0	25	0	29
20-25	5	24	1	21	0	18
25-30	0	12	0	10	0	10
30-35	6	34	5	29	5	35
35-40	0	8	0	8	0	9
40-45	0	17	0	17	0	19
45-50	3	61	2	60	1	135
50-55	3	30	3	29	3	31
55-60	40	20	24	16	0	15

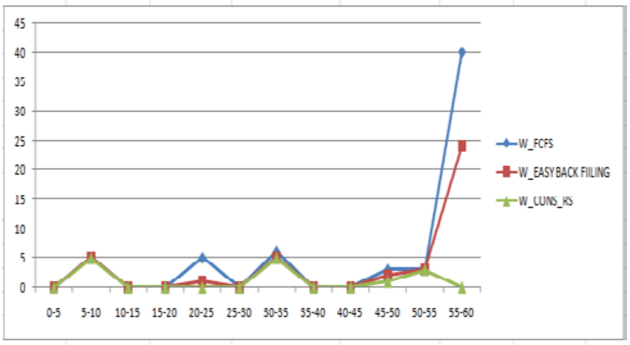

Fig: 6 Number of waiting jobs with different load balancing strategy.

6. CONCLUSIONS AND FUTURE WORK

The proposed CNBF-RS Algorithm implements load balancing for scheduling the jobs. Experiments have been done for makespan that serves as a parameter for evaluating the efficiency of the algorithm and finally average resource utilization that serves as the evaluation parameter for proper load balancing. From the results and discussion section it is observed that load balance threshold achieves a better result than the existing FCFS and EASY Backfilling algorithms. The proposed CNBF-RS algorithm considers the load of each resource at the time of scheduling which are very necessary in grid environment. This can be continued in future with factors for minimizing the communication overhead of the grid system.

We showed the design and implementation of a protocol (CNBF-RS) for load balancing with scheduling in a Computational Grid. The Grid is partitioned into a number of clusters and each cluster has a coordinator to perform local load balancing opinion and also to link with other cluster coordinators across the Grid to provide inter-cluster load transfers, if needed. Our results confirm that the load balancing method is scalable and has low message and time complexities. Our work is ongoing and we are looking into using the proposed model for real-time load balancing where scheduling of a process to a Grid node should be performed to meet its hard or soft deadline.

In future researches nodes can be designed hierarchically and different classes of sites can be considered for nodes (resources) in terms of computational capacity including low, medium and high classes and the efficiency of sites can be discussed based on them. In addition, for evaluating of effectiveness of a node, load of each site can be considered into value function, so that the best site for executing the task can be selected. Future work will focus on :

- Different scheduling can be used to optimization approaches.

- Quality of Service Constrains such as reliability, availability used as performance parameter.

Our research in this area is still at an early stage and there are many facets worthy for future study. Firstly, we have not modeled the contacts of accuracy of task execution estimation of time on the effectiveness of our proposed load balancing algorithm. Secondly, we will use migration threshold dynamically according to real-time observation of load behavior of system resources. Finally, we do not take network and hardware failures take into account of in this study. A failure model may be implemented to examine this influence. Matured to the dynamic nature of the practical grid environment, designing an ideal load balancing algorithm still remains a challenge.

7. REFERENCES

[1] B.Priya and Dr.T.Gnanasekaran, "Grid Architecture for Scheduling and Load Balancing – An Assessment", ICICES, 2014.

[2] Massimiliano Caramia and Stefano Giordani, “Resource Allocation In Grid Computing: An Economic Model”, Wseas Transactions On Computer Research, Page 19-27, Issue 1, Volume 3, January 2008.

[3]P. Keerthika and N. Kasthuri, “A Hybrid Scheduling Algorithm With Load Balancing For Computational Grid”, International Journal of Advanced Science And Technology, Vol.58, Pp.13-28, 2013.

[4] K. Sathish and A. Rama Mohan Reddy, “Maximizing Computational Profit in Grid Resource Allocation using Dynamic Algorithm”, Global Journal of Computer Science and Technology Cloud and Distributed, Volume 13, Issue 2, 2013.

[5] Leyli Mohammad Khanli and Behnaz Didevar, “A New Hybrid Load Balancing Algorithm in Grid Computing Systems”, International Journal of Computer Science Emerging Technology, Vol-2 No 5, Page 304-309, October, 2011.

[6] Resat Umit Payli, Kayhan Erciyes and Orhan Dagdeviren, “Cluster-Based Load Balancing Algorithms For Grids”, International Journal Of Computer Networks & Communications, Vol.3, No.5, Sep, Page 253-269, 2011

[7] Belabbas Yagoubi and Yahya Slimani, “Dynamic Load Balancing Strategy for Grid Computing”, International Journal of Computer, Electrical, Automation, Control and Information Engineering Vol. 2, No:7, 2008

[8] Ralf Diekmann, Andreas Frommer and Burkhard Monien, “Efficient schemes for nearest neighbor load balancing”, www.elsevier.com/locate/parco, 1999.

[9] U. Karthick Kumar, “A Dynamic Load Balancing Algorithm in Computational Grid Using Fair Scheduling”, International Journal of Computer Science Issues, Vol. 8, Issue 5, No 1, September 2011.

[10] Pawandeep Kaur and Harshpreet Singh, “Performance Analysis of Adaptive Dynamic Load Balancing in Grid Environment using GRIDSIM”, International Journal of Computer Science and Information Technologies, Vol. 3 (3), Page 4473-4479, Apr-2014.

[11] Kai Lu, Riky Subrata and Albert Y. Zomaya, “On the performance-driven load distribution for heterogeneous computational grids", www.elsevier.com/locate/jcss, Feb-2007.

[12] Priyanka Chauhan and Ritu Bansal, “Efficient Load Balancing and Resource Scheduling for Optimizing Cost and Execution Time Using ACO-A*Algorithm”, International Journal of Recent Research Aspects ISSN: 2349-7688, Vol. 1, Issue 2, pp. 189-196, September 2014.

[13] Frank C. H. Lin and Robert M. Keller, “The Gradient Model Load Balancing Method”, IEEE Transactions on Software Engineering, Vol. Se-13, No. 1, January 1987.

[14] Nikolaos D. Doulamis, Anastasios D. Doulamis, Emmanouel A. Varvarigos and Theodora A. Varvarigou, “Fair Scheduling Algorithms In Grids”, IEEE Transactions On Parallel And Distributed Systems, Vol. 18, No. 11, November 2007.

[15] Rajkumar Buyya and Manzur Murshed, “GridSim: A Toolkit for the Modeling and Simulation of Distributed Resource Management and Scheduling for Grid Computing”, http://www.buyya.com/gridsim/, 2011.