Improving Performance in Load Balancing Problem on the Grid Computing System

Prabhat Kr.Srivastava MCA Department IIMT College of Engineering Greater Noida, India Sonu Gupta MCA Department IIMT College of Engineering Greater Noida, India Dheerendra Singh Yadav MCA Department Dr. KNMIET Modinagar, India

ABSTRACT

Load Balancing is a technique to improve resources, utilizing parallelism, exploiting throughput managing and to reduce response time through proper distribution of the application. Load balancing strategies is always used for the emergence of Distributed systems. Generally there are three type of phases related to Load balancing i.e. Information Collection, Decision Making, Data Migration.

Grid computing is a replica of distributed computing that uses geographically and disperses resources. To increase performance and efficiency, the Grid system needs competent load balancing algorithms for the distribution of tasks. Load balancing algorithms is of two types, static and dynamic. Our projected algorithm is based on dynamic strategies.

Keywords

Information Gathering Policy, Firing Triggering Policy, Hitting Selection Policy

1. INTRODUCTION

Grid computing is a type of parallel and distributed system that enables the distribution, selection and aggregation of geologically resources dynamically at run time depending on their availability, capability, performance, cost, user quality-of -self-service requirement [1].Grid computing, individual users can retrieve computers and data, transparently, without taking into account the location, operating system, account administration, and other details. In Grid computing, the details are abstracted, and the resources are virtualized. Grid Computing should enable the job in question to be run on an idle machine elsewhere on the network [2]. Grids functionally bring together globally distributed computers and information systems for creating a universal source of computing power and information [3]. A key characteristic of Grids is that resources (e.g., CPU cycles and network capacities) are shared among various applications, and therefore, the amount

of resources available to any given application highly fluctuates over time. Load balancing is a technique to enhance resources, utilizing parallelism, exploiting throughput improvisation, and to reduce response time through an appropriate distribution of the application [4].

Load balancing algorithm are two type static and dynamic, Static load balancing algorithms allocate the tasks of a parallel program to workstations based on either the load at the time nodes are allocated to some task, or based on an average load of our workstation cluster. The decisions related to load balance are made at compile time[5].

A few static load balancing techniques are Round robin algorithm, Randomized algorithm, simulated annealing or genetic algorithms, and Dynamic load balancing algorithms make changes to the distribution of work among workstations at run-time; they use current or recent load information when making distribution decisions [6]. Multicomputers with dynamic load balancing allocate/reallocate resources at runtime based on no a priori task information, which may determine when and whose tasks can be migrated [7].

As a result, dynamic load balancing algorithms can provide a major improvement in performance over static algorithms. However, this comes at the additional cost of collecting and maintaining load information, so it is important to keep these overheads within reasonable limits [8].

There are three major parameters which usually define the strategy a specific load balancing algorithm will employ [9]. These three parameters answer three important questions:

Who makes the load balancing decision

What information is used to make the load balancing decision, and

Where the load balancing decision is made.

2. RELATED WORK

Various Load Balancing Algorithms are available now days but they contain several drawbacks. Such type of problems can be eradicated by our proposed dynamic load balancing algorithm [10].

Comparison of Existing and newly Proposed Algorithm are:

	Information Gathering Policy	Firing Triggering Policy	Hitting Selection Policy
Existing Load Balancing	Load Balancing information is composed using periodic approach	Load Balancing is triggered based on Queue Length	Task is selected for migration using Job Length as criteria.
Proposed Load Balancing	Load Balancing information is collected using Activity based approach	Load Balancer is triggered based on Queue Length and current CPU Load	Task is selected for migration based upon CPU consumption of tasks

In Condor (Existing) based algorithm, Information Policy may be fired by periodic approach while in Proposed algorithm it may be Activity based.

Triggering Policy in Existing Algorithm is based on Queue Length while in Proposed Algorithm it is based on Queue Length and current CPU Length. Selection Policy in Existing Algorithm is done by Selected Task may be migrated using Job Length while in Proposed Algorithm Selection Policy may be fired by Selected task which is migrated based upon CPU Utilization.

Major purpose of our proposed algorithm is Time Complexity i.e., in Proposed Algorithm time complexity is 3 while in Existing Algorithm time complexity is more than newly proposed algorithm. Execution time in .Net Framework using our proposed algorithm is too much fast compared to existing algorithms[11].

3. OUR PROPOSED ALGORITHM

In this paper we propose a dynamic load balancing algorithm for improving performance of grid computing. In this there are four basic steps: Monitoring workstation performance (load monitoring), exchanging this information between workstations (synchronization), Calculating new distributions and making the work movement decision (rebalancing criteria) Actual data movement (job migration). In proposed load balancing algorithm the activities can be categorized as following:

Arrival of any new job and queuing of that job to any particular node, Completion of execution of any job, Arrival of any new resource, Withdrawal of any existing resource.

Segment of code related to algorithm:-

Function: LoadBalancing_start Return Type: Boolean Start: If (CPU Idle of Node is Min and Free Memory of Node is Min and Queue Length of Node is Max) HeavilyLoaded_Node End if If (CPU Idle of Node is Max and Free Memory of Node is Max and Queue Length of Node is Min) Lightlyloded_Node End if Migrate Heavy Loded_Node_Job to Lightly Loded_Node End

Functions used in the above algorithm are:-

Activity_happens (): this function return Boolean value. If any of above defined activity occurs it returns true otherwise it returns false.

LoadBalancing_start (): this function also return Boolean value. If on the basis of given parameters (CPU utilization and queue length) load balancing will be required it will return true else it will return false. This function also updates two lists: HeavilyLoaded_list and LightlyLoaded_list.

4. IMPLEMENTATION AND RESULTS

A Load Balancing component has been developed which executes in simulated grid environment (i.e., Gridsim toolkit). This application has been developed using ASP.NET 3.5 and SQLServer 2005 database server. Following is the snapshot of the index page.



Screen Shot 4.1: Home Page

It contains link to different pages: show jobs, show resources, show allocation and show status. A click on one of the links will link to the different pages.

Jose JACK Jose JACK Kong Key Style Fly MF Pachet Konz Streatters Kong Key Style Fly MF Pachet Konz Streatters Kong Key Style Fly MF Pachet Konz Streatters Streat Xold Streat Kong		0.0		44 A 22		
Jose Part Home Nume Num Nume Nume	Back 🔹 💭 🛚 📓 🔏 🔑 Search 🏋 I	Favorites 🚷 🔂	🏐 🖻 • 🛄 🟭	11 13 - 13		
Description Annu () Show Add () <						💌 🄁 Go 🛛 Lir
Job Id Langth "The Kind 35 1223 But endes the stains, selection and aggregation of groupsheak distributed ensures distributed ensures distributed ensures distributed 36 1256 aggregation of groupsheak distributed ensures distributed ensures distributed ensures distributed ensures distributed ensures distributed 38 1367 on these available, containing, services and ensures distributed ensures distri	ALGORITH	M	tava. Hon	1e Show Job Sh	ow Resource Show Allocation Show Status	
55 1223 During the state if it is a state if it is	Lis	t Of Job Genera	ated By Grid Sim		The Grid	
56 1256 approximation <		Job Id	Length		"Grid is type of parallel and distributed system	
37 1259 avg/status/ status/status/ 38 1367 ensures dynamically status/status/ status/					that enables the sharing, selection and	
33 1367 on the available, or drafted and on the developed on the available, performance, on the available performance, performance, performance, performance, performance, performanc		36	1256		aggregation of geographically distributed	
39 1432 on the available, capability, patrimitate, capability, capability, patrimitate, 44 1633 on the available, capability, patrimitate, capability, capability, patrimitate, capability, patrimitate, statistics, patrimitate, 45 44 1634 statistics, patrimitate, statistics, patrimitate, statistics, patrimitate, 45 statistics, patrimitate, statistics, patrimitate, 45 45 1604 statistics, patrimitate, 47 statistics, patrimitate, 47 statistics, patrimitate, 48 46 1825 Characteritatics, of Grid statistics, patrimitate, 48 statistics, patrimitate, 48 50 2149 Evaluatity, 53 Statistics, patrimitate, 54 statistics, patrimitate, 55 statistics, patrimitate, 55 statistics, patrimitate, 55 statistics, patrimitate, 56 statistics, patrimitate, 57 statistics, patrimitate, 56 statistics, patrimitate, 57 statistics					resources dynamically at run time depending	
44 1563 col1, uster quality ofield stance 45 16/4 mquimmer' 46 1825 Characteristics Of Grid 47 1956 Characteristics Of Grid 49 218 Heimperaly 50 12469 Scalastity 51 2480 Scalastity 53 2142 Dynamckor Astastitipe 54 2873 Parallel CPU execution					on their availability, capability, performance,	
45 1694 requiriner* 46 1825 Characteristics Of Grid 47 1956 Characteristics Of Grid 48 2087 Heteropretiv 50 2249 Scalable 51 2480 Scalable 53 2561 Dynamick of Activities 54 2873 Parallel CPU exection					cost, user quality-of -self-service	
46 1835 47 1956 48 2087 49 2118 50 2149 51 2490 53 2742 53 2742 55 3004					requirement"	
47 1956 Characteristics Of Grid 48 2087 Heteroperative 40 218 Heteroperative 50 2349 Exatable 51 2480 Scatable 53 2142 Dynamotor Adaptable 53 2142 Dynamotor Adaptable 54 2873 Parallel CPU execution						
48 2087 Chaladematics Of Gind 49 218 Helerogenativ 50 2149 Scalability 51 2480 Scalability 53 25011 Dynamicity of Adaptability 54 2873 Parallel CPV execution						
40 2218 50 2349 51 2480 52 2611 53 2142 54 2033 55 3004					Characteristics Of Grid	
50 2149 Hetergenety 51 2480 Scatality 52 3511 53 2142 Dynamck or Alsobitive 54 2873 55 5044 Parallel CPV encoden						
51 2480 Scatality 52 3611 Scatality 53 1741 Drumick of Adaptabilitie 54 3873					Heterogeneity	
52 2611 53 2142 54 2873 55 3004						
53 7-42 Dynamicity or Adjustvillige 54 2873					Scalability	
54 2873 55 3004 Purelin CPU Reciden						
55 3004 Parallel CPU execution					Dynamicity of Adaptabilitye	
					Parallel CPU execution	
56 3135		56	3135			
813 Table Virbual organizations					Virtual organizations	

Screen Shot 4.2: Show Job Page

Screenshot 4.2 shows two columns: first column is JobID and second is job length. JobID is unique for each job submitted to Grid. Length is the expected job time length given at the time of submission.

0 · 🖹 🗟 🏠 /	🔎 Search 🤺 Favo	ntes 🚱 🕻	• 🎍 🖯 •	📃 🖗 🛍 🌘	18	
ttp://localhost:1060/Load%20Bala	ncing%20Algorithm/List)Resource.asp:				× .
	LOAD BALANG ALGORITHM Developed By Mr Pr		ivastava.	Home	Show Job S	How Resource Show Allocation Show Status
-	List Of R	esources	Generated B	iy Grid Sim		The Grid
IP Address	CPU Speed(Khz	CPU Idle(%	Total Memory	Free Memory (ueue Length	"Grid is type of parallel and distributed system
172.31.5.22	2200	90	1033832	1200	1	that enables the sharing, selection and
172.31.5.36	2100	89	1033832	2100	1	appreciation of geographically distributed
172.31.5.51	2200	78	1033832	700	1	resources dynamically at run time depending
172.31.5.127	2200	75	1033832	2400	1	
172.31.5.117	2100	80	1033832	3000	1	on their availability, capability, performance,
172.31.5.57	2200	65	1033832	34367	1	cost, user quality-of -self-service
172.31.5.147	2300	90	1033832	5841323	1	requirement"
172.31.5.27	2200	80	1033832	5933567	2	
172.31.5.97	2200	69	1033832	5727397	1	
172.31.5.120	2300	85	1033832	5356785	1	Characteristics Of Grid
						Heterogeneity
						Scalability
						Dynamicity or Adaptabilitye
						Parallel CPU execution
						Parallel CPU execution

Screen Shot 4.3: Show Resources Page

Screenshot 4.3 is the image of show resources page. This page shows six fields: IPAddress (resource name), CPUSpeed, CPUIdle, TotalMemory, FreeMemory and Queuelength. Three fields CPUIdle, FreeMemory and Queuelength are important because these fields are used to decide that particular resource is heavily loaded or lightly loaded resources and is shown in screen Shot 4.4. If any resource is heavily loaded then the actual load balancing starts.

lote * fields are mendatory		
	CPU Speed*	кнг
	Idle CPU *	%
	Amount Of Memory*	

Screen Shot 4.4: Input Field format

	🔎 Search 🤺 Favorites 🎸		æ~ <u>□</u>	
ittp://localhost:1060/Load%20		cation_aspx:		v 🔁 😡
*	LOAD BALANCING ALGORITHM Developed By Mr Prabhat Ko	ımar Srivast	312	iome Show Job Show Resource Show Allocation Show Status
-	Allocation Of Job Ger	nerated	By Grid Sim 1) Resource The Grid
	Job Id	Job Leng	th Resource	"Orid is type of parallel and distributed system
	35	1223	172.31.5.57	that enables the sharing, selection and
	44	1563	172.31.5.22	appregation of geographically distributed
	45	1694	172.31.5.147	resources dynamically at run time depending
	46	1825	172.31.5.147	on their availability, capability, performance,
	47	1956	172.31.5.36	cost, user qualit-of -self-service
	48	2087	172.31.5.51	requirement"
	49	2218	172.31.5.51	requirement.
	50	2349	172.31.5.127	
	51	248D	172.31.5.51	Characteristics Of Grid
	52	2611	172.31.5.22	
	53	2742	172.31.5.22	Heterogeneity
	54	2873	172.31.5.57	Heatingeneit
	55	3004	172.31.5.147	Scalability
	56	3135	172.31.5.27	
	57	3266	172.31.5.147	Dynamicity or Adaptabilitye
	58	3397	172.31.5.97	
			172.31.5.147	Parallel CPU execution
	59	3528	1/2.31.5.14/	

Screen Shot 4.5: Show Allocation Page

Screenshot 4.5 shows three fields JobId, JobLength, Resource. This is generated after allocation of job to resource. Allocation is based on free resource according to requirement of job that has been submitted to resource. This allocation maximizes the resource utilization. In this screenshot corresponding to each JobID, there is a resource name.

	ad%208alancing%204gorithe	(Status-aspic				🗸 🔁 Go 🛛 Lhi
	Status	Of Resources	Generated By G	rid Sim		
_	Status			id oin	The Grid	
-		List of Higly Lo	aded Resources		"Orid is type of parallel and distributed system	
	Resource	CPU Idle	Free Memory	Queue Length	that enables the sharing, selection and	
	172.31.5.22	60	1800	3	aggregation of geographically distributed	
					resources dynamically at run time depending	
_					on their availability, capability, performance,	
-		List Of Lightly L	oaded Resources		cost, user quality-of -self-service	
	Resource	CPU Idle	Free Memory	Queue Length	requirement"	
	172.31.5.36	89	2100	2		
	172.31.5.51	70	2000	2		
	172.31.5.127	75	2400	2	Characteristics Of Grid	
	172.31.5.117	80	3000	2		
	172.31.5.57	65 80	34367	2	Heterogeneity	
	1/2.31.5.14/	30 110	5553823 5634567	1		
	172.31.5.27	90	5427897	1	Scalability	
	172.31.5.120	85	5356785	1	Dynamicity or Adaptabilitye	
					Parallel CPU execution	
Per	form Load Balanci	ng click <u>mere</u>				
					Virtual organizations	
Cor	npare between ex	isting and prop	osed algorithms		Resource balancing	
Cor	npare with help of	f araph				
20.00	Contra Triteri Charge Ser	Carl Marcu			Reliability and Management	

Screen Shot 4.6: Show Status Page

Screenshot 4.6 shows information about gathered about Load Balancing. This page contains two Grids. First Grid contain name of those resource which is heavily loaded and second Grid contain those resources which are lightly loaded. These two Grids are used to decide which resource will act sender and which resource will act as receiver. Job will be migrated from sender (heavily loaded) resource to receiver (lightly loaded) resource if sufficient lightly loaded resources are available then after load balancing no heavily loaded resource will be available. Job from all heavily loaded resource will be migrated to lightly loaded resource. This page also gives information about which heavily loaded resource will migrating the Job to which lightly loaded resource. In above Screenshot resource 172.31.5.22 (name of resource) is heavily loaded resource which has three Jobs. This Page has a link button (i.e. perform load balancing) that performs the action to migrating the job from heavily loaded resource to lightly loaded resource, which is the main concept of our proposed algorithm and is demonstrated in our developed application.

- D - 🔰 🕯	🕽 🟠 🔎 Search 👷 I	avontes 🙆 🍰	· 🎍 🖯 - 🗖 🕻	0 🛍 🔒 🦓 👘	
	Load%20Balancing%20Algorithm				× 🖯
	LOAD BALA ALGORITHI Developed By M		estava.	Home Show Job	Show Resource Show Allocation Show Status
	Status (Of Resources (Generated By G	rid Sim	The Grid
					"Grid is type of parallel and distributed system
		List Of Lightly L	oaded Resources		that enables the sharing, selection and
	Resource	CPU Idle	Free Memory	Queue Length	aggregation of geographically distributed
	172.31.5.22	90	2300	2	resources dynamically at run time depending
	172.31.5.36	89	2100	2	on their availability, capability, performance,
	172.31.5.51	70	2000	2	cost, user quality-of -self-service
	172.31.5.127	75	2400	2	requirement*
	172.31.5.117	80	3000	2	
	172.31.5.57	65	34367	2	
	172.31.5.147	80	5553823	1	Characteristics Of Grid
	172.31.5.27	80	5634067	2	
	172.31.5.97	90	5427897	1	Heterogeneity
	172.31.5.120	85	5356785	1	
		or job has been mi	grated successfully !		Scalability
					Dynamicity or Adaptabilitye
	Home				Parallel CPU execution

Screen Shot 4.7: Load Balancing Page

Screenshot 4.7: Load Balancing Page appears after Load Balancing has been performed and job is migrated to resource 172.31.5.27. This is one particular case when heavily loaded resource has been finalize and job which should be migrated has been finalize and no heavily loaded resource is available If no lightly resource is available then no migration will be done.

5. CONCLUSION

Through this proposed algorithm, we have described multiple aspects of load balancing algorithm and introduced numerous concepts which illustrate its broad capabilities. Proposed algorithm is definitely a promising tendency to solve high demanding applications and all kinds of problems. Objective of the grid environment is to achieve high performance computing by optimal usage of geographically distributed and heterogeneous resources.

But grid application performance remains a challenge in dynamic grid environment. Resources can be submitted to Grid and can be withdrawn from Grid at any moment. This characteristic of Grid makes Load Balancing one of the critical features of Grid infrastructure. There are a number of factors, which can affect the grid application performance like load balancing, heterogeneity of resources and resource sharing in the Grid environment. In this we have focused on Load Balancing and tried to present the impacts of Load Balancing on grid application performance and finally proposed an efficient Load Balancing algorithm for Grid environment. Every Load Balancing algorithm implements five policies. The efficient implementation of these policies decides overall performance of Load Balancing algorithm. In this work we analyzed existing Load Balancing algorithm and proposed an enhanced algorithm which more efficiently implements three out of five policies implemented in existing Load Balancing algorithm. These three policies are: Information Policy, Triggering Policy and Selection Policy. Proposed algorithm is executed in simulated Grid environment. Load Balancing is one of most important features of Grid Middleware for efficient execution of compute intensive applications. The efficiency of Load Balancing Module overall decides the efficiency of Grid Middleware.

6. REFERENCES

- [1] Krishna rams Kenthapadi, Stanford University, kngk@cs.stanford.edu and Grummet Singh Mankuy, Google Inc., manku@google.com, Decentralized Algorithms using both Local and Random Probes for P2P Load Balancing.
- [2] B. Yagoubi , Department of Computer Science, Faculty of Sciences, University of Oran and Y. Slimani , Department of Computer Science, Faculty of Sciences of Tunis, Task Load Balancing Strategy for Grid Computing.
- [3] Rajkumar Buyya , Grid Computing and Distributed Systems (GRIDS) Lab., Department of Computer Science and Software Engineering, University of Melbourne, Australia and Manzur Murshed, Gippsland School of comp and IT, Monash University, Gippsland Campus , GridSim: a toolkit for the modeling and

simulation of distributed resource mgmt and scheduling for Grid computing.

- [4] Dazhang Gu, Lin Yang, Lonnie R. Welch ,Center for Intelligent, Distributed and Dependable Systems ,School of Electrical Engineering & Computer Science ,Ohio University, A Predictive, Decentralized Load Balancing Approach.
- [5] Akshay Luther, Rajkumar Buyya, Rajiv Ranjan, and Srikumar Venugopal, "Peer-to-Peer Grid Computing and a .NET-based Alchemi Framework", GRIDS Laboratory, The University of Melbourne, Australia.
- [6] Francois Grey, Matti Heikkurinen, Rosy Mondardini, Robindra Prabhu, "Brief History of Grid", http://Gridcafe.web.cern.ch/Gridcafe/Gridhistory/history. html
- [7] Rajkumar Buyya and S Venugopal, "A Gentle Introduction to Grid Computing and Technologies", http://www.buyya.com/papers/GridIntroCSI2005.pdf
- [8] Gregor von laszewaski, Ian Foster, Argonne National Laboratory, Designing Grid Based Problem solving Environments.
- [9] Junwei Cao1, Daniel P. Spooner, Stephen A. Jarvis, and Graham R. Nudd, Grid Load Balancing Using Intelligent Agents.
- [10] Jennifer M. Schopf, Mathematics and ComputerScience Division, Argonn National Lab, Department of Computer Science, Northwestern University, Grids: The Top Ten Questions.
- [11] Karl Czajkowski, Ian Foster and Carl Kesselman, Resource Co-Allocation in Computational Grids.
- [12] Ann Chervenak, Ian Foster, Carl Kesselman, Charles Salisbury and Steven Tuecke, The Data Grid: Towards an Architecture for the Distributed Management and Analysis of Large Scientific Data sets.
- [13] Klaus Krauter, Rajkumar Buyya, and Muthucumaru Maheswaran, a Taxonomy and Survey of Grid Resource Management Systems.
- [14] Arie Shoshani, Alex Sim and Junmin Gu, Lawrence Berkeley National laboratory, Storage Resource Managers: Middleware Components for Grid Storage.
- [15] Hai Zhuge, Xiaoping Sun, Jie Liu, Erlin Yao, and Xue Chen, A Scalable P2P Platform for the Knowledge Grid.