

Handling Complex Use Scenario in Grid Environment

By

Lata J Gadhavi

08MCE002



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
AHMEDABAD-382481**

May 2010

Handling Complex Use Scenario in Grid Environment

Major Project

Submitted in partial fulfillment of the requirements

For the degree of

Master of Technology in Computer Science and Engineering

By

Lata J Gadhavi

08MCE002



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
AHMEDABAD-382481

May 2010

Declaration

This is to certify that

- i) The thesis comprises my original work towards the degree of Master of Technology in Computer Science and Engineering at Nirma University and has not been submitted elsewhere for a degree.
- ii) Due acknowledgement has been made in the text to all other material used.

Lata J Gadhavi

Certificate

This is to certify that the Major Project entitled "Handling Complex Use Scenarios In Grid Environment" submitted by Lata J Gadhavi (08MCE002), towards the partial fulfillment of the requirements for the degree of Master of Technology in Computer Science and Engineering of Nirma University of Science and Technology, Ahmedabad is the record of work carried out by her under my supervision and guidance. In my opinion, the submitted work has reached a level required for being accepted for examination. The results embodied in this major project, to the best of my knowledge, haven't been submitted to any other university or institution for award of any degree or diploma.

Prof. Madhuri Bhavsar
Guide, Sr. Associate Professor,
Department of Computer Engineering,
Institute of Technology,
Nirma University, Ahmedabad

Dr. S.N. Pradhan
Professor, PG Coordinator,
Department of Computer Engineering,
Institute of Technology,
Nirma University, Ahmedabad

Prof. D.J. Patel
HOD, Professor,
Department of Computer Engineering,
Institute of Technology,
Nirma University, Ahmedabad

Dr. Ketan Kotecha
Director,
Institute of Technology,
Nirma University, Ahmedabad

Abstract

Computational Grids are a new trend in distributed computing systems. In the distributed environment every node has its own operating system, own resources, own processing speed, so the responsibility of communication between different platforms have been done by middleware. For this purpose, middleware takes the jobs from users and according to the job specification resource discovery has been done by middleware as a first step then assign the job to the particular resource, which has been discovered. So the middleware is responsible for the resource discovery, resource allocation and job will be executed successively in the grid environment. To reduce the load from the middleware & provide better facility to grid for job submission, selection of best resource from all the available resources & advance reservation of resources are to be handled. This thesis proposes the approach, which is Handling Complex Use Scenario in Grid Environment. In addition to processor utilization, it is important to consider the waiting time, throughput, and response times of jobs in evaluating the performance of grid scheduling strategies. To handle use scenarios for best resource selection & advance reservation of resources here two different algorithms has been proposed.

Acknowledgements

With immense pleasure, I would like to present the project report on "**Handling Complex Use Scenarios In Grid**". I am very much grateful to all the people who have helped me and provided guidance during the course of the project.

First of all, I would like to thank **Prof. Madhuri Bhavsar**, Institute of Technology, Nirma University, Ahmedabad, who persuaded in taking this particular project and constantly supported and motivated to complete the project. Her keen understanding and knowledge of the grid concepts helped in understanding the grid architecture and workflow, which made it easy for me to implement the same.

I would certainly like to thank **Dr. S.N.Pradhan**, PG-Coordinator, Computer Engineering Department, Institute of Technology, Nirma University, Ahmedabad for his constant encouragement and motivation throughout the course of the project.

I would like to thank **Dr. Ketan Kotecha**, Honorable Director, Nirma Institute of Technology, Nirma University for the facilities and environment for research.

Last, but not the least, no words are enough to acknowledge constant support and sacrifices of my husband **Mukesh**, my parents & my parents in-laws for their love, support and encouragement that they have given me throughout my life, helping me to persevere in my studies.

Finally, I am thankful to **Nirma University** for providing all the required resources that were needed for the project. I would also like to thank all those people who have directly or indirectly helped me in my project.

- **Lata J Gadhavi**

08MCE002

Contents

Declaration	iii
Certificate	iv
Abstract	v
Acknowledgements	vi
List of Tables	ix
List of Figures	x
Abbreviation	1
1 Introduction	2
1.1 Grid Application: An Example	3
1.2 Grid Architecture	4
1.3 Overview Of Project	5
1.4 Objective	6
1.5 Scope Of Work	6
1.6 Thesis Organization	7
2 Review of Literature	8
2.1 Methodology	9
2.2 Performance Analysis	10
3 Globus Configuration Preamble	14
3.1 Globus Configuration	14
3.1.1 Installing and Configuring Linux	15
3.1.2 Deploying Torque	15
3.1.3 Deploying Torque	15
3.1.4 Deploying The PostgreSQL Relational Database	15
3.1.5 Fixing Java And ANT	15
3.1.6 Deploying The Globus Toolkit 4.2.1	15

3.1.7	Connecting Globus Gram WS and Torque (Open PBS)	17
3.1.8	Connecting Globus Gram WS and SUN Grid Engine	17
4	Immediate job execution	18
4.1	Introduction	18
4.2	Process of Job Execution	19
4.3	Proposed Algorithm for Best Resource Selection	20
4.3.1	Introduction of the algorithm	20
4.3.2	Pseudocode of the algorithm	22
4.4	Implementation of Algorithm	24
4.4.1	Submission of Job	24
4.4.2	Requirement of resources for Job	25
4.4.3	Available Resources	26
4.4.4	Comparison between job & available resources's parameters	28
4.4.5	Comparison Results	30
4.4.6	Tool for Load Monitoring	31
5	Advance Reservation	33
5.1	Job execution that requires advance scheduling	33
5.2	Grid Simulation	34
5.2.1	GridSim	34
5.3	Advance Reservation Design	36
6	Installation & Implementation	40
6.1	Installation of Pre-requisites and Necessary Components	40
6.1.1	Installation of GridSim Toolkit	40
6.1.2	Salient Features of GridSim	41
6.1.3	Installation of Plus Manager Tool	42
6.2	Implementation	42
6.2.1	Steps for selection of job from multiple jobs	42
6.2.2	Advance Reservation of Resources for Job Execution	45
7	Conclusion and Future Scope	48
7.1	Conclusion	48
7.2	Future Scope	49
A	Troubleshooting	50
B	List of Paper Accepted/Published	53
	Website References	54
	References	54
	Index	56

List of Tables

I	Requirement of resources for Job	26
II	Available Resources	28
I	Calculation of job's average waiting time & response time	45
II	Analysis of Results	47

List of Figures

1.1	Weather prediction using Grid [4]	3
2.1	View of Grid Configuration[7]	10
4.1	Process Of Job Execution	19
4.2	Flow Graph for Best Resource Selection	23
4.3	Job Submission	25
4.4	Successful submission	26
4.5	Resources connected to Grid Environment	27
4.6	WEBMDS web front end	28
4.7	WEBMDS web front end for information of resources	29
4.8	WEBMDS web front end for information of resources	30
4.9	Comparison between demand of job's Kflops & Available Resource's Kflops	31
4.10	Ganglia Web Front End	32
5.1	Sequence Diagram for Job Submission	36
5.2	A state transition diagram for advance reservation	38
5.3	Flowchart for requesting a new reservation	39
6.1	Process of Whole Scenario	41
6.2	Submission of multiple jobs	43
6.3	Job selected using Combination of scheduling	44

Abbreviation

GRAM- Globus Resource Allocation Manager

MDS- Metacomputing Directory Service

GSI- Globus Security Infrastructure

HBM- Heartbeat monitor

GASS-Global Access to Secondary Storage

GARA-Globus Advance Reservation Architecture

GIIS-Grid Index Information Service

GRIS-Grid Resource Information Service

PBS- Portable Batch System

SGE- Sun Grid Engine

GRIR-Grid Resource Information Retrieving

GRIM-Grid Resource Information Monitoring

GridSim-Grid Simulator

Chapter 1

Introduction

In today's complex world of high speed computing, computers have become extremely powerful, even home-based desktops are powerful enough to run complex applications. But still we have numerous complex scientific experiments, advanced modeling scenarios, genome matching, astronomical research, a wide variety of simulations, complex scientific & business modeling scenarios and real-time personal portfolio management, which require huge amount of computational resources. To satisfy some of these aforementioned requirements, Grid Computing [1] is born. The Grid is emerging as a wide-scale, distributed computing infrastructure that promises to support resource sharing and coordinated problem solving in dynamic, multiinstitutional Virtual Organization [2].

Grid Computing is applying the resources of many computers in a network for a single problem at the same time - usually to a scientific or technical problem that requires a great number of computer processing cycles or access to large amounts of data. Grid Computing can be thought of as distributed and large-scale cluster computing and as a form of network-distributed parallel processing.

Grid Resources [3] fall into the categories of computation (i.e. a machine sharing its CPU), storage (i.e. a machine sharing its RAM or disk space), communication (i.e. sharing of bandwidth or a communication path), software and licenses and special equipment (i.e. sharing of devices).

1.1 Grid Application: An Example

A typical example of a Grid application is weather prediction. This involves collaboration between several partners: TV stations that produce regular weather news reports, a Satellite company that regularly provides space images of the earth, a super computing center that rapidly analysis the images and a visualization center that produces visual interpretations of the weather analysis (Figure 1.1). The smooth running of this project for the timely production of regular weather reports crucially depends on appropriate schemas for securely sharing, exchanging, and coordinating information between these partners.

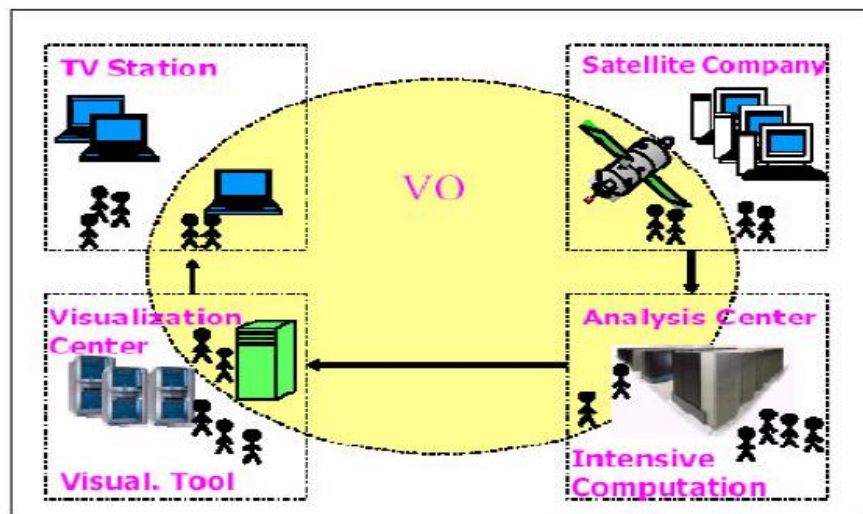


Figure 1.1: Weather prediction using Grid [4]

The power of Grid is particularly useful in areas involved in intensive processing such as life science research cite4, financial modeling ,industrial design and graphics rendering[4].

1.2 Grid Architecture

The architecture of the Grid is often described in terms of layers, each providing a specific function. In general, the higher layers are focused on the user, whereas the lower layers are more focused on computers and networks [5].

Fabric Layer: This layer represents all the physical infrastructure of the Grid, including computers and the communication networks. It is made up of the actual resources that are part of the Grid, such as computers, storage systems, electronic data catalogues and even sensors such as telescopes or other instruments, which can be connected directly to the network.

Resource and Connectivity Protocols: Resource and connectivity protocols handle all "Grid specific" network transactions between different computers and other resources on the Grid. Grids have to be able to recognize messages that are relevant to them, and filter out the rest. This is done with communication protocols, which let the resources speak to each other, enabling exchange of data, and authentication protocols, which provide secure mechanisms to verify the identity of both the users and the resources involved [5].

Collective Services: The collective services are also based on protocols: information protocols, which obtain information about the structure and state of the resources on the Grid, and management protocols, which negotiate access to resources in a uniform way. The services include:

- Keeping directories of available resources updated at all times.
- Brokering resources
- Monitoring and diagnosing problems on the Grid
- Replicating key data so that multiple copies are available at different locations
- Providing membership/policy services for keeping track on the Grid of who is allowed to do what and when.

As Grid computing matures, deciding which systems to use, where the data resides for a particular application domain, how to migrate the data to the point of computation (or vice versa), and data rates required to maintain a particular application behavior become significant.

Globus a middleware which is a de facto standard for grid computing. This project contains setup for a grid environment, in which 50s or 100s of execution nodes out of which one has been chosen as a submission node, and the other two are container nodes. These container nodes are actually having the schedulers of OpenPBS, SGE and also condor which creates heterogeneous environment for clusters. When the submission node submits jobs to the containers, then these jobs are scheduled to run on other nodes attached to the OpenPBS and SGE clusters in parallel.

1.3 Overview Of Project

Handling complex use scenario are ordered by increasing complexity, where complexity is loosely defined as the number of decisions that must be made by the grid software, or the scope of those decisions. Steps identified by the handling complex use scenario are,

- Immediate job execution
 - Run a job on a specified grid computer; local I/O & non-local file I/O required.
 - In a basic approach to remote job submission, the user specifies the execution host to be used and submits a job for which either the code already exists on the target machine or else is uploaded as part of the request.
 - Run a job on Best grid computer.
 - Instead of mandating the specific grid computer in the request, a user may run a job on the "best" computer defined to the host that is based on the comparison of available and required resources.
- Job execution that requires advance scheduling (Advance Reservation)
 - Make a reservation for the simultaneous resources.

-Claim the reservations.

1.4 Objective

Main objectives of project, are as follows-

- The first objective was to setup the grid environment , using the Globus Toolkit 4.2.1. Also setup and configured OpenPBS ,SGE ,and Condor schedulers on two of the grid nodes.
- Integration of globus with condor is done,then identified nodes are
 - guser01-nodeC.grid.nirma.com(Ip-10.1..3.13)(servernode)
 - guser02-nodeA.grid.nirma.com(Ip-10.1..3.14)(client node)
 - guser03-nodeB.grid.nirma.com(Ip-10.1..3.19)(PBS node)
- A client is a user who submit a job to the server node. A job refers the task which is to be carried out on computational grid. Then job submitted on scheduler and scheduler will divide a job into smaller tasks and sends that tasks to the available resources.
- To handle complex use scenarios on grid environment which includes the best resource selection & advance reservation of resource.

1.5 Scope Of Work

The grid environment setup includes a network of 25 of workstation established in a lab of the university. One of the special features of this project is study and configuration of OpenPBS , SGE clusters and condor, which can certainly be helpful in applications.

As computational grids can take advantage of the capabilities of computational grids in supporting their load balancing and quality-of-service requirements. Also designing and implementing steps for handling complex scenario provides Interactive, and on-demand access to grid resources.

1.6 Thesis Organization

The Project work is implemented in two major phases. It includes the all complex use scenario solution by the using grid computing. The initial phase contains grid environment has been setup, which was the fundamental task. Next phase is to design, identify resource design, implement and integrate resources and job Management in heterogeneous framework. Installation of Ganglia & Gridsim(Tool) for load monitoring & simulation of resources of grid environment respectively. The rest of the thesis is organized as follows:

Chapter 2,*Literature survey*

Chapter 3,*Globus Configuration Preamble* includes the prerequisites, installation and configuration of the Globus on submission and container nodes.

Chapter 4,*Immediate Job Execution* It includes the all description about the Best Resource Selection, proposed algorithm for best resource selection, process of job execution, implementation of algorithm.

Chapter 5,*Advance Reservation* It includes all description about advance reservation of resources for job execution, design of gridsim & proposed algorithm for advance reservation.

Chapter 6,*Installation & Implementation* It includes the installation of gridsim, plus manager tool for advance reservation. It also shows the implementation in detail for selection of job from multiple jobs using different algorithm & advance reservation steps for reserving the resources in advance.

Chapter 7,*Conclusion & Future Scope* includes conclusion and future work that can be done on the grid & which will be done in the future.

Chapter 2

Review of Literature

This chapter covers the brief explanation of the Grid Computing & it's all benefits & work which is relevant to the thesis work. It covers the all documentation of the grid resource management, Job execution & many things more. This is not just an overview of the Grid. This should be entirely reviewed for a reader familiar with the workings of Globus & Grid and the kernel role in the same.

- As discussed in the paper[6] Grid computing paradigm unites geographically distributed and heterogeneous computing, storage, and network resources and provide unified, secure, and pervasive access to the combined capabilities. Therefore, grid platforms enable sharing, exchange, discovery, selection, and aggregation of distributed heterogeneous resources such as computers, data-bases, visualization devices, and scientific instruments. Grid computing, therefore, leads to the creation of virtual organizations (VOs) by allowing geographically distributed communities to pool resources in order to achieve common objectives. Grid dynamical resource management system is to identify application requirements, matching grid resources for the application, dynamically allocating the

resources,scheduling the application onto the resources,and monitoring the application and resources as the application executes on them. Grid architecture is protocol architecture,with protocols defining the basic mechanisms by which VO users and resources negotiate,establish,manage,and exploit sharing relationships.Grid architecture organizes into layers.From the bottom of the stack and moving upward,these layers are fabric layer,connectivity layer,resource layer,collecrive layer,and application layer.

2.1 Methodology

Projects such as GridOS define a Grid OS to be an operating system with complementary support for Grid middleware. Projects such as Legion and WebOS are aimed at the development of a middleware infrastructure for Grid computing. In the paper [7] author refers to a Grid OS as a completely integrated operating system which integrates major Grid computing components into the machine operating system. It defines a Grid OS to be an operating system which transparently enables a user to peruse discovered distributed resources, to share his resources in a P2P fashion,to launch and to migrate tasks on them and to enable the control and monitoring of executed processes.Redundant grid configuration is shown in the figure.

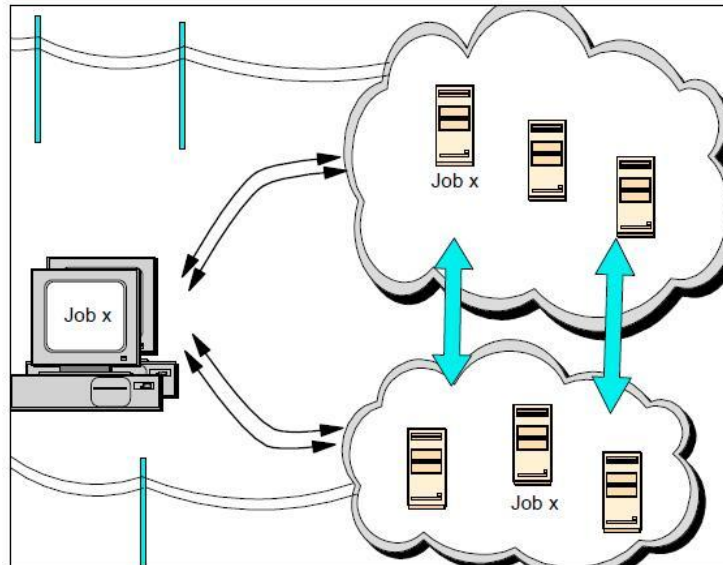


Figure 2.1: View of Grid Configuration[7]

2.2 Performance Analysis

- The task of a Grid resource broker and scheduler is to dynamically identify and characterize in this paper [8] the available resources, and to select and allocate the most appropriate resources for a given job. The resources are typically heterogeneous, locally administered, and accessible under different local policies. A decentralized broker, as the one considered here, operates without global control, and its decisions are entirely based on the information made available by individual resources and index servers providing lists of available resources and aggregated resource information.
- A workflow scheduler[9] is an application-level scheduler whose goal is to improve the application performance, i.e. to complete the workflow execution as fast as possible. But it is much harder to measure the performance of a workflow application than that of a single executable, because it involves both the concurrent and/or sequential execution of workflow tasks and the transfer of files between them. Workflow performance measurement in the next subsection

has discussed. A workflow scheduler in a grid is also a grid scheduler that works on top of multiple local schedulers, e.g SGE , LSF , and PBS . Author referred to this as the grid scheduling hierarchy and a grid scheduler as a grid metascheduler . The grid scheduling process makes resource allocation decisions involving resources over multiple administrative domains, which introduce several challenges for workflow scheduling.

- In the paper [10] author has described a large number of computing resources to provide a powerful capability to perform complex tasks or to share a large data set is a prevalent technology in inter-networking. Grid technology is one of the solutions for resource sharing. It combines the power of heterogeneous machines dispersed in different networks to increase the efficiency of processing tasks or accessing data. The Grid technique can speed up the job processing and reduce the access time for data acquisition. With grids, the dream to use the power of scattered resources will come true. Grid resources are inherently distributed over the wide-area network. They are independent and may be owned by a person, a group, or a public organization. Though shared, the owner has autonomy over the resources. To support various applications, these resources need to be integrated with corresponding Grid services. However, the fidelity of resource information is related to the efficient management of information gathering. To find and keep track of Grid resources, a resource monitoring mechanism is needed . By maintaining the resource status constantly, the necessary information can be quickly provided as requested. However, the cost of maintaining resource status is highly related to the number of resources and the frequency of status updating. Therefore, trade-off between maintenance cost and data accuracy should be considered. In this paper, focus on this resource monitoring trade-off and provide a novel information retrieving protocol, called the Grid Resource Information Retrieving (GRIR) protocol, to obtain the precise resource status. To get the latest information without unduly increasing the

maintenance load in Grids, an efficient Grid Resource Information Monitoring (GRIM) protocol is introduced.

- For complete support of Quality of Service[11], it is better that environment itself predicts resource requirements of a job by using special methods in the Grid computing. The exact and correct prediction causes exact matching of required resources with available resources. After the execution of each job, the used resources will be saved in the active database named "History". At first some of the attributes will be exploit from the main job and according to a defined similarity algorithm the most similar executed job will be exploited from "History" using statistic terms such as linear regression or average, resource requirements will be predicted.
- In the paper[12] author present an extension to devise and implement advance reservation as part of the scheduling and resource management services of the ASKALON Grid application development and runtime environment. The scheduling service has been enhanced to offer a list of resources that can execute a specific task and to negotiate with the resource manager about resources capable of processing tasks in the shortest possible time. Author has introduced progressive reservation approach which tries to allocate resources based on a fair-share principle. Experiments are shown that demonstrate the effectiveness of our approach, and that reflect different QoS parameters including performance, predictability, resource usage and resource fairness.

- In Grids, users may require assurance for completing their jobs on shared resources. Such guarantees can only be provided by reserving resources in advance. However, if many reservation requests arrive at a resource simultaneously, the overhead of providing such service due to adding, deleting, and searching, will be significant. An efficient data structure for managing these reservations plays an important role in order to minimize the time required for searching available resources, adding, and deleting reservations. In the paper[13], author has presented new approaches to advance reservation in order to deal with the limitations of the existing data structures, such as Segment Tree and Calendar Queue in similar problems. Here propose a Grid advanced reservation Queue (GarQ), which is a new data structure that improves some weaknesses of the aforementioned data structures.
- In this thesis work my attempts are made for handling complex use scenario ,which are immediate job execution ,run a job on specified grid computer,run a job on "Best' grid computer & advance reservation of resources.

Chapter 3

Globus Configuration Preamble

Globus is a grid middleware, considered to be the de facto standard for grid computing. So, for practical experience of the grid computing it is best to start with the installation of Globus.

3.1 Globus Configuration

- Globus is a grid middleware, considered to be the de facto standard for grid computing. So, for practical experience of the grid computing it is best to start with the installation of Globus[14].
- Operating System : Fedora 8.0
- Middleware : Globus 4.2.1
- Three main nodes for grid are-

guser01-nodeB.grid.nirma.com(Ip-10.1..3.13)(server node)

guser02-nodeA.grid.nirma.com(Ip-10.1..3.14)(client node)

guser03-nodeC.grid.nirma.com(Ip-10.1..3.19)(PBS node)

3.1.1 Installing and Configuring Linux

- Installing and Configuring Linux
- Creating accounts

3.1.2 Deploying Torque

Deploying Torque (OpenPBS) Torque Open PBS or just PBS is deployed on nodeB as a remote batch system, and then later configured Globus GRAM WS so that jobs can be submitted into PBS via Globus from the grid.

3.1.3 Deploying Torque

Deploying Sun Grid Engine (SGE) Sun Grid Engine (SGE) is deployed on nodeC as a remote batch system, and then later configured Globus GRAM WS so that jobs can be submitted into SGE via Globus from the grid.

3.1.4 Deploying The PostgreSQL Relational Database

It is required to run the Globus Reliable File Transfer (RFT) service on nodes B and C since they are the head nodes for our clusters. RFT requires a relational database backend in order to preserve state across machine shutdowns.

3.1.5 Fixing Java And ANT

The default java installed for Fedora Core 4.2.1 MUST be removed. It will not work with the Globus toolkit and it is easier and less troublesome to remove it entirely.

3.1.6 Deploying The Globus Toolkit 4.2.1

- Preliminaries

The Globus Toolkit should be installed as user 'globus' and not as root. The

toolkit should be deployed on

- nodeA since that node will serve as the 'client' machine
- nodeB since that node will host the Globus GRAM WS that front ends the PBS batch system, along with other Globus grid services.
- nodeC since that node will host the Globus GRAM WS that front ends the SGE batch system, along with other Globus grid services.

- Building and Installing
- Installing updated packages
- Creating a Certificate Authority
- Obtaining a Host Certificate on nodeB
- Making a Copy for the Container
- Creating the grid-mapfile
- Configuring the RFT Service
- Configuring sudo
- Starting the Container
- Starting globus-gridftp-server
- Repeat for nodeA
- Obtaining Credentials for Generic User
- Testing the Grid Services on nodeB
- Obtaining host credentials for nodeA
- Testing globus-gridftp-server on nodeA

- Testing file staging
- Completing Deployment on nodeC

3.1.7 Connecting Globus Gram WS and Torque (Open PBS)

Here now connect Globus GRAM WS so that jobs can be submitted into the PBS batch queue.

- Building the WS GRAM PBS jobmanager
- Testing the GRAM WS PBS jobmanager

3.1.8 Connecting Globus Gram WS and SUN Grid Engine

Here now connect Globus GRAM WS on nodeC to SGE so that jobs can be submitted into the SGE batch queue.

- Turning on reporting for SGE
- Building the WS GRAM SGE jobmanager
- Testing the GRAM WS SGE jobmanager

Chapter 4

Immediate job execution

4.1 Introduction

- Run a job on a specified grid computer:- In a basic approach to remote job submission, the user specifies the execution host to be used and submits a job for which either the code already exists on the target machine or else is uploaded as part of the request.
- Run a job on Best grid computer:- Instead of mandating the specific grid computer in the request, a user may run a job on the "best" computer defined to be the host that is based on the comparison of available and required resources. To select the Best Resource from all available resources here some implementation steps are done.
- Criteria for the Best Resource is applied that is Quickest & Cheapest resource, based on the previous history of the resource's performance, then with the comparison of job's parameter requirement for resource and available resources' parameter. Here Comparison is held & got the best resource from all available resources. Implementation for getting available resources is done using the Java Programming. First here described the algorithm for selection of best resource, then process of job execution & implementation steps of algorithm.

4.2 Process of Job Execution

- Computational grid which explores idle processing power is formed in the lab. This heterogeneous grid environment is configured by the interconnecting 25 nodes with GT 4.2.1 & condor 7.2.1.
- Head node is loaded with globus & condor, where jobs are submitted by the user. Other nodes are configured with PBS & SGE.
- Job will be submitted on head node and then it will run on the grid node which is selected for to run & it will show the result of the job execution. Flow of an algorithm as shown in fig 4.1, elaborates the process of job execution.

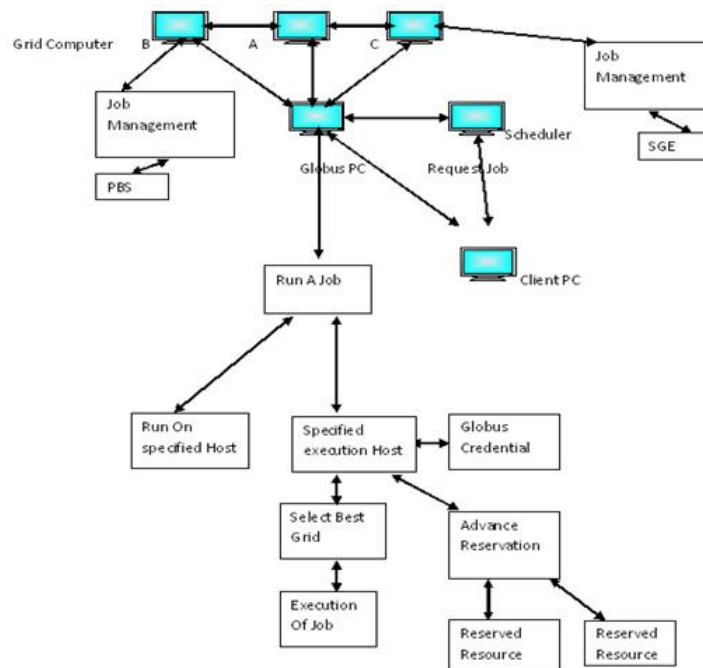


Figure 4.1: Process Of Job Execution

4.3 Proposed Algorithm for Best Resource Selection

4.3.1 Introduction of the algorithm

- Proposed brokering algorithm will perform a series of tasks, e.g., it processes the xRSL specifications in the job requests, discovers and characterizes the resources available, estimates the TTD for each resource of interest, makes advance reservation of resources, and performs the actual job submission. This algorithm presents a high-level outline of the tasks performed. The input xRSL specification(s) contain one or more job requests including information about the application to run (e.g., executable, arguments, input/output files), actual job requirements (e.g., amount of memory needed, architecture requirements, execution time required, million instructions per second (MIPS), Gflops), and optionally, job characteristics that can be used to improve the resource selection (e.g., listing of benchmarks with performance characteristics relevant for the application). The user input can also include a request for advance reservations which is covered in next step.
- Step 1, the user's request is processed and split into individual job requests.
- Step 2, the broker discovers what resources are available by contacting one or more index servers like Webmds. It is shown in the figure.
- Step 3, The specific characteristics of the resources found are identified through configuration & configuration of the resource list, by querying each individual resource. Each resource may provide static information about architecture type, memory configuration, CPU clock frequency, operating system, local scheduling system, clock rate, etc., and dynamic information about current load, batch queue status and various usage policies. Steps 2 and 3 are both performed

by LDAP queries sent from the broker to the index servers and the resources, respectively.

- step 4, The actual brokering process is mainly performed here, which is repeated for each job request.
- Step 5, resources are evaluated according to the requirements in the job request and only the appropriate resources are kept for further investigation.
- Step 6, predicts the performance of each resource by estimating the TTD, a step that may include the creation of advance reservations. Then, the currently considered job is submitted in the loop started at Step 7.
- Step 7, The loop is repeated until either the job is successfully submitted or all submission attempts fail, the latter causing the job to fail. After the configuration of the all available resources then it shows the information about the resources which are connected in the Grid. First of all the demand of resource will be get by using some configuration & by using the gprof & calculating execution time for the particular job, flops for that job means floating point instructions per second. Then it gives the kflops for that job. It will give the kflops of the program & execution time required for the job so there is comparison will held which is described in the next step.
- Step 8, When the requirements of the job will match with available resources' parameter then it will select that resource as a best resource and run a job on that resource, which is the Best resource.

Based on the previous history of the resources and based on performance measured by the configuration we can specify that which resource is best from all the available resources then run the job on that resource which fulfills the requirements of the job. Job properties is shown in the figure. By comparing the kflops of job & kflops of available resources here got that resource, which resources kflops

is best suitable with the demanding kflops of job requirements. After comparison it will select the Best resource from the all available resources .

- Step 9, the actual job submission & execution process is performed & it will show that this resource is "Best resource" so job will submitted by head node of grid and job will run on Best Resource & it will show execution result.

4.3.2 Pseudocode of the algorithm

- Algorithm -Job submission and selecting resource.

Input:-Job(J_i), $i=1,2,3\dots n$. (Different Jobs)

P_i , $i=1,2,3\dots n$ (Number of processor)

$S=S_1 \geq S_2 \geq S_3 \geq S_n$ (Different size of job)

Action: Job identifier(s) for the submitted job(s) & searching the best resources from the available resources.

Output: Run job on the best resource from all available resources based on the performance.

```

for(every  $J_i$  in  $J$ ) Do
    if( $R_q(J_i) > P$ ) then
        Select Best Resource
        Run
        Execute
    else if ( $R_q(J_i) > P$ ) then
        Run through Best fit algorithm
        execute
    else
        Run on all ordinary resources
        ( $P_{ji}, S_i$ ), ( $P_{ji}, S_i$ ), ( $P_{ji}, S_n$ )
    end if
end for

```


Process of best resource selection through algorithm is shown in the figure4.2.

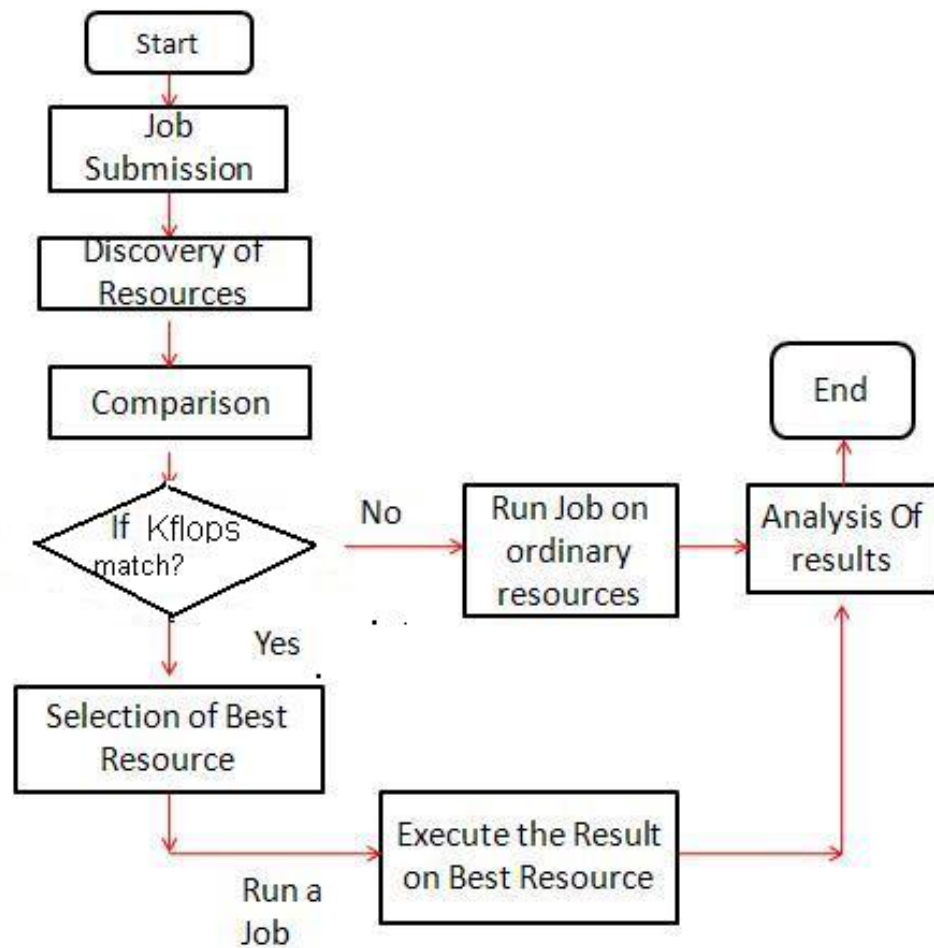


Figure 4.2: Flow Graph for Best Resource Selection

4.4 Implementation of Algorithm

implementation there is some requirements are required which is provided by the configuration with the Meta computing Directory Services & other functionality. Three main nodes of grid are-

- guser01-nodeC.grid.nirma.com(Ip-10.1..3.13)(server node)
- guser02-nodeA.grid.nirma.com(Ip-10.1..3.14)(client node)
- guser03-nodeB.grid.nirma.com(Ip-10.1..3.19)(PBS node)

A user will submit job on the head node of grid and selection of resources will done through the comparison. When the demanding of job specification will obtained then description of the available resources will found, which will show the all resource's description with the all performance parameters.

4.4.1 Submission of Job

First user will submit job on main node, which is shown in the figure 4.3. There is user have to enter the jobid & joblocation for the selected job, then user will submit a job on the main node of the grid, job will submitted successfully then that job will compare it's kflops with the available resources which are configured with the grid, by the best fit algorithm it will match Best suitable resource to run job on that resource. So selection of resource is implemented here. After the selection of the resource current job will run on that resource, which calls Best resource. After the submission of job by user it will show successful submission.



Figure 4.3: Job Submission

4.4.2 Requirement of resources for Job

- After the submission of job here criteria of job will compared with resources's parameters. When the requirements of the job will match with available resources then it will select that resource and run on that resource which is matched with required criteria then Best resource otherwise job will run on all ordinary resources. Based on the historical performance of resources and based on configuration detail identified by webmds, user can specify that which resource is best from all the resources and run the job on that resource which fulfills the requirements of the job. First i obtained the existing job requirements for running on the resource. It shows the parameters related job which covers job location, Job type, MIPS, kflops (floating point instructions per second), time taken for execution ,etc. It is shown in the table I. Steps for job requirements are,
 - (1) Availability of resources is checked against the required resources.
 - (2) Then algorithm selects the best resource to satisfy the need of job.
 Requirements of Job properties is shown in the table.

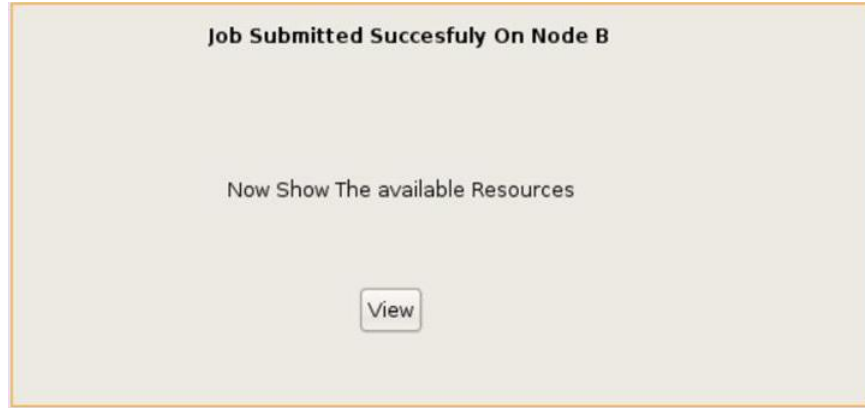


Figure 4.4: Successful submission

Table I: Requirement of resources for Job

No	Job Location	Job Type	Kflops	Time(sec)
1	D:/bc/bin	tsp.c	140	0.01
2	D:/bc /bin	mm1.c	60	0.03
3	D:/bc/bin	Performance.c	100	0.02

4.4.3 Available Resources

- From the WebMds getting the list of all available resources then it will give all information of the all available resources .

This available resources will give the information like machine name, Machine IP, Total condor Load Average, Total Load Average, MIPS of Machine, Kflops Of Machine & operationg System. Based on the available resources and requirement of resources for job there is Best resource will found.

Available Resources								
Name	Machine	Condor Load Ave...	Load Average	Operating System	KFlops	Mips	Total Load Avera...	Total Condor Lo...
slot1@A208cc-p...	A208cc-pc-1	0.100000	0.000000	WINNT5.1	1315958	5294	0.080000	0.000000
slot2@A208cc-p...	A208cc-pc-1	0.000000	0.080000	WINNT5.1	1315958	5222	0.080000	0.200000
slot1@mtech-1	mtech-1	0.100000	0.010000	WINNT5.1	1351737	4708	0.010000	0.100000
slot2@mtech-1	mtech-1	0.000000	0.000000	WINNT5.1	1351730	4624	0.010000	0.100000
slot2@A104-CC...	A104-CC-PC-61	0.000000	1.000000	WINNT5.1	627666	2025	1.110000	0.000000
slot1@A104-CC...	A104-CC-PC-62	0.000000	0.000000	WINNT5.1	685981	2140	0.010000	0.000000
slot2@A104-CC...	A104-CC-PC-62	0.000000	0.010000	WINNT5.1	685981	2140	0.010000	0.000000
slot1@A104cc-u...	A104cc-user51	0.000000	0.010000	WINNT5.1	646275	2140	0.010000	0.000000
slot2@A104cc-u...	A104cc-user51	0.000000	0.000000	WINNT5.1	646275	2140	0.010000	0.000000
slot1@A104cc-u...	A104cc-user52	0.000000	0.010000	WINNT5.1	606061	2140	0.010000	0.000000
slot2@A104cc-u...	A104cc-user52	0.000000	0.000000	WINNT5.1	606061	2140	0.010000	0.000000
slot1@A104cc-u...	A104cc-user53	0.000000	0.000000	WINNT5.1	640547	1916	0.010000	0.000000
slot2@A104cc-u...	A104cc-user53	0.000000	0.010000	WINNT5.1	640547	1916	0.010000	0.000000
slot1@A104cc-u...	A104cc-user54	0.000000	0.000000	WINNT5.1	618062	2025	0.000000	0.000000
slot2@A104cc-u...	A104cc-user54	0.000000	0.000000	WINNT5.1	618062	2025	0.000000	0.000000
slot1@A104cc-u...	A104cc-user55	0.000000	0.000000	WINNT5.1	646275	2025	0.000000	0.000000
slot2@A104cc-u...	A104cc-user55	0.000000	0.000000	WINNT5.1	646275	2025	0.000000	0.000000
slot1@A104cc-u...	A104cc-user56	0.000000	0.000000	WINNT5.1	660892	2025	0.010000	0.000000
slot2@A104cc-u...	A104cc-user56	0.000000	0.010000	WINNT5.1	660892	2025	0.010000	0.000000
slot1@A104cc-u...	A104cc-user57	0.000000	0.000000	WINNT5.1	660892	2025	0.000000	0.000000
slot2@A104cc-u...	A104cc-user57	0.000000	0.000000	WINNT5.1	660892	2025	0.000000	0.000000
slot1@A104cc-u...	A104cc-user59	0.000000	0.010000	WINNT5.1	670900	2025	0.010000	0.000000
slot2@A104cc-u...	A104cc-user59	0.000000	0.000000	WINNT5.1	670900	2025	0.010000	0.000000
slot1@A104cc-u...	A104cc-user60	0.000000	0.000000	WINNT5.1	447631	1402	0.000000	0.000000
slot2@A104cc-u...	A104cc-user60	0.000000	0.000000	WINNT5.1	447631	1402	0.000000	0.000000
slot1@A104cc-u...	A104cc-user64	0.000000	0.000000	WINNT5.1	738351	2599	0.000000	0.000000
slot2@A104cc-u...	A104cc-user64	0.000000	0.000000	WINNT5.1	738351	2599	0.000000	0.000000
slot1@A104cc-u...	A104cc-user66	0.000000	0.000000	WINNT5.1	657097	2025	0.000000	0.000000
slot2@A104cc-u...	A104cc-user66	0.000000	0.000000	WINNT5.1	657097	2025	0.000000	0.000000
slot1@A104cc-u...	A104cc-user67	0.000000	0.000000	WINNT5.1	665052	2025	0.000000	0.000000
slot2@A104cc-u...	A104cc-user67	0.000000	0.000000	WINNT5.1	665052	2025	0.000000	0.000000
slot1@A104cc-u...	A104cc-user69	0.000000	0.000000	WINNT5.1	632290	2025	0.000000	0.000000
slot2@A104cc-u...	A104cc-user69	0.000000	0.000000	WINNT5.1	632290	2025	0.000000	0.000000
slot1@A104cc-u...	A104cc-user70	0.000000	0.000000	WINNT5.1	637278	2140	0.000000	0.000000
slot2@A104cc-u...	A104cc-user70	0.000000	0.000000	WINNT5.1	637278	2140	0.000000	0.000000
slot1@A104cc-u...	A104cc-user71	0.000000	0.000000	WINNT5.1	623959	2140	0.000000	0.000000
slot2@A104cc-u...	A104cc-user71	0.000000	0.000000	WINNT5.1	623959	2140	0.000000	0.000000
slot1@A104cc-u...	A104cc-user72	0.000000	0.000000	WINNT5.1	642345	2025	0.000000	0.000000
slot2@A104cc-u...	A104cc-user72	0.000000	0.000000	WINNT5.1	642345	2025	0.000000	0.000000

Figure 4.5: Resources connected to Grid Environment

When webmds run on the localhost it shows the all information of the connected grid resources like address of the resource,GRAM version,host name,total CPUs,etc. This figure shows the other information of the resources like total number of bytes transferred.

Table II: Available Resources

Machine Name	Load average	OS	Kflops	MIPS	Total Load Avg
Slot1@A208cc-pc-1	1.00000	WINNT51	1307936	4421	1.100000
Slot1@A208cc-pc-19	0.000000	WINNT51	1349050	5222	0.000000
Grid2.nitdomain	0.000000	WINNT51	586644	1655	0.000000
Slot1@mtech-1	0.120000	WINNT51	1386172	2655	0.120000
	1.12		4588688	13953	1.22

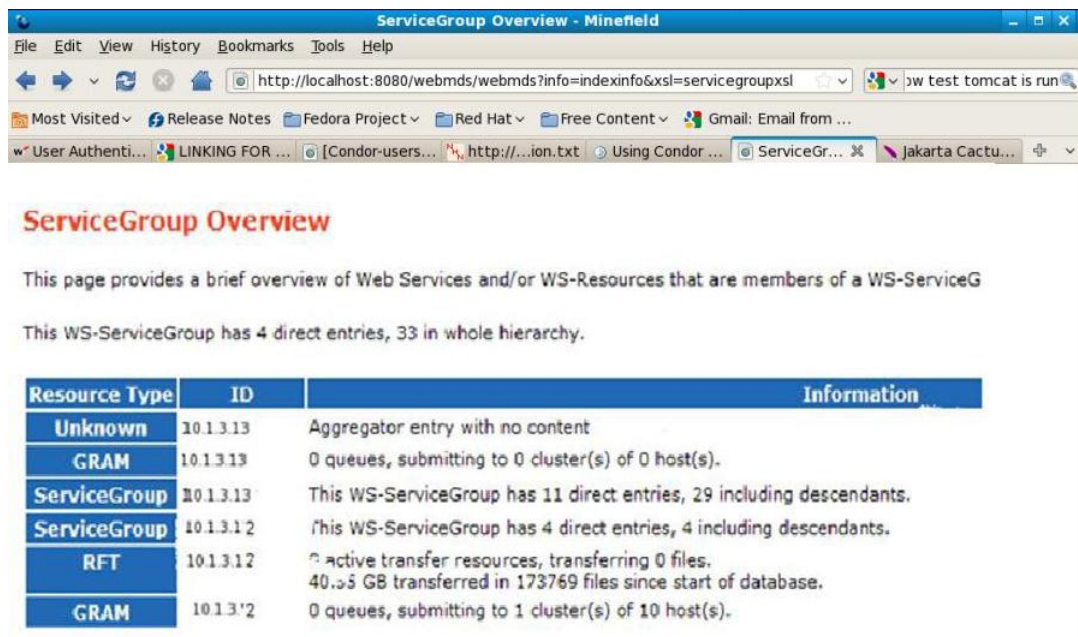
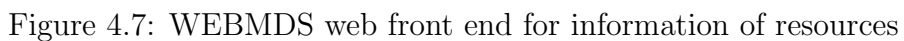


Figure 4.6: WEBMDS web front end

4.4.4 Comparison between job & available resources's parameters

- Here comparison is held based on the kflops criteria. Programs kflops & Available resources kflops are compared. By comparing the kflops got the results for Best resource, which resources kflops is best suitable with the demanding kflops so by using the programming on the grid it will select the Best resource from the all available resources.

As shown in the figure Best resource is selected amongst all the resources. After



the selection of the Best resource job will only run on that resource, job will execute on that resource & it will give better result than the ordinary resources. If job will run on Best resource then there are execution time to execute job will be taken less comparatively other resources because Best Resource is Quickest & faster than other resources in grid environment.

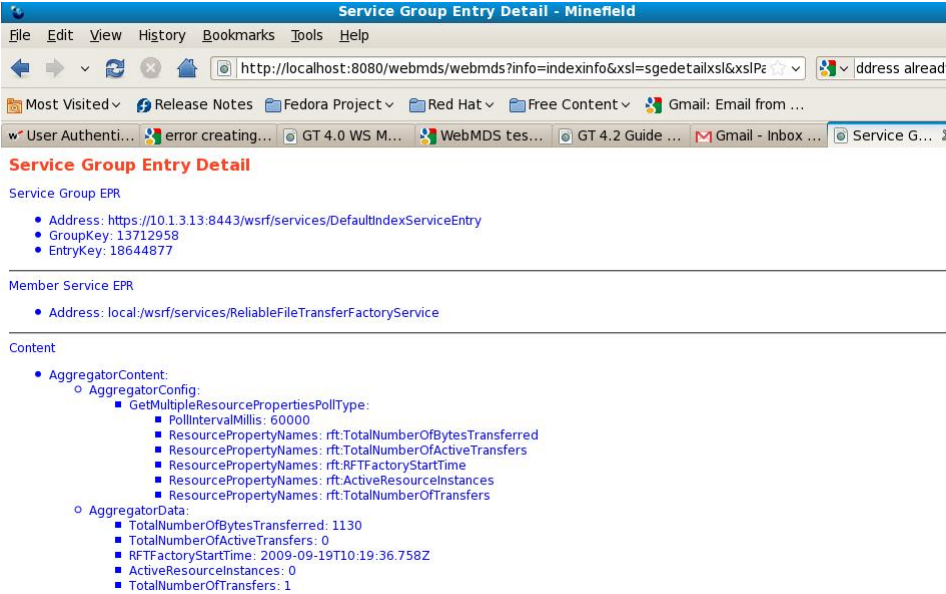


Figure 4.8: WEBMDS web front end for information of resources

4.4.5 Comparison Results

In this table it shows the all resources which are available and kflops of the resources & selected resource for job submission. It will find the which resource's kflops match with the job's requirements, which calls kflops of job and other functionality parameters related with the job, which is submitted on the main node.

Machine Name	Load average	OS	Kflops	MIPS	Total Load Avg
Slot1@A208cc-pc-1	1.00000	WINNT51	1307936	4421	1.100000
Slot1@A208cc-pc-19	0.000000	WINNT51	1349050	5222	0.000000
Grid2.nitdomain	0.000000	WINNT51	586644	1655	0.000000
Slot1@rntech-1	0.120000	WINNT51	1386172	2655	0.120000
	1.12		4588688	13953	1.22

Figure 4.9: Comparison between demand of job's Kflops & Available Resource's Kflops

4.4.6 Tool for Load Monitoring

Ganglia is a scalable distributed monitoring system for high performance computing systems such as clusters and Grids. Ganglia uses RRDtool (Round Robin Database) to store and visualize historical monitoring information for grid, cluster, host, and metric trends over different time granularities ranging from minutes to years[15]. It shows the Grid resources's load ,cpu ,network & memory last hour. It is shown in the figure4.8.



Figure 4.10: Ganglia Web Front End

Chapter 5

Advance Reservation

5.1 Job execution that requires advance scheduling

- Make a reservation for simultaneous resources

-Grid computing has emerged as the next-generation parallel and distributed computing that aggregates dispersed heterogeneous resources for solving a range of large-scale parallel applications in science, engineering and commerce . In most Grid scheduling systems, submitted jobs are initially placed into a queue if there are no available resources. Therefore, there is no guarantee as to when these jobs will be executed. This causes problems in parallel applications, where most of them have dependencies among each other.

-Advance Reservation (AR) is a process of requesting resources for use at a specific time in the future. Common resources whose usage can be reserved or requested are CPUs, memory, disk space and network bandwidth. AR for a grid resource solves the above problem by allowing users to gain concurrent access to adequate resources for applications to be executed. AR also guarantees the availability of resources to users and applications at the required times.

-There are some systems that support AR capability, such as Globus Architecture for Reservation and Allocation (GARA) and Maui Scheduler.

5.2 Grid Simulation

- Simulation is the imitation of some real thing, state of affairs, or process. The act of simulating something generally entails representing certain key characteristics or behaviors of a selected physical or abstract system. Grid environment also can be simulated using several Grid simulators e.g. GridSim [16], Eclipse [17] etc. Grid simulators enable Grid users to work on Grid like environment without having to worry about the other external factors that may influence the Grid environment.

5.2.1 GridSim

- some tools are available for application scheduling simulation in the Grid computing environment, such as Bricks , SimGrid and OptorSim . However, none of them have the capability of simulating reservation-based systems. To address this weakness, we extend GridSim to support AR mechanisms.
- GridSim[16] is a software platform that enables users to model and simulate the characteristics of Grid resources and networks with different configurations. Study Grids, or test new algorithms and strategies in a controlled environment. By using GridSim, they are able to perform repeatable experiments and studies that are not possible in a real dynamic Grid environment. GridSim is a Java-based grid simulation package that provides features for application com-

position, information services for resource discovery, and interfaces for assigning applications. GridSim also has the ability to model heterogeneous computational resources of variable performance.

- In this work, GridSim has been extended with the ability to handle:
 - (1) creation or request of a new reservation for one or more CPUs;
 - (2) commitment of a newly created reservation;
 - (3) activation of a reservation once the current simulation time is the start time;
 - (4) modification of an existing reservation; and
 - (5) cancelation and query of an existing reservation.

5.3 Advance Reservation Design

- Sequence Diagram for Job Submission

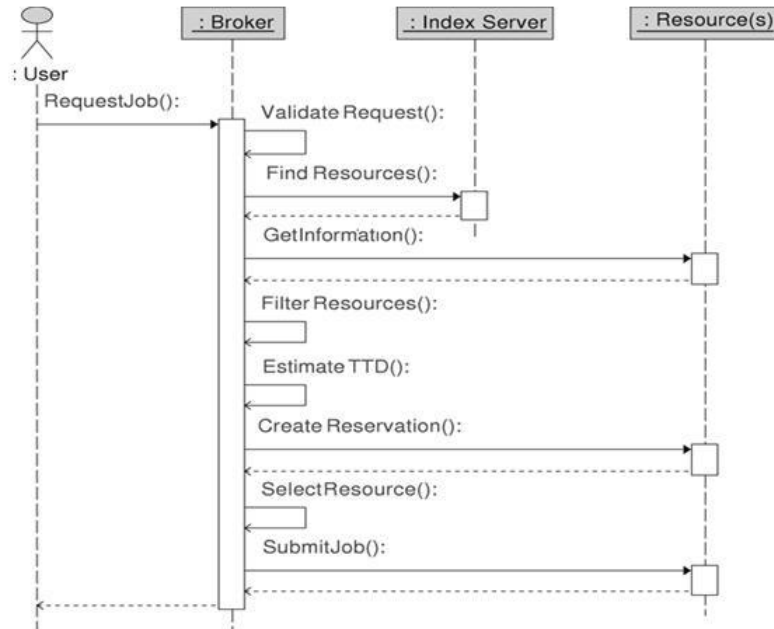


Figure 5.1: Sequence Diagram for Job Submission

- Introduction of the Sequence Diagram

- In step 1,the user's request is processed and split into individual job requests.
- In Step 2, the user discovers what resources are available by contacting one or more index servers like webmds.
- The specific characteristics of the resources found are identified in Step 3, by querying each individual resource. Each resource may provide static information about architecture type, memory configuration, CPU clock frequency, operating system, local scheduling system, etc., and dynamic information about current load, batch queue status and various usage policies.

- The actual brokering process is mainly performed in Step 4, which is repeated for each job request.
 - In Step 5, resources are evaluated according to the requirements in the job request and only the appropriate resources are kept for further investigation.
 - Step 6 predicts the performance of each resource by estimating the TTD, a step that may include the creation of advance reservations.
 - Then, the currently considered job is submitted in the loop started at Step 7.
 - The loop is repeated until either the job is successfully submitted or all submission attempts fail, the latter causing the job to fail.
 - In last step, the submission of one job is completed, any non-utilized reservations are released, which shows the advance reservation of the resources.
- For the advance reservation there are some requirements, like Rid (node id of the resource) for reserving the particular resource bases on the Rid or performance parameters. Which is shown in the implementation part.

- **States of Advance Reservation.** Transitions between the states are defined by the operations that a user performs on the reservation. These states are defined as follows. A reservation can be one of several states during its lifetime as shown in the figure 1.
- Requested: Initial state of the reservation, when a request for a reservation is first made.
- Rejected: The reservation is not successfully allocated due to full slots, or an existing reservation has expired.
- Accepted: A request for a new reservation has been approved. Committed: A reservation has been confirmed by a user before the expiry time, and will be honored by a resource. Change Requested: A user is trying to alter the requirements for the reservation prior to its starting. If it is successful, then the reservation is committed with the new requirements, otherwise the values

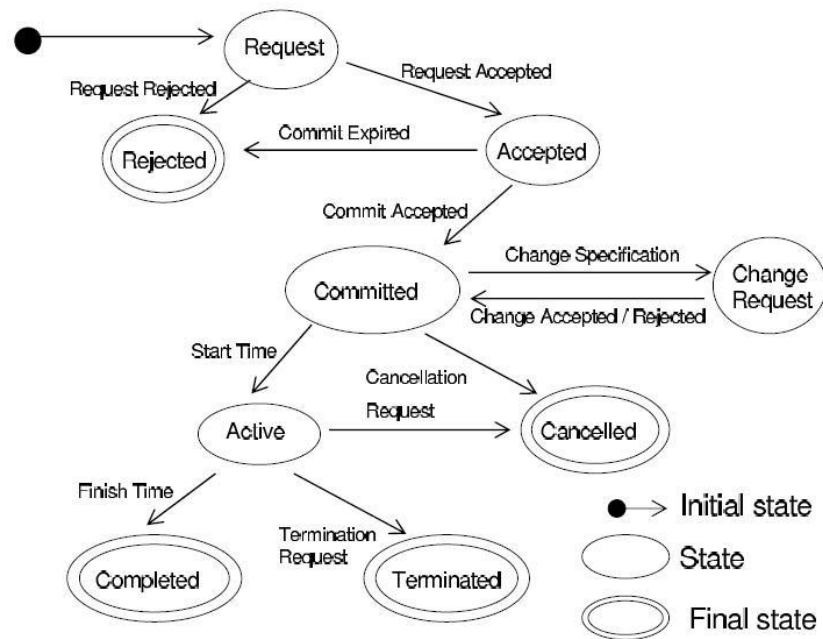


Figure 5.2: A state transition diagram for advance reservation

remain the same.

- Active: The reservations start time has been reached. The resource now executes the reservation.
- Cancelled: A user no longer requires a reservation and requests that it is to be cancelled.
- Completed: The reservations end time has been reached.
- Terminated: A user terminates an active reservation before the end time.

Process of creating reservation of resources are shown in the flowchart.

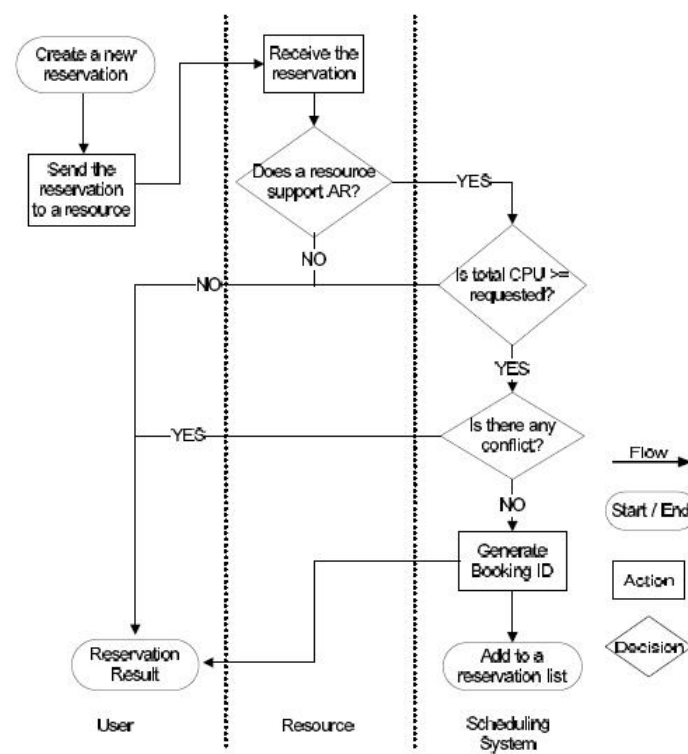


Figure 5.3: Flowchart for requesting a new reservation

Chapter 6

Installation & Implementation

A user goes through three stages to schedule a job when it involves multiple sites. Phase one is Resource Discovery, in which the user makes a list of potential resources to use. Phase two involves gathering information about those resources and choosing a best set to use. In phase three the user runs the job. It is shown in the figure. The Priority Based Queue Algorithm & Advance reservation algorithm has been implemented in JAVA programming language on top of GridSim [16] Toolkit 4.0. The steps followed for the implementation are described below in detail with appropriate screenshots.

6.1 Installation of Pre-requisites and Necessary Components

6.1.1 Installation of GridSim Toolkit

In the implementation of the Priority Based Queue & Advance Reservation algorithm I have used GridSim Toolkit version 4.0. This open source JAVA based toolkit can be downloaded from the homepage of Grid-Bus Society.

The GridSim [16] toolkit provides a comprehensive facility for simulation of different classes of heterogeneous resources, users, applications, resource brokers, and sched-

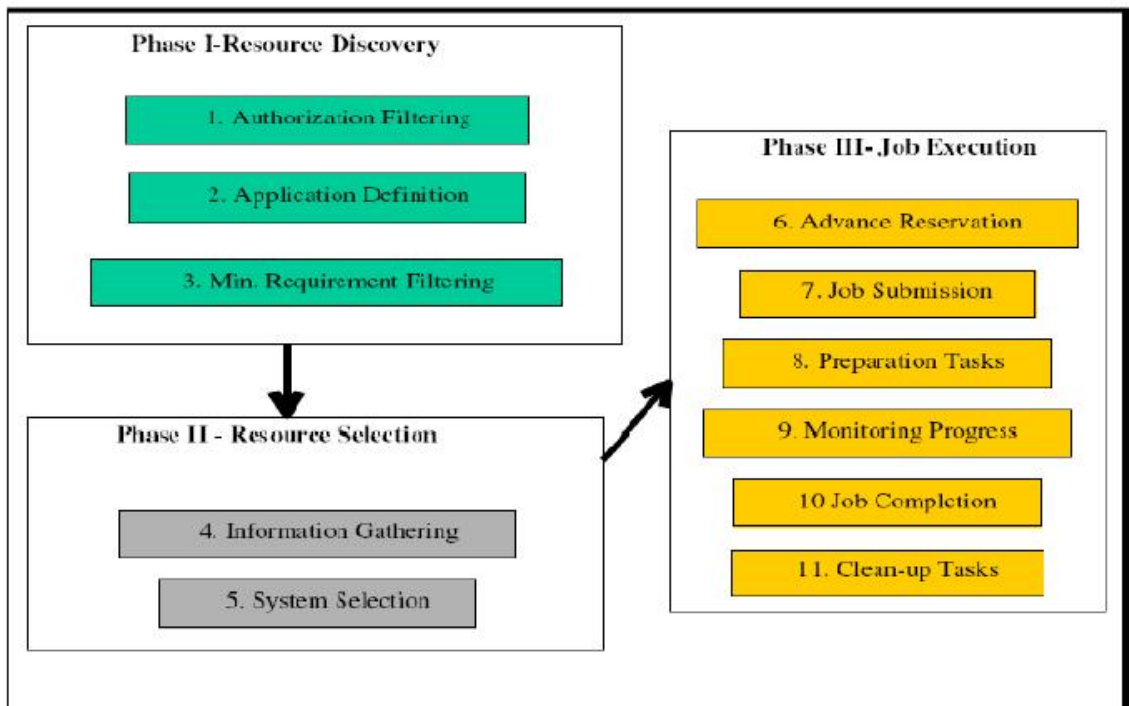


Figure 6.1: Process of Whole Scenario

ulers. It can be used to simulate application schedulers for single or multiple administrative domains distributed computing systems such as clusters and Grids.

6.1.2 Salient Features of GridSim

It allows modeling of heterogeneous types of resources.

- Resources can be modeled operating under space-or time-shared mode.
- Resource capability can be specified in the form of MIPS (Million Instructions per Second) or as per SPEC (Standard Performance Evaluation Corporation).
- Resources can be located in any time zone.
- Weekends and holidays can be mapped depending on resources local time to model non-Grid (local) workload.

- Resources can be booked for advance reservation.
- Applications with different parallel application models can be simulated.
- Application tasks can be heterogeneous and they can be CPU or I/O intensive.
- No limit on the number of application jobs that can be submitted to a resource.
- Multiple user entities can submit tasks for execution simultaneously in the same resource, which may be time-shared or space-shared. Network speed between resources can be specified.
- It supports simulation of both static and dynamic schedulers.
- Statistics of all or selected operations can be recorded and they can be analyzed using GridSim statistics analysis methods.

6.1.3 Installation of Plus Manager Tool

In the implementation of advance reservation algorithm I have used Plus manager tool for reserving the resources in advance & gridsim for simulation.

PluS is a reservation manager that can work with TORQUE (PBS) and Grid Engine. It provides Advance Reservation capability, which is essential to perform co-allocation with other resources on the Grid. I have installed Plus Manager Tool version(1.1.0) which is downloaded from the homepage of g-lambda Plus[18] with TORQUE(PBS).

6.2 Implementation

6.2.1 Steps for selection of job from multiple jobs

The jobs submitted to the Grid through a job submission service such as Globus WS-GRAM[19], must be validated in order to check if the requested resources were

actually reserved by the user. When there are many jobs are placed into queue for submitting on the middleware there is scheduling algorithm are used for selecting the sequence of multiple jobs to submit & execute, then one job will selected for submission on the main node ,now here from the all available resources if user wants to reserve any node in advance then he/she can do. Providing this facility to user here Scheduling & Advance Reservation algorithm are used.

Implementation steps are described below. In the starting window the user is prompted to enter the values for the number of Jobs needed to be simulated. For the simulation here gridsim is used for selecting the job from multiple jobs. Here jobs are simulated using the different algorithm like first come first serve, shortest job first & round robin algorithm. Here for scheduling the jobs one combination is taken. GridSim create the jobs and assigned to the respective priority queue. There are three queues with three different scheduling algorithms.

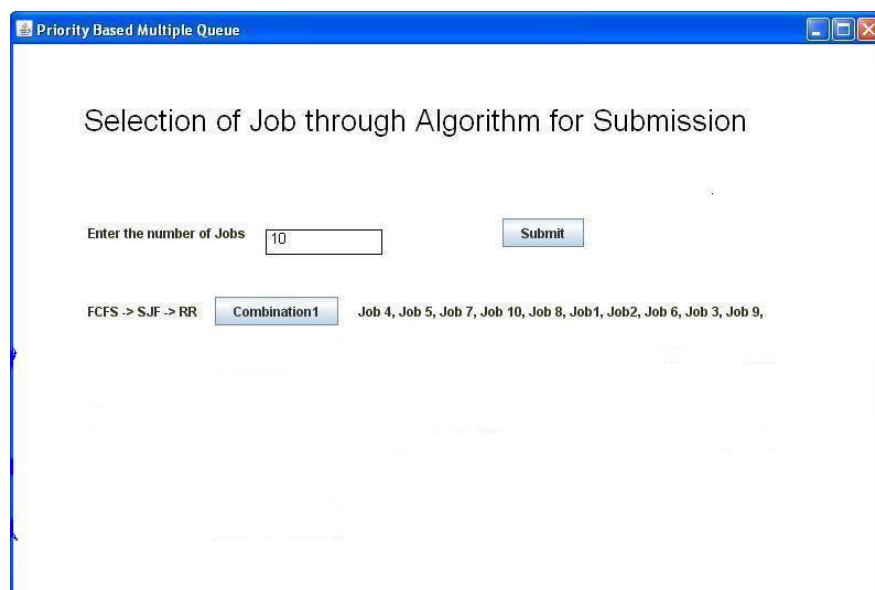
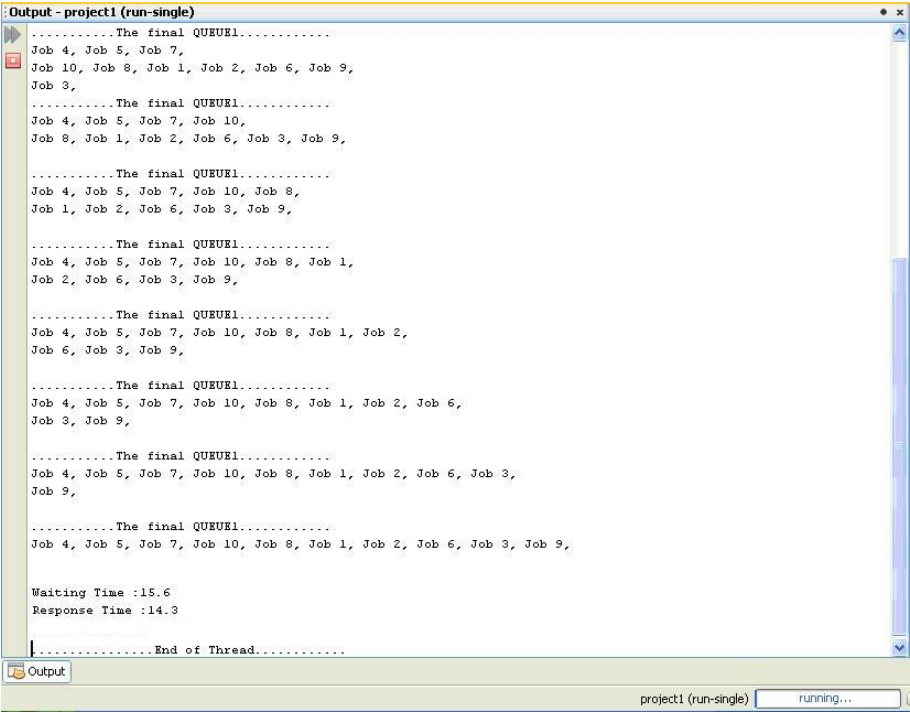


Figure 6.2: Submission of multiple jobs

In this figure it shows the combination for the job execution which is first come first serve, shortest job first then round robin queue (FCFS -> SJF -> RR). In this figure it shows the combination for the job execution which is first come first serve, shortest



```
Output - project1 (run-single)
.....The final QUEUE1.....
Job 4, Job 5, Job 7,
Job 10, Job 8, Job 1, Job 2, Job 6, Job 9,
Job 3,
.....The final QUEUE1.....
Job 4, Job 5, Job 7, Job 10,
Job 8, Job 1, Job 2, Job 6, Job 3, Job 9,

.....The final QUEUE1.....
Job 4, Job 5, Job 7, Job 10, Job 8,
Job 1, Job 2, Job 6, Job 3, Job 9,

.....The final QUEUE1.....
Job 4, Job 5, Job 7, Job 10, Job 8, Job 1,
Job 2, Job 6, Job 3, Job 9,

.....The final QUEUE1.....
Job 4, Job 5, Job 7, Job 10, Job 8, Job 1, Job 2,
Job 6, Job 3, Job 9,

.....The final QUEUE1.....
Job 4, Job 5, Job 7, Job 10, Job 8, Job 1, Job 2, Job 6,
Job 3, Job 9,

.....The final QUEUE1.....
Job 4, Job 5, Job 7, Job 10, Job 8, Job 1, Job 2, Job 6, Job 3,
Job 9,

.....The final QUEUE1.....
Job 4, Job 5, Job 7, Job 10, Job 8, Job 1, Job 2, Job 6, Job 3, Job 9,

Waiting Time :15.6
Response Time :14.3

.....End of Thread.....
Output
project1 (run-single) running...
```

Figure 6.3: Job selected using Combination of scheduling

job first then round robin queue(FCFS -> SJF-> RR).It will also count the average waiting time & average response time for jobs.Which is shown in the below table.

Table I: Calculation of job's average waiting time & response time

Calculation
->Average Waiting Time (Wt) for jobs
$Wt = (Wj1 + Wj2 + .. + Wji) / Nj ;$
->Average Response Time (Rt) for jobs
$Rt = (Rj1 + Rj2 + .. + Rji) / Nj ;$

6.2.2 Advance Reservation of Resources for Job Execution

From the WebMds getting the list of all available resources then it gives all information of the all available resources. Based on the available resources and requirement of resources there is Best resource is found. Then from multiple jobs to submitting on the middleware here job is selected in queue. Now for reserving the resource in advance to execute job on that resource which was reserved. After the selection of job that job is run on the plus manager tool for reserving the resource.

For advance reservation here Plus Manager tool & PBS(TORQUE) is installed on the main node of grid environment. Advance Reservation (AR) is a process of requesting resources for use at a specific time in the future. There are some systems that support AR capability, such as Globus Architecture for Reservation and Allocation (GARA). Here job1 is taken for submission & it is run on the plus manager tool which is in M.Tech group.

- Configuration Steps for Advance Reservation:-

(1) Registering reservations

-plus reserve -s startTime [-e endTime— -D duration]

-n numNodes [-U users] [-mem memSize][-ncpu cpuNum] [-arch arch] [-os os]

Rid [-sgeq qNames][jobname] [-x]

As example,

```
-plus reserve "2010-02-02T12:34:56+09:00" "2010-02-03T02:30:51+09:00" 02:32 2
lata 124M 2 x86 WINNT51 pc-1 SGE job4 -X
```

When using the `-x` option, the information is displayed in XML format.

(2) Canceling reservations

```
-plus cancel [-r rsvID] [-o owner]
-plus cancel pc-1 lata reservation is cancelled
```

(3) Changing reservations

```
-plus modify -r rsvID [-s startTime] [-e endTime] [-D duration] [-n numNodes]
[-U users]
```

(4) Discarding reservations

```
-plus destroy [-r rsvID] [-o owner]
```

(5) Confirming reservation operations

```
-plus commit -r reservationID
```

(6) Discarding reservation operations

```
-plus abort -r reservationID
```

(7) Viewing account of reservation usage

```
-plus account [-s startTime] [-e endTime] [-d days] [-f log file] [-o [owner]] [-n
[nodes]] [-l] [-S]
```

"Reservation is created for node pc-1".

Table II: Analysis of Results

Job ID	Group	Status	Reserved Node
Job4	Mtech	Reserved	pc-1
		Unreserved	pc-19,Pc-16,pc-18,pc-14,pc-11,pc-10
Job5	B.Tech	Reserved	Pc-19
		Unreserved	Pc-16,pc-14,pc-18 ,pc-11,pc-10

From this configuration steps we can conclude that which node are reserved in advance for which job, which is shown in the below table. Here it shows that job1 from Mtech group has reserved the node 1 for execute that job on this node. When once a node is reserved for job execution then it come out from all available resources which are not reserved. So new user can not submit any other job on that node at that time. Here for job2 pc-19 is reserved to execute the job on that node.

Chapter 7

Conclusion and Future Scope

7.1 Conclusion

It is envisaged that the grid infrastructure will be a large-scale distributed system that will provide high-end computational and storage capabilities to differentiated users. Complex scenario appeared while job submission & execution are identified. Out of the identified phases, 1st phase which is "selection of best resource" is implemented. Resource brokering algorithm finds a suitable & best match in the existing grid environment. It also obtains the demanding resources & compares with available resources. In the conclusion, it is observed that best grid nodes improve the performances of the job. 2nd phase of the scenario "Advance Reservation" is implemented. From the multiple jobs here one job is selected for submission using combination of different algorithms. Then that job is ready to run on that node, where user wants to reserve the node. Selected job is run on reserved node which is reserved in advance. From the results observations are shown that one resource is reserved for particular selected job.

7.2 Future Scope

In future, the improvements can be done by also taking into account the dynamic behavior of the grid resources. Multiple user can submit job from different middleware. At present work is done for local grid but next this work can be developed for VO (virtual Organization).

Appendix A

Troubleshooting

Error: while GT4 make - UNTAR FAILED

Sol: It can be done by editing the file in *GLOBUSLOCATION/var/lib/perl/Grid/GPT/wheregunzip* is being set,its called LocalEnv.pm.The root cause of the error is that gunzip version was returning gzip in the version string instead of gunzip.

Error: Starting Container gives error Check if port number and host are correct and postmaster is accepting tcp/ip connections.

Sol: Check if another instance of postgresql is not running.Check the log file for hint.if this is so change user to postgres use kill -SIGTERM process id and start postmaster again.

Error: cannot create regular file /opt/globus-4.0.1/etc/hostcert.pem: Permission denied (while signing certificate)

Sol:when grid-cert-request is run it creates three files hostcert request.pem, hostkey.pem, hostcert.pem. Here hostcert.pem file is empty remove that file.

Error: while running command `globus-url-copy -vb gsiftp` gives `globus xio:Unable to connect to nodeb.grid:2811`

Sol: start gridftp server using command `globus-gridftp-server -S -p 2811`

Error: `globusrun-ws -submit -streaming -F https://hostname:8443/wsrf/services/ManagedJobFactoryService -c /usr/bin/whoami` ERROR:Delegating user credentials... Failed. `globusrun-ws: Error trying to delegate globus xio: Unable to connect to hostname:8443 globus xio: System error in connect: Connection refused globus xio: A system call failed: Connection refused`

Sol: copy `globus-host-ssl.conf` , `globus-user-ssl.conf` , `grid-security.conf` to `/etc/gridsecurity/` certificates and `globus-user-ssl.conf.hashno` and `globus-host-ssl.conf.hashno` to `/etc/gridsecurity/`.

Error: GRAM Job Submission failed because the connection to the server failed (check host and port) (error code 12)

Sol: Your client is unable to contact the gatekeeper specified. Possible causes include: * The gatekeeper is not running The host is not reachable. The gatekeeper is on a non-standard port.

Error: `globus-mds-start` is not found in `$GLOBUS_LOCATION/sbin`

Sol: GT4 uses web-mds other than MDS follow `webmdsAdminGuide` to start MDS service.

Error: Starting Condor daemons Error in MasterLog cant obtain lock on Instance-Lock

Sol: The condor master process tries to lock a file when it starts up to prevent you from starting multiple instances of the condor daemons. There are two instances of condor master running so first stop condor master and then it start again..

Error: Running condor submit gives ERROR: submitting jobs as user/group 0 (root) is not allowed for security reasons.

Sol: When we say to run the daemons as root, that means you **start** them as root. They then switch to a non-root user whenever they dont need root power. This is to reduce the risk of screwing up the system if theres a bug or other problem. When the daemons need to the something that requires root power (like starting a job as the user), the daemons switch to root, do the deed, then return to nonroot. When we say to run the daemons as root, that means you **start** them as root. They then switch to a non-root user whenever they dont need root power. This is to reduce the risk of screwing up the system if theres a bug or other problem. When the daemons need to the something that requires root power (like starting a job as the user), the daemons switch to root, do the deed, then return to non-root. Also during configuration u have to give ownership to globus user using command:

Condor configure -owner=globus -type=submit,manager This will allow globus user to submit job.

Appendix B

List of Paper Accepted/Published

Paper accepted in The 2010 International Conference on Informatics,Cybernetics, and Computer Applications(ICICCA2010),July-19-21,Jain University,Banglore,India,
"Handling Complex Use Scenarios for Selection of Best Resources in Grid Environment"

Website References

- [1] <http://en.wikipedia.org/wiki/Globus>
- [2] <http://www.globusconsortium.org/tutorial>
- [3] <http://www.globus.org/toolkit>
- [4] <http://www.bestgid.org/>
- [5] www.redbooks.ibm.com/redbooks
- [6] <http://www.cs.wisc.edu/condor>
- [7] <http://www.cs.wisc.edu/condor/condor-V7.2.1-Manual.pdf>

References

- [1] Kleinrock, L. MEMO on Grid Computing, University of California, Los Angeles, 1969.
- [2] Foster,I, Kesselman,C. and 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.
- [3] Jarek Nabrzyski, Jennifer M. Schopf & Jan Weglarz, Grid Resource Management State of the art and Future trends, Kluwer Academic Publisher.
- [4] Yanmin ZHU, A Survey of Grid Scheduling, Department of Computer Science Hong Kong University of Science and Technology
- [5] Ferreira, L., Bieberstein, N., Berstis, V., Armstrong, J., Introduction to Grid Computing with Globus, Redbook IBM Corp.
- [6] Chunlin Li,Layuan Li,Three-layer control policy for grid resource management, Department of Computer Science,Wuhan University of Technology,Wuhan 430063,PR China, Dec,2008.
- [7] Arshad Ali, Richard McClatchey, Ashiq Anjum,Irfan Habib,Kamran Soomro ,From Grid Middleware to a Grid Operating System,National University of Sciences and Technology, Rawalpindi, Pakistan,2006,IEEE.
- [8] Erik Elmroth , Johan Tordsson, Department of Computing Science,Resource selection based on performance predictions,Umea University,Sweden ,June 2007.

- [9] Yonghong Yan, Barbara Chapman, Department of Computer Science, University of Houston, Scientific Workflow Scheduling in Computational Grids, May, 2007.
- [10] Wu-Chun Chung, Ruay-Shiung Chang, Department of Computer Science, A new mechanism for resource monitoring in Grid computing, National Dong Hwa University, Hualien, Taiwan, June 2008.
- [11] M. Bohlouli, M. Analoui, Predicting Resource Requirements of a Job in the Grid Computing Environment, International Journal of Computer Science 3; <http://www.waset.org>, Spring, 2008.
- [12] Marek Wiecezorek, Mumtaz Siddiqui, Alex Villazon, Radu Prodan, Thomas Fahringer, Applying Advance Reservation to Increase Predictability of Workflow Execution on the Grid, University of Innsbruck, May, 2007.
- [13] Anthony Sulistio, Uros Cibej, Sushil K. Prasad, and Rajkumar Buyya, GarQ: An Efficient Scheduling Data Structure for Advance Reservations of Grid Resources, GRIDS Laboratory, The University of Melbourne, Australia, July, 2008.
- [14] Globus Consortium, <http://www.globusconsortium.org>
- [15] Matthew L. Massiea, Brent N. Chun, David E. Culler, The ganglia distributed monitoring system: design, implementation, and experience, June, 2004, ScienceDirect.
- [16] Manzur Murshed Gippisland, Rajkumar Buyya, Using the GridSim ToolKit for Enabling Grid Computing Education, National Kolkatta University, June, 2005, ScienceDirect
- [17] Eclipse Project, <http://www.eclipse.org/eclipse/>
- [18] www.g-lambda.net/packages/plus
- [19] WS-GRAM. <http://www.globus.org/toolkits/docs/4.2.1/execution/wsgram>

Index

Best Resource Selection Algorithm, 20

Ganglia, 7, 31

GARA, 45

Gflops, 20

Globus Configuration, 15

Gridsim, 7

GRIM, 12

GRIR, 11

Kflops, 25

MDS, 24

Middleware, 5

MIPS, 20, 25

PBS, 5, 16, 19

SGE, 5, 17, 19

TTD, 20

VOs, 8

WebMds, 20, 25, 26