Dynamic Resource Allocation in Desktop Grids

Bismi A S

II ME
Department computer Science and Engineering SNS college of Technology,Vazhiyampalayam Sathiy Road,coimbatore -35,Tamil Nadu,India

Dr.S.Karthik

Professor,&Head
Department Computer Science &EngineeringSNS college of Technology,Vazhiyampalayam
Sathiy Road,coimbatore -35,Tamil Nadu,India

Abstract

Grid computing (or the use of computational grids) is the application of several computers to a single problem at the same time. In grid computing ,the computing becomes pervasive and individual users(or client application)gain access to computing resources (processors ,storage ,data,applications,and so on)as needed with little or no knowledge of where those resources are located or what the underlying technologies, hardware ,operating system and so on. This paper address the problem of resource scheduling ,and there are so many resource scheduling algorithms .Resource scheduling algorithm may be centralized one and distributed one. One of the latest one is the multi-variable best fit algorithm, which deals with storage systems, and it enhanced in this work. and this algorithm is dealing with data grids for blade servers. so, this algorithm is modified for computational grids, especially with desktop systems with minimum spanning algorithm for node selection and it is giving good results. In desktop grid environment, dynamic scheduling becomes very important and globus like toolkits are not having the scheduler as its own, and it needs other schedulers are not specially for computational grids.

Keywords: grid computing, Dynamic resource allocation, Resource monitoring, Resource assignment system.

1. Introduction

Today's IT systems are connected by any network and execution software is spread over, and they need a centralize25d computational facility for reducing cost and to improve manageability. Efforts are now being made to increase the degree of sharing of these consolidated computing and to provide them to the end-user as a utility. Such systems are being coined as Computing Grids. In such systems, geographically distributed computing sites host the nodes, which are allocated dynamically and on-demand to the applications of the end-user.

Grid computing is a category of distributed computing applies the resources of many computers in a network to 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. This environment is proposed as the way to enhance the efficiency of underlying system by maximizing utilization of distributed resources. Such system pools together the resources of many work station to create a virtual computing reservoir. The advantage of using such platform includes easy access to powerful computing facilities and effective use of distributed resources. Grid computing is a form of autonomy parallel and distributed computing which involves coordinating and sharing of distributed and autonomous application, data storage, network resources autonomously and efficiently to satisfy user's quality of services as per a Service Level Agreement (SLA). Resource allocation is one of the important system services that have to be available to achieve the objectives from a grid computing. A common problem arising in grid computing is to select the most efficient resource to run a particular program. Also users are require to reserve in advance the resources needed to run their programs on the grid .The execution time of a program can be estimated by using aspects of static analysis, analytical bench marking and compiler based approach.

Scheduler monitors the client and collect the parameter value regularly. Using this parameter values scheduler calculate the weight values .Depending on the client request scheduler selects the apt computation node for the request.

2. Grid Architecture

One of the fundamental system management services needed in the middle ware to enable the vision of Computing Grids is a Resource Allocation service. This service is responsible for the dynamic allocation of a fraction of a compute server's resources in response to an end-user request. Fig.1.1 shows users allocation core services within the grid core middleware. Resource allocation mechanisms are responsible for the allocation of user's applications to grid execution hosts according to SLA. SLA describes an application requirement in terms of resources to ensure quality of services to the application.

The architecture view of a grid environment is shown bellow

+

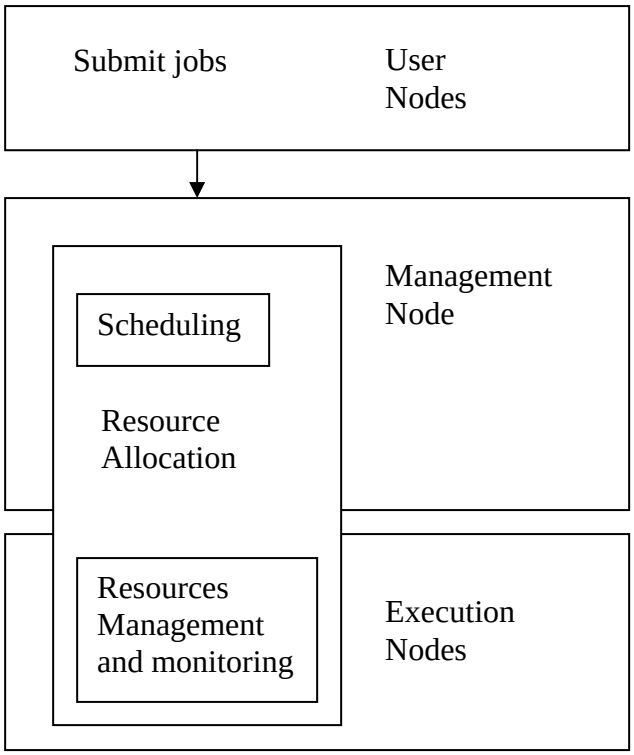


Figure 2.1 Grid Architecture

Today's IT systems typically pre-install, pre-allocate, and reserve the systems for end-customers applications, leading to over-provisioning and higher costs. On the other hand, a computing grid envisions desktop resources to be sharable across end-customer's applications and be allocated dynamically as the need arises. This brings out the need for a resource allocation service that has to consider the real-time system, and the dynamic requirements of requests while making an allocation decision.

The resource allocations made, must further meet the minimum performance requirements of the hosted applications, while avoiding over provisioning of resources so as to maintain high system utilization. Prior work has looked at building resource allocation services for supporting batch applications, three tier enterprise applications and for server applications. This work is like to address the needs of interactive remote desktops, which are typically more sensitive to performance needs. This proposed architecture services requests for remote desktop In Grid module an interface for Process Request is created, which includes client side processing and client-server sessions from end-users dynamically, and allocates a desktop session on-demand for the end-user's request. The fraction of the resources to be chosen is, determined through the dynamic generation of the performance model for the requested remote desk-top session. The dynamic generation takes place using pre-generated application performance models for the applications that would execute within the requested remote desktop session. Actually the strength of the Grid technologies is based on the scheduling and access control of resources with the goal of increasing the utilization and agility of enterprise infrastructure.

3 . Architectural Design

The system model considered in this paper is a single intranet. This intranet consists of a server, desktop systems and a resource management system. This proposed resource allocation service components are resident on the resource management system Figure 2.1 shows the flow diagram of the sequence of steps executed in the system. The end-users submit requests for remote desktop sessions to the Resource Management System. The resource management system then allocates a desktop system to the user's request for the remote desktop session. A request to start the remote desktop session is then dispatched to the allocated compute node. Once the session is started, the user interactively starts applications through the established remote desktop session connection. This is shown as top level requests in Figure 2.1. These middle level requests go through a user verification process. Once the applications are started, the user interacts with those applications through an application specific workload. We thus have a hierarchical request structure in the system, top level requests, middle level requests, and application specific workload.

Resource allocation involves two grid system services which are resource monitoring and scheduling. Resource monitoring refers to recording and reporting information about each resource's capabilities, usage and future reservations. A resource can be a CPU, disk, memory, network bandwidth etc. The information on a resource is then retrieved by the scheduler which decides on the allocation of user's applications to appropriate resources. The grid computing resource allocation problem is only a special case of traditional parallel computing resource allocation problem addressed in traditional parallel and distributed system such as symmetric multiprocessor machine(SMP),massively parallel processors machine(MMP) and cluster of workstation(COW).

The work includes two phases as Grid module and Scheduler module.

socket creation. Through a runnable interface user can submit the job and then client inherits the grid class. Client

also finds child process and assigns separate id for each child process.

Process execution includes server side processing. Scheduler keeps track about each nodes in which the processes are executing. Scheduler sends the response /result to the client node. Server side processing receives class files from client and generates id and sends to the client. Scheduler selects best node to allocate the request. Send the request to the selected node along with class files.

Periodically each node will update its parameter status with the scheduler by sending a parameter thread which helps to update the status of each node.

The other module is scheduler module, which includes request/response routing and scheduling. request/response routing implements protocols for dentifying request/response, i/o messages. Requests will be allocated to each node by the scheduler & provides identity for each request. Responses will be routed to scheduler. Request from the client & response to the scheduler are to be identified and redirected. Scheduling builds a parameter tree and selects the apt node for process execution.

For node selection the parameter status is to be analyzed. Some parameters plays positive role in compute node selection and some other parameters play negative role.

- CPU Includes

-CPU speed (If high then the program execution can be done faster. So it plays +ve role.)
-CPU type(Performance depends on processor type (P III, P IV etc.)
-CPU usage(Determines currently executing processes. Throughput of the computing node can be determined from this parameter which can be measured by the server system with its previous performance. If high then the system performance is also high.It plays +ve role.

- Percentage of memory availability:
  Calculates the ratio of total free memory to total available memory which helps to allocate the task to the nodes having more memory capacity.
  High storage availability gives better performance for database oriented program and gives more swap area for other program execution. It plays +ve role.
- Network Bandwidth:
  High network Bandwidth is a good quality for database oriented and also interactive/communicative programs. It plays +ve role.
- Pending work of a compute node:
  High pending work increases the burden of a system. So it plays –ve role in selecting a compute node.
- Number of threads executing at a node:
  More number of threads will decrease the time slot given for each thread. It also plays –ve role.

So +ve parameters can be added and –ve parameters can be subtracted while calculating the weight of a compute node.

- Prior to request allocation the scheduler will check about the platform availability on that particular node.

These six parameters totally give a solution to all the three types of programs. For finding these parameter values, there are standard shell commands available for some of the parameters in Linux and in Windows also. For some other parameters like free memory availability, are available in win32 directory under Windows and /proc directory in Linux. Because these platforms are standard and have standard commands, the results of these commands will also be perfect.

4. Implementation

Initially resources are to be identified and details of each resource should be known such as network capacity, storage capacity, CPU utilization etc. Socket program/Remote shell commands can be used to read the additional details. Algorithm is implemented using Kruskal's algorithm, i.e., one source to all destinations are needed.

Many scheduling algorithms are available with First come first serve scheduling, Priority scheduling, Best fit scheduling. All these algorithms are executed at the time of request. If the compute node timely updates these parameter values in the server then the server can calculate the weight previously. At the time of request, the time taken to calculate the weight can be reduced. Depending on the request the resources are selected and ordered in a particular manner such that with minimum spanning time a apt compute node can be selected.

Experiments can be done using presently available resource allocation techniques.Network Peak time and off time results can be measured for both and can be compared. Work load and weight assignment can be modified.

MULTI-VARIABLE MINIMUM SPANNING TREE ALGORITHM

The pseudo code for a multiple variable minimum spanning algorithm that takes resource requirement heuristics into consideration for resource assignment is given below. However, as mentioned earlier, we allow resource sharing i.e. there would be multiple remote desktop sessions allocated on the same compute node simultaneously. A minimum spanning algorithm for assigning compute nodes to remote desktop sessions would always try to pack up bins tightly.

This would enable to assign more sessions onto different compute nodes and should help in reducing the wait time for the requests in the Pending Queue. We therefore consider a minimum-spanning algorithm for resource assignment. However, we have to consider multiple variables in the algorithm - CPU, network bandwidth, and storage bandwidth. For a particular remote desktop session, one or more of these resources may be a bottleneck resource.

The weight functions are introduced corresponding to each of these fine grain resources and adjust the weight assignment accordingly for the bottleneck resource variables. For example, for CAD design sessions, the CPU would be the bottle-neck resource variable and weight should depend on CPU utilization values for such sessions. For real time applications, the network latency would be the bottle-neck source variable. Further, the algorithm determines the difference between the available and required resource utilizations, and assigns the weight functions as inversely proportional to these delta values.

Thus, it does weighted minimum spanning values along multiple dimensions. The weights are assigned for the different parameters, variables as functions, and to pick the compute node that has the highest aggregate weight across dimensions.

The resource and latency requirements used for the remote desktop sessions in the algorithm are those obtained from the finding remote desktop session model by using the socket program concept. The algorithm is given below and it uses the following variables also. C represents CPU capacity, N represents Network capacity, S represents the Storage Capacity, T represents the Number of threads running in a system, TP rep-resents the Throughput of the

2.Pick the compute node with the maximum assigned weight Weffective for this request. In case of equally ranked compute nodes, we pick the one with the least load where load is defined in terms of CPU utilization.

In case of equally ranked compute nodes, we pick the one with theleast load where load is defined in terms of CPU utilization.

system, PR represents the Pending Requests in a system. The algorithm is as follows

Algorithm:

1. For each compute node that satisfies site admission control test

a. Determine the free CPU cycles, network bandwidth, and storage bandwidth available on this compute node for a user's request

b. Determine the delta values between the available resources from step a., and the desired resources for the requested remote desktop session. These delta values are denoted as Cdelta, Ndelta , Sdelta , Tdelta , TPdelta, PRdelta .

c. We now assign the following weights:

WC = f(Cdelta, Compute Intensiveness)

WN = f(Ndelta, Average expected display data size)

WS = f(Sdelta, Data intensiveness)

WT = f(Tdelta, compute Intensiveness)

WTP = f(TPdelta, Compute intensiveness)

WPR = f(PRdelta , Compute intensiveness)

The weights (WC, WN, WS, WT, WTP,WPR) are inversely proportional to the

first parameter (Cdelta, Ndelta , Sdelta , Tdelta , TPdelta , PRdelta) and directly

proportional to the second parameter Compute intensiveness, Average expected display data size, Data intensiveness, compute intensiveness, Data intensiveness, compute intensiveness, Compute Intensiveness respectively.

d. The effective weight of this compute node for the currently considered assignment is
$W_{effective} = WC + WN + WS - WT + WTP – WPR$

5 .Experimental Results

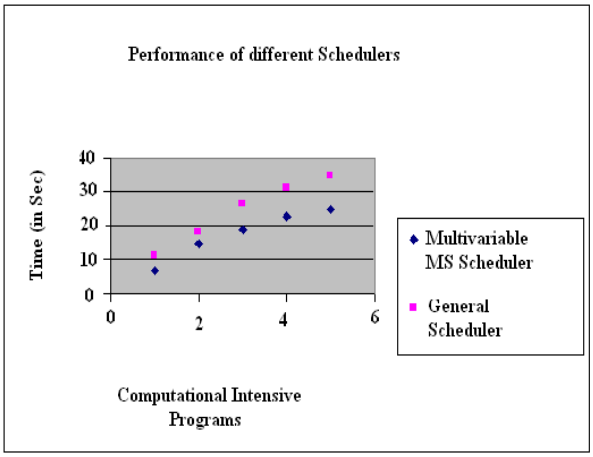
WEIGHT CALCULATION IN DIFFERENT NODES

Each parameters are in different forms and they are converted to a linear value called delta value.
After calculating all the weight values, Weffective is calculated as per the equation. Minimum spanning node is identified as per the algorithm

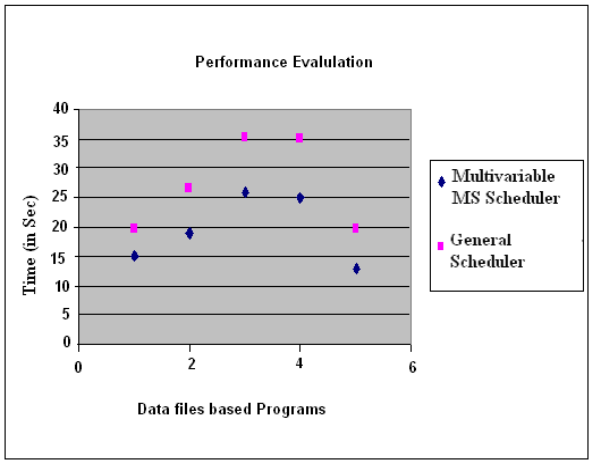
PERFORMANCE EVALUATION

Output is obtained with all the parameters included. Experiments are done using presently available resource allocation techniques and got the results. When these results are compared with the algorithm developed and nearly 50% good performance in network Peak Time. Network off time results are giving 20% good performance. Work load and weight assignment can be modified to get these results.

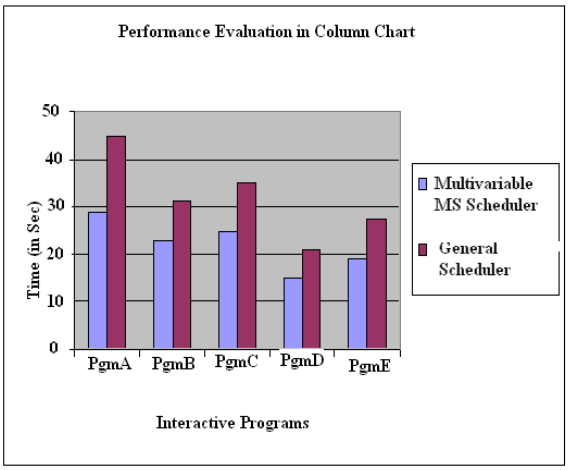
Performance evaluation charts



Computational intensive program



Data files based program



Interactive program

6 Related Work

In grid computing divisible load theory has emerged as a powerful tool for modeling data-intensive computational problems. Divisible load theory offers a tractable and realistic approach to scheduling that allows integrated modeling of computation and communication in grid computing systems. . Divisible load theory does not recognize precedence relations among data, it assumes that computation and communication loads can be partitioned arbitrarily among numerous processors.

Present schedulers are available for all types of Grids and specifically for Data Grids. They use Multivariable Best Fit algorithm for Data Grids.

7 Conclusion

Using this new proposed algorithm, resource allocation can be done in a better way with the computational grid environment. The performance evaluation shows that the new algorithm gives better results than the present algorithm.

Unlike a general algorithm this proposed one is specifically suitable for computational grids. The nodes are selected according to the present conditions of a node. User friendly environment is created for request submission.

8 References

[1] Liang Chen, Gagan Agarwal, “Resource allocation in a Middleware for Streaming data”, 2nd Workshop on Middleware for Grid computing, Toronto, Canada.

[2] Leila Ismail, Bruce Mills, Alain Hennebelle, “A Formal Model of Dynamic Resource Allocation in Grid Computing Environment”, IEEE international conference on Networking and parallel distributing computing, 2008 August.

[3] Nabrzyski, J., Schopf, J.M.,Weglarz J. "Grid Resource Management:State of the Art and Future Trends". Kluwer Academic Publishers, 2003.
[4] Rolia J., Pruyne J., Zhu X., Arlitt M. "Grids for enterprise applications", Proceedings of 9th Workshop on Job Scheduling Strategies for Parallel Processing, Seattle, WA, June 2003.
[5] Vanish Talwar,Biksha Agarwala, Sujoy Basu, Raj kumar, Klara Nahrstedt, " Resource allocation for Remote Desktop Sessions in utility Grids", Concurrency and Computation: Practice and Experience,InterScience, November 2006.