Processor Co-Allocation Enabling Advanced Reservation of Jobs using Priorities in MultiCluster Systems

A.C.Kaladevi,
Department of Computer
Science and Engineering,
Sona College of Technology,
Salem, India.

M.V.Srinath,
Department of Computer
Science and Engineering,
Mahendra Engineering
College,
Thiruchengode, India

C.Babitha Lilly Mary,
Department of Computer
Science and Engineering,
Sona College of Technology,
Salem, India.

ABSTRACT

Grid Computing is intended to provide resource sharing and solve huge amount of complicated problems. In Multicluster Systems, jobs may require co-allocation, that is, the simultaneous allocation of resources such as processors in multiple clusters. Such jobs may have low runtimes because they access more number of resources. In processor co-allocation, advanced processor reservation is needed. But this becomes one of the challenges in processor co-allocation. Advanced Processor Reservation Algorithm based on the priority of jobs using the usage of memory is implemented in this paper. The results indicate that the overall performance of the algorithm is increased by satisfying large number of reservation requests using the priority.

Keywords

Grid Computing; Processor Co-allocation; Advanced Processor Reservation.

1. INTRODUCTION

Grid Computing involves heterogeneous, geographically distributed resources that can work on a task together. In today's world of high speed computing, computers have become very powerful and also it is enough to run complex applications. But still we have numerous complex scientific experiments, outlines for advanced modeling, business modeling, astronomical research and wide variety of simulations which require large amount of computational resources. Grid Computing satisfy the above requirements.

A grid environment is more modular and doesn't have any single points of failure. If one of the servers within the grid environment fails, then there are plenty of other resources can able to process the load. Jobs can automatically restart if a failure occurs.

The Grid applications use resources such as processors in multiple clusters simultaneously. So they need processor co-allocation. With this processor co-allocation, the execution time of grid applications may increase because of communication overhead and heterogeneity among clusters. [1] Despite this drawback, Co-allocation provides lower average job response times.

The basic concept in the grid technology is to identify the resources in the computer network that are geographically spread and allow the computers to share the resources to solve the problems in short time and at economical rate. Grid is a problem-solving environment in which one or more users can submit the job without knowing where the resources are available.

Grid scheduling is the process of scheduling jobs over Grid resources. Task scheduling is the mapping of tasks to a selected group of resources which may be distributed in multiple domains. The scheduling of task is used to enhance the functional capability of the Grid System.

Advance Reservation (AR) is a process of requesting resources that can be used in a specific point of time. Some of the commonly used resources that can be reserved are CPUs, memory, disk space and network bandwidth. AR guarantees the availability of resources to users at the required times. Advance reservations have been widely used for predict the performance of the system, satisfying resource requirements and providing guaranteed quality of service to applications. GridSim [5] is a tool mainly used for Advance Reservation. The [11] introduced scheduling algorithm for advance reservation. Scheduling Algorithm mainly deals with a Reservation Scheduler with Benefit Functions and Priorities. Benefit Function is used in this algorithm to evaluate the "profit" for the client. The Resource Broker provides pairing services between service requester and service provider. This type of pairing enables the selection of best available resources from the service provider for the execution of specific task. This involves allocation of resources and support functions such as combination of resources and supporting user's deadline.

Schedulers are the main application of Grid Computing. This is responsible for allocating resources needed for specific job, partitioning and parallel execution of jobs. There are three types of Grid Schedulers such as Application Schedulers, Job Schedulers and Resource Schedulers used to reach the performance goals of the application. Application Schedulers uses the performance of individual application. Job Schedulers are used to optimize the overall performance of the system. Resource Schedulers coordinates all the user requests for accessing the resource.

Other applications include discovery and scheduling of tasks, distributing application codes to system nodes, self-recovery and self-management. Resource Discovery is defined as searching and locating resource which are suitable for executing jobs.

2. RELATED WORK

In [1] Omer et al. a job model for processor co-allocation in multicluster grids is presented. The KOALA grid scheduler has been designed for multicluster systems. For submission of a job, KOALA uses one of the placement policies to try to place job components on execution sites. In [2] Ozan et al. presented two job placement policies such as Cluster Minimization (CM) and Flexible Cluster Minimization

(FCM). The CM policy sends job requests, in which jobs are divided into components and each job having a fixed processor requirement, whereas FCM sends job requests in which jobs are represented as a single component that is allowed to be divided into smaller components.

In [5] Anthony Sulistio et al. stated that GridSim has been extended with the ability to handle creation of a new reservation for one or more resources, commitment of a newly created Reservation and activation of reservation in the start time. H. Mohamed et al. [6] presented the design and implementation of a co-allocating grid scheduler called KOALA that meets the co-allocation challenges such as locating sufficient resources in geographically distributed sites, allocating, managing and coordinating the execution of single jobs at multiple sites.

In [7] H.H. Mohamed et al. used the method called Close to Files Policy (CF). CF tries to place job components on clusters with enough number of idle processors which are close to the execution sites where the input files are placed. Rue Min et al. [11] presented a scheduling algorithm for advance reservation based on the priority of jobs. In [12] Erik Elmroth presented an algorithm for Grid resource manager to perform resource brokering and job scheduling. In [13] K. Czajkowski et al. presented the co-allocation problem and the mechanisms that allow an application to select resources during the co-allocation process. These mechanisms address issues relating to the allocation, monitoring, control, and configuration of distributed computations.

In [14] A. Iosup et al. presented that many advances is increased by the fast development of single, large clusters in grid resource management and these advances cannot be achieved without an appropriate toolbox. The current grid simulation environments lack system modeling and workload features. So they presented the implementation of DGSim, a framework for simulating resource management architectures. An efficient and effective algorithm based on the combinational backfilling strategy [16], can select huge amount of jobs combined from the job waiting queue to maximize the use of idle resources. This effective algorithm can have the lower waiting time of jobs and higher utilization of resources when compared to the existing backfilling algorithms. Kwang Mong Sim [17] describes the bargaining models for Grid resource allocation, design considerations, market dynamics, relaxing bargaining terms, and co-allocation of resources when building Grid negotiation mechanism.

Luis Tomas et al. [18] presented two techniques that deal with the idle period of resources. One of the techniques consider the heterogeneity of resources to estimate the job execution. This provides the efficiency to meet the QoS requirements of users.

Claris Castillo et al. [19] considered the problem to provide QoS guarantees through advance reservation of resources. They used the concepts to provide a framework for tackling the resource fragmentation and formulating some of the scheduling strategies. The main contribution is to provide a timely, practical and efficient solution to the problem of scheduling.

Ponsy R. K Sathia Bhama et al. [20] described an algorithm, which deals with single and multi-component jobs. In Single component jobs, the data intensive nature is considered and the cluster chosen for data transmission is having low network delay. In multi-component jobs, the Worst Fit policy is used which provides the solution for the heterogeneity of the Grid environment.

3. PROPOSED WORK

In this proposed work, advance reservation is mainly concentrated based on the priority of jobs and the Benefit Function.

3.1 System Architecture

Fig 1: System Architecture

The job given by the client is placed in the placement Queue based on first come first serve order. Index server is a server which contains all available resources. Index server is maintained separately. The jobs placed in the placement Queue is given to the scheduler for advance reservation. If the resource needed for the job is not available in the index server then the job is placed in the tail of the placement Queue. For advance reservation, first the jobs are placed in the shortest job first queue based on the used memory of the job. After placing the jobs the Scheduler finds out the total, free and used memory for the job. Then the Scheduler scans the job from first to last in the queue based on the used memory of the job. After sorting the job, priority should be given for each job in advanced reservation. First priority is given for the job having least used memory. Then the prioritized job is given to the cluster which has the resource needed to satisfy the prioritized job requirement. Second priority is given for the job having second least used memory. Then the job is given to the cluster and check whether the resource present in the cluster is used by another job or not. If it is used by another job then the scheduler allocates this job to another cluster which satisfies the job requirements. Otherwise the job is allocates to the same cluster. Next priority is given to the third least used memory job and so on.

3.2 Index server

Scheduler is used for advance reservation. For selecting the resources needed for job request, Scheduler interacts with the Index server to find out the resources available in it. The Index server is like a repository to store all resources. Figure 2 shows the interaction between scheduler and the index server. First the broker contacts index server to discover what Grid resources are available. After discovering resources, the Scheduler finds the resources needed for the job request from the total available resources.

Fig 2: Interactions between Scheduler and Index server

3.3 Grid Scheduling Process

A Scheduler uses three stages to schedule a job when it involves multiple sites. The three stages are: Resource Discovery, System Selection, and Job Execution as shown in Fig 3, 4 and 5.

3.3.1 Resource Discovery

Fig 3: Resource Discovery

Resource Discovery helps the scheduler to select a set of resources. At the beginning of this phase the resource set is empty and the end of this phase, there is a presence of resources which satisfies the user's requirements. Authorization Filtering is generally assumed, that a user will know which resources he has to access. At the end of this step the user will have a list of resources which is to be access. In Application Definition, a user must be able to specify minimum set of requirements in order to filter it from the set of resources. In Requirement Filtering, users filter out the resources that do not meet the minimum job requirements.

3.3.2 Resource Selection

Fig 4: Resource Selection

Group of resources are available in the index server. From these resources, a user needs to gather the information about the resources in question. Based on the application and resources in question, different information such as load of machines, queue length, physical characteristics may be needed. After gathering the information about the resources,

the scheduler selects the system (cluster) in which the job should run.

3.3.3 Job Execution

Fig 5: Job Execution

The third stage of scheduling process is the execution stage. In Advance Reservation, to make the best use of resources, all of the resources will have to be reserved in advance. Once resources are selected, the jobs must be submitted to the resources in Job Submission. The preparation stage may involve setup or any other actions needed to prepare the resource to execute the application. Depending on the job and its running time, users may monitor the progress of their application. In Job Completion the user needs to be notified about the job completion. After a job is run, the user may need to retrieve the results to do analysis for clean-up tasks.

3.3.4 Notation and Mathematical Model

Advanced Reservations are performed by a centralized unit called Reservation Scheduler. Benefit Functions are used to quantifying the CPU requirements. In this function, the Scheduler checks the availability of resources from the reservation start time to the end time.

Let c be the number of clusters in the Grid environment. For each job, request R_q arriving at the reservation Scheduler following parameters are defined

ts: reservation start time for R_q

te: reservation end time for R_q

pk: priority of job

Bk: benefit function associated with R_q

3.3.5 Reservation Scheduling Using Priority of Jobs and Benefit Functions

This section examines the reservation scheduling algorithm using the priority of jobs. The reservation Scheduling algorithm uses priority of jobs and benefit functions. At first, the algorithm sorts the requests in ascending order based on the used memory of the job. Then the start and end time of the reservation request is obtained. After that the $MEMORY_{total}$ for each job is calculated. Then get the $MEMORY_{used}$ for each job. This is calculated by taking the difference between $MEMORY_{total}$ and $MEMORY_{free}$. Then Benefit Function is determined and also sets the priority for each job by using the used memory. Benefit function determines the resource availability in the machine from start and end time of the reservation. If the resource in the machine satisfies the used memory requirement of the job then give first priority to that job. If the

resource in the machine does not satisfies the used memory requirement of the job because the resource present in the machine is used by another job then give second priority to that job and so on.

The parameters required for the algorithm are:

R_q: the request for reservation

M_i: the i^{th} machine in the system

MEMORY_{used}: amount of used memory for the job

MEMORY_{free}: amount of unused memory for the job

MEMORY_{total}: total amount of memory in the machine

Resource_{availablej}(ts, te): the availability of resources to reserve the request within duration ts to te in machinej

B_{jk}: The benefit function for jth machine

B_{ik}: The benefit function for ith machine

for all requests R_q

sort the requests in ascending order based on the size of the job

Sort(Job Request, Ascending Order)

for each sorted request

get ts and te

calculate **MEMORY_{total}** in each machine

for each machine M_i in the system

get **MEMORY_{free}**(ts,te) for each cluster

MEMORY_{used} = **MEMORY_{total}** - **MEMORY_{free}**

if (Shortest Job Queue contain reservation request for Job)

B_{jk} = **B_k**(**Resource_{availablej}**(ts,te))

B_{ik} = **B_k**(**Resource_{availablei}**(ts, te))

if(Resource in Machine satisfies **MEMORY_{used}** requirements for a job)

then set priority pk as 1

Resource_{availablei}(pk) $\leftarrow$ 1

else

if(Resource in Machine does not satisfies **MEMORY_{used}** requirements for a job because of the resource used by another job)

then job is allocated to another machine and set priority pk as 2

Resource_{availablej}(pk) $\leftarrow$ 2

endfor

Algorithm: Reservation Scheduling Algorithm Using Priority of Jobs and Benefit Functions

4. EXPERIMENTAL SETUP

Grid environment can be simulated using several Grid simulators e.g. GridSim. Grid simulators enable Grid users to work on Grid environment without having to worry about the external factors that may influence the Grid environment. In this proposed system, we implemented the concept by using GridSim toolkit. The motivation for using simulation consists of the following factors:

- Setting up a Grid environment is expensive, time consuming and resource intensive.
- The real test-bed does not provide a repeatable and controllable environment for experimentation and evaluation of scheduling strategies.
- Simulation is very effective in working with large scale problems.
- Simulation allows analyzing existing as well as new economic models and scheduling algorithms.

Fig 6: Gridlet and Resource Creation

In Fig 6, ten gridlets and ten resources are created. Each resource created is displayed in a message box.

Fig 7: Job Allocation

In Fig 7, five jobs such as Factorial, Prime number generator, Fibonacci Series, Sum of Integers and Odd number Generation are shown.

Fig 8: Memory Information of Job

Fig 8 shows the memory information such as used and free memory for factorial job.

Fig 9: Priority for first Job

The Scheduler sorts the used memory of each job. Based upon the sorted used memory, highest priority is given to the least used memory job. It is shown in Fig 9.

Fig 12: Cluster used for first Job

In Fig 12, it shows that the resource found is present in Cluster 1. So the job is given to that cluster for execution.

Fig 10: Information Message for first Job

In Fig 10, information's such as disk space of the resource, status, Cluster, Used memory and the name of the job is shown.

Fig 13: Priority for second Job

Fig 13 shows the priority and used memory for second least used memory for a job.

Fig 11: Resource allocated for first Job

Based upon the information gathered, the resource for Factorial Job is found and is shown in Fig 11.

Fig 14: Information message for second Job

In Fig 14, it shows the information message for second job. But the resource found for the second job is already allocated to the first job. So the status message shows "used" message.

Fig 15: Finding New Resource

In Fig 15, it shows the resource finding for second job.

Fig 16: Cluster used for second Job

In Fig 16, it shows that the resource found for second job is present in Cluster 2. So the second job is allocated to that Cluster for execution.

4.1 Result Analysis

This section provides some results from a simulation study designed to evaluate the performance of the algorithm provided in the previous section.

Fig 17: Number of resources vs. Number of clusters

In Fig 17, Clusters and the Resources for the job is shown with respect to the CPU time and cost. 10 machines participated in the simulation. The total CPU time taken for the simulation is 106 sec and the cost is Rs 328. As shown in

Fig 17, the total CPU time is increased when compared to the total cost of the simulation.

Fig 18: Number of resources vs. Number of jobs

In Fig 18, the number of resources and clusters needed for a job is shown based upon priority and the used memory. As shown in Fig 18, the used memory and the priority are increased. This increased priority for job is based on the sorted order of the used memory. The job which has the least used memory having highest priority.

While comparing with the existing system, overall performance and large number of advanced reservation requests using priority is achieved in the proposed system

5. CONCLUSION AND FUTURE WORK

With processor co-allocation advance processor reservation is needed to reserve jobs. But this reservation becomes one of the challenges in processor co-allocation. So in this paper we proposed advance processor reservation based on the priority of jobs. Priority based reservation promotes the efficient utilization of resources and time for which the grid technology is intended.

Large numbers of applications are benefit from processor co-allocation, but still there are many issues that need to be overcome for co-allocation. So our Future work on Advanced Reservation will look into the system reliability and cluster communication speeds.

6. REFERENCES

- [1] Omer Ozan Sonmez, Hashim Mohamed, and Dick H.J. Epema, "On the Benefit of Processor Coallocation in Multicluster Grid Systems," IEEE Transactions on parallel and distributed systems, June 2010.
- [2] O.O. Sonmez, H.H. Mohamed, and D.H.J. Epema, "Communication-Aware Job Placement Policies for the KOALA Grid Scheduler," Proc. Second IEEE Int'l Conf.e-Science and Grid Computing (E-SCIENCE '06), p. 79, 2006.
- [3] Dr. D.I. George Amalarethnam, P. Muthulakshmi, "An Overview of the Scheduling Policies and Algorithms in Grid Computing," International Journal of Research and Reviews in Computer Science (IJRRCS), Vol. 2, No. 2, April 2011.
- [4] Srikumar Venugopal, Xingchen Chu, and Rajkumar Buyya, "A Negotiation Mechanism for Advance

- Resource Reservations using the Alternate Offers Protocol,” Grid Computing and Distributed Systems (GRIDS) Laboratory, Department of Computer Science and Software Engineering, 2008.
- [5] Anthony Sulistio and Rajkumar Buyya, “A Grid Simulation Infrastructure Supporting Advance Reservation,” GRIDS Laboratory and NICTA Victoria Laboratory, Department of Computer Science and Software Engineering, 2004.
- [6] H. Mohamed and D. Epema, “Koala A Co-Allocating Grid Scheduler, Concurrency and Computation,” Practice and Experience, vol. 20, no. 16, pp. 1851-1876, 2008.
- [7] H.H. Mohamed and D.H.J. Epema, “An Evaluation of the Close-to-Files Processor and Data Co-Allocation Policy in Multiclusters,” Proc. IEEE Int’l Conf. Cluster Computing (CLUSTER ’04), pp. 287-298, 2004.
- [8] Kwang Mong Sim and Benyun Shi, “Concurrent Negotiation and Coordination for Grid Resource Coallocation,” IEEE Transactions on systems, man and cybernetics-Part B: cybernetics, vol. 40, NO. 3, June 2010.
- [9] A.C. Sodan, C. Doshi, L. Barsanti, and D. Taylor, “GangScheduling and Adaptive Resource Allocation to Mitigate Advance Reservation Impact,” Proc. Int’l Symp. Cluster Computing and the Grid (CCGRID), pp. 649-653, 2006.
- [10] J. Sauer, A.M. Meystel: “Modeling and Solving Multi-Site Scheduling Problems,” Planning in Intelligent Systems: Aspects, Motivations and Methods, pp. 281-299, Wiley, 2006.
- [11] Rue Min, Muthucumaru Maheswaran, “Scheduling Advance Reservations with Priorities in Grid Computing Systems,” Advanced Networking Research Laboratory, 2000.
- [12] Erik Elmroth and Johan Tordsson, “Grid Resource Brokering Algorithms Enabling Advance Reservations and Resource Selection Based on Performance Predictions,” Dept. of Computing Science and HPC2N, Nov 2006.
- [13] K. Czajkowski, I. Foster, and C. Kesselman: “Resource Co-Allocation in Computational Grids,” Proc. Eighth IEEE Int’l Symp. High Performance Distributed Computing, 1999.
- [14] A. Iosup, O. Sonmez, and D. Epema, “DGSim: Comparing Grid Resource Management Architectures through Trace-Based Simulation,” Proc. 14th Int’l Euro-Par Conf. Parallel Processing (Euro-Par’08), pp. 13-25, 2008.
- [15] Claris Castillo, George N. Rouskas, Khaled Harfoush: “Efficient Resource Management using Advance Reservations for Heterogeneous Grids,” IEEE conference on parallel and distributed processing, 2008.
- [16] Shengwei YI, Zhichao WANG, Shilong MA, Zhanbin CHE, Yonggang HUANG, Xin CHEN, “An Effective Algorithm of Jobs Scheduling in Clusters,” Journal of Computational Information Systems, Oct 2010.
- [17] Kwang Mong Sim: “Grid Resource Negotiation: Survey and New Directions,” IEEE Transactions On Systems, Man, And Cybernetics, May 2010.
- [18] Luis Tomas, Carmen Carrion, Blanca Caminero, Agustin Caminero, “Meta-Scheduling in Advance using Red-Black Trees in Heterogeneous Grids,” IEEE Transactions on parallel and distributed Systems, 2010.
- [19] Claris Castillo, George N. Rouskas, Khaled Harfoush, “On the Design of Online Scheduling Algorithms for Advance Reservations and QoS in Grids,” IEEE Transactions on parallel and distributed systems, 2008.
- [20] Ponsy R. K. Sathia Bhama, S. Thamarai Selvi, “Schedule Based Processor Co allocation in Multiclusters for Scheduling in Grid,” European Journal of Scientific Research, 2011.