ISSN 2278-3091

International Journal of Advanced Trends in Computer Science and Engineering (IJATCSE), Vol.2 , No.3, Pages : 13-17 (2013)
Special Issue of ICCSIE 2013 - Held during 24 May, 2013 Bangalore, India

A Hybrid Algorithm For Moldable Jobs Scheduling in Heterogeneous Multi-Cluster System

Navroop Kaur[1], Amit Chhabra[2], Nancy[3], Bhuvnesh Kumar[4]
Department of Computer Science and Engineering
Guru Nanak Dev University, Amritsar (Punjab), India.
Email: [1]navonline@yahoo.co.in
[2]chabbra_amit78@yahoo.com
[3]nancy.k1307034@yahoo.co.in
[4]bhunesh.gsp@gmail.com

Abstract : In systems consisting of multiple clusters of processors, the processors can differ in the computing speed and number of processors both within and among the clusters. In this paper, we propose a scheduling technique that schedules moldable jobs in such a heterogeneous system. A unit called, Basic Processor Unit (BPU) is used to measure the computing speed of processors. The scheduling process integrates the techniques of job selection, site selection and processor selection into single algorithm with the objectives of improving mean response time and utilization in a heterogeneous multicluster system

Key words : Moldable, Multi-cluster, Resource Heterogeneity.

INTRODUCTION

A collection of computing resources (often formed from inexpensive Commodity-Off-The-Shelf (COTS) computers) that are interconnected through a network switch is called *cluster*. The geographically co-located clusters can be connected via an interconnection network to form a larger computational resource known as a *multi-cluster*. Multi-cluster systems added more computational power to the system because the jobs can be distributed among the available clusters. Though multi-clusters added such flexibility to the system but they also increased the complexity of effectively managing both computing and networking resources. Hence there is a need of job scheduling at multi-cluster level.

In order to make effective use of the multi-clusters, intelligent scheduling algorithms must be designed and implemented that not only caters to the specific needs of its users, but also seeks to optimize overall system performance. For effectively managing the resources of the multi-cluster system, the job scheduling algorithms must address three issues: (i) nature of job, (ii) feasibility, (iii) heterogeneity.

The jobs can be categorized into three types depending upon the nature of jobs, namely, rigid, moldable, and malleable. A *rigid* job is one that requires a fixed number of processors. In *moldable* jobs, the number of processors can vary and are adapted only at the start of the execution. The number of processors for both rigid and moldable jobs cannot be changed during runtime. If the number of processors assigned to the job can be changed during their runtime, the job is termed as *malleable*. The work here is aimed at moldable jobs.

The local users join the multi-cluster only if there is a performance improvement for all the participating sites. Performance can be measured in the terms of job response time or average waiting time. A scheduling algorithm is said to be feasible if no participating sites' average response time for their jobs get worse after joining the multicluster system (rather it should improve).

Another important factor in context of job scheduling in a multicluster system is resource heterogeneity. In a real world, the multicluster normally consists of clusters which can differ in the computing speed and the number of processors at each site as well as among the clusters. Heterogeneity puts a challenge on designing efficient scheduling algorithms. This paper proposes a scheduling policy based on the moldable property of parallel jobs for heterogeneous multicluster system.

RELATED WORK

Parallel job scheduling has been an active field of research for a long time. Substantial amount of work has been done on various platforms viz. shared memory systems, distributed memory multiprocessors, clusters, multi-clusters and grid.

Extensive work has been done on single cluster system. Most of the research has been done in scheduling techniques, scheduler evaluation, workload modelling and fairness. Dandamudi et al. [4] proposed a two level space-sharing policy for heterogeneous cluster systems. The policy is based on the concept of Basic Processor Units (BPUs) to compute the partition size.

Also, substantial amount of work has been done on job scheduling in a multi-cluster system. Many scheduling techniques have been developed and their performance has been evaluated.

Bucur [3] used Distributed ASCI Supercomputer (DAS) (multi-cluster) system in her research and studied the performance of co-allocation. Co-allocation is a technique in which a parallel job is broken into components and each component can be processed in a different cluster. For

ISSN 2278-3091

International Journal of Advanced Trends in Computer Science and Engineering (IJATCSE), Vol.2 , No.3, Pages : 13-17 (2013)
Special Issue of ICCSIE 2013 - Held during 24 May, 2013 Bangalore, India

example, suppose that a job is waiting in a cluster's ready queue. This job may require more nodes than are presently available on its particular cluster, but collectively there may be enough available nodes elsewhere in the multi-cluster to accommodate the job. Co-allocation allows jobs to be mapped across cluster boundaries. In doing so, resource fragmentation is reduced and system utilization is increased.

Bucur and Epema [2] studied the effect of system configurations on performance of co-allocation. Their observations show that in addition to the scheduling techniques, architectural and placement considerations also improve the performance of co-allocation.

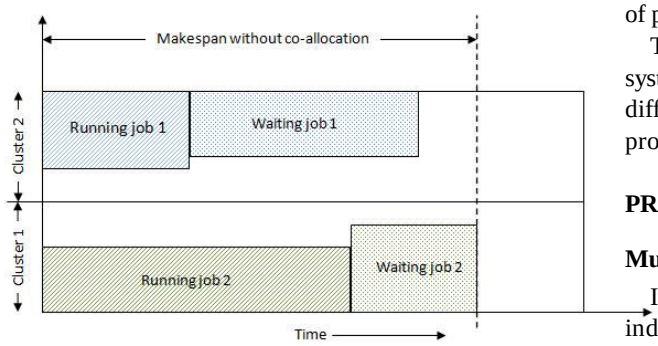

Fig 1: Scheduling without co-allocation

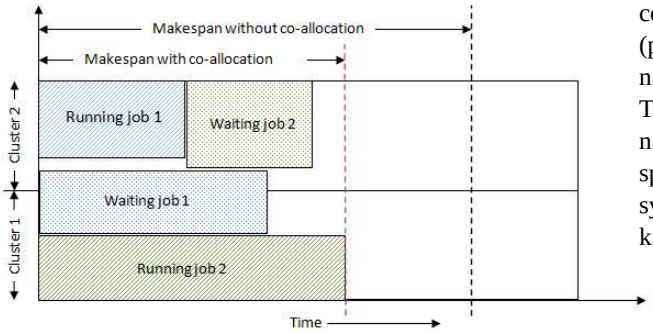

Fig 2: Scheduling with co-allocation

Jones [11] used the multi-cluster system at Clemson University in his research. He observed that inter-cluster communication pose challenges for co-allocation. But with good techniques such as in [6] and [11] the effect of communication can be minimized

John Ngubiri [12][13] evaluated the performance of co-allocation in multi-cluster system and observed that (i) co-allocation is viable if the execution time penalty caused is low; (ii) due to possible heterogeneous communication pattern, co-allocation may not be as viable.

The work by Bucur et al. is focused in scheduling rigid jobs in a multi-cluster system. Huang [10], in his work, used moldable property of parallel jobs for scheduling in a grid. Multi-cluster systems may be looked at as a small version of the conventional grid. Multi-clusters are smaller in size than a grid and the clusters in multi-cluster systems are connected by a more reliable and dedicated backbone other than the Internet in the conventional grid. So the techniques of grid can be easily applied on multi-cluster systems.

England and Weissman [5] analyzed the costs and benefits of load sharing of parallel jobs in both homogeneous and heterogeneous grids. The heterogeneous grid differs only in capacity and workload characteristics and not in the computing speeds at different sites. Huang and Chang [8] showed that the best site selection policy for such a heterogeneous grid is *best-fit*. In this policy a particular site is chosen on which a job will leave the least number of free processors if it is allocated to that site.

Later, Huang [9][10] studied the load sharing policies in a heterogeneous grid in which nodes on different sites may have different computing speeds but the nodes on same site have same speed. In this paper, Huang developed adaptive processor allocation policies based on the moldable property of parallel jobs for heterogeneous computational grids.

This paper is focused on the heterogeneous multi-cluster system that not only differs in the computing speed at different sites but also in the computing speed and number of processors at the same site.

PROPOSED POLICY

Multi-Cluster Model

In the model, the multi-cluster system consists of several independent clusters (sites). Each participating site is a heterogeneous parallel computer. Each site is heterogeneous in the sense that the number of processors as well as the computing speed of each processor may vary. The nodes (processors) are linked together using fast interconnection network that do not favour any communication pattern [7]. This means a parallel job can be allocated on any subset of nodes in a site. The parallel computer system uses space-sharing and run the jobs in an exclusive fashion. The system deals with an on-line scheduling problem without any knowledge of future job submissions.

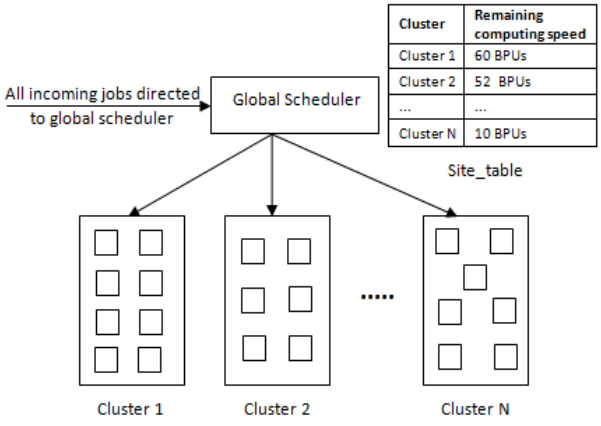

Fig 3: Multi-cluster System

The processors in the system are rated in terms of Basic Processor Unit (BPU) [4] to take the processor heterogeneity into consideration. The physical processor with the lowest processing capacity in the system is considered to represent 1 BPU. The ratings of other processors are expressed in terms of BPUs. The system also maintains a table (called site_table in the pseudo code). This table contains every site arranged in

ISSN 2278-3091

International Journal of Advanced Trends in Computer Science and Engineering (IJATCSE), Vol.2 , No.3, Pages : 13-17 (2013)
Special Issue of ICCSIE 2013 - Held during 24 May, 2013 Bangalore, India

the decreasing order of its computing speed. The computing speed is calculated by taking the average number of unallocated BPUs at each site. The table is updated after every job allocation and job completion.

Scheduling Process

When a moldable job is submitted to the system, it is first handled by local scheduler. If local site do not have enough processing capacity, it sends the job to global scheduler. At global scheduler, the scheduling is done in three phases: (i) Job Selection, (ii) Site Selection, and (iii) Processor Selection.

Algorithm Schedule_a_job
For each parallel job J_i to be scheduled do
Perform Job Selection using FCFS algorithm
Let x_i be the number of BPUs needed by J_i
Perform Site Selection using site_select algorithm
For the selected site S_j do
Processor Selection using processor_select(S_j , J_i , x_i) algorithm
End for
Reconstruct site_table using build_table algorithm
End for
For each job leaving the system do
Reconstruct site_table using build_table
End for

Fig 3: Schedule_a_job Algorithm

Algorithm site_select
Read the first row of the table site_table
Let S_j be the site in the first row of the table
return(S_j)

Fig 4: Site_Select Algorithm

Algorithm processor_select(S_j , J_i , x_i)
Partition the processors at S_j such that the partition size(in terms of BPUs) >= x_i
For all such partitions do
Choose a partition with partition size having least difference from x_i
If two or more partitions have the same size
Choose the partition with fewest processors
End if
End for

Fig 5: processor_select Algorithm

Algorithm Build_table
//Let n_j represent the number of unallocated BPUs at site S_j
For each job J_i leaving the system do
For the site S_j on which J_i was running using x_i BPUs do
$n_j = n_j + x_i$
End for
For each job J_i allocated x_i BPUs at site S_j do
$n_j = n_j - x_i$
End for
Recalculate the average number of BPUs at S_j
Arrange the entries in the site_table in non-increasing order of the average BPUs.

Fig 6: Build_table Algorithm

When a job is submitted to the local site, it is directed to the global scheduler and is added to the end of the queue at global scheduler. Job Selection policy selects the job from the queue to be sent for scheduling. Initially, we assume that the selection is done using the fairest policy i.e. First Come First Serve (FCFS) Policy.

During the second phase, the site is to be selected on which the job must be run. The fastest-first policy is assumed for site-selection. According to this policy, the site with fastest computing power is selected for computation. The computing power is calculated in terms of average unallocated BPUs at each site. Thus, the first site in the table site_table is selected for execution of the current job. If no single site has enough computing power then job is coallocated.

When the job and the site has been selected, the next step is to select the appropriate processors at the selected site. The processors are partitioned at the selected site in such a way that the partition size (in terms of BPUs) is nearly equal to the demanded computing power (in terms of BPUs). This corresponds to best-fit policy. If two or more partitions have same size then choose the partition with fewest number of processors (fastest-first policy). Thus, the processor selection uses best-fit with fastest-first policy.

PERFORMANCE EVALUATION

The performance of an algorithm depends on the nature of jobs arriving and on the policy used to schedule an algorithm. Various parameters are used to evaluate the performance using various workloads.

Workload Characteristics

Jobs can arrive in a system in various fashions. To simulate the job arrivals various job arrival distributions can be used. The various job arrival distributions are exponential distribution, poisson distribution and hyperexponential distribution.
While simulating our algorithm, we are using hyper exponential distributions for both job arrival time and service times. The simulation is to be done using various workloads depending upon different Arrival Rate, Cofficient of Variation (CV) of Inter-Arrival times and Coefficient of Variation of Service times and service rates.

ISSN 2278-3091

International Journal of Advanced Trends in Computer Science and Engineering (IJATCSE), Vol.2 , No.3, Pages : 13-17 (2013)
Special Issue of ICCSIE 2013 - Held during 24 May, 2013 Bangalore, India

Performance Parameters

The performance of algorithms is usually measured using performance evaluation parameters such as mean response time, average utilization, average waiting time, etc.
In our simulation, we use two main parameters viz. mean response time and mean waiting time of the arriving jobs.

Performance Comparison

This section present the simulation results of the experiments.

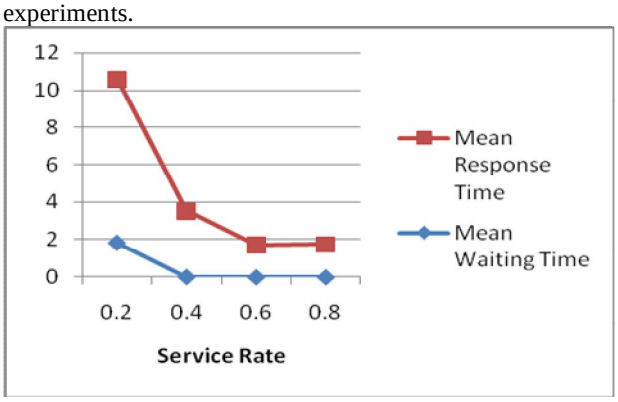

Fig 7: Performance Sensitivity of Mean Response Time and Mean Waiting Time to Service Rate

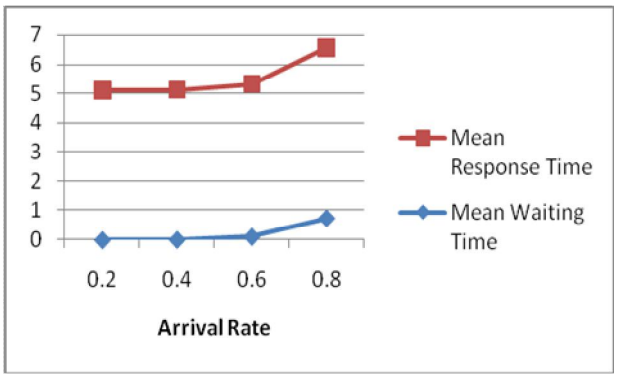

Fig 8: Performance Sensitivity of Mean Response Time and Mean Waiting Time to Arrival Rate

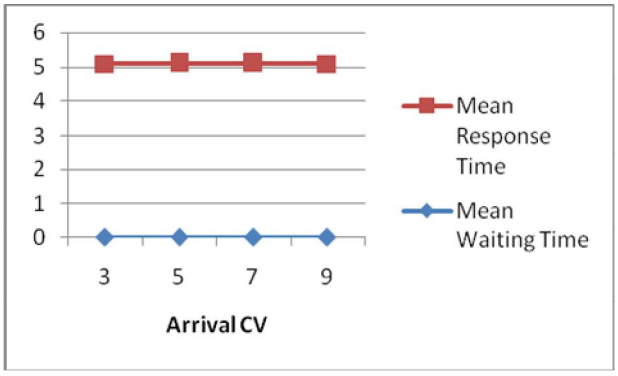

Fig 9: Performance Sensitivity of Mean Response Time and Mean Waiting Time to Arrival CV

The results show the performance sensitivity of various performance evaluation parameters towards job arrival process and job service times.

The results show that the mean response time is very sensitive to the variation of service rate and coefficient of variation of inter-service times. The mean response time reduces with the increase in service rate and service CV.
While the waiting time depends on the policy used for scheduling more than the arrival and service rates. The simulation results show that the mean waiting time of the incoming jobs is very low. This will further increase the service rate of the incoming jobs.
The proposed policy suggests that a job can be allocated variable number of BPUs for its execution. The number of BPUs to be allocated to the job is calculated at run time and thus, making this policy highly suitable for moldable jobs in a multicluster system.

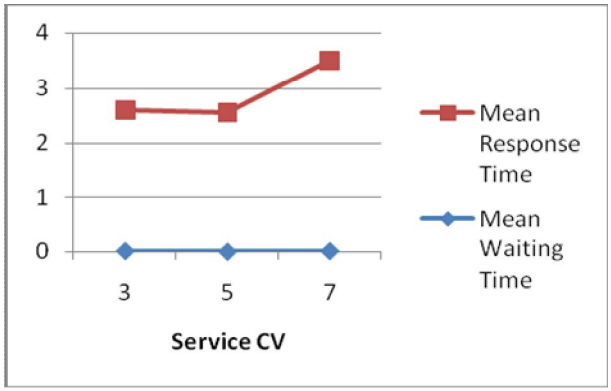

Fig 10: Performance Sensitivity of Mean Response Time and Mean Waiting Time to Service CV

CONCLUSION

We have proposed a new policy for scheduling moldable jobs in a multi-cluster system. This policy takes into consideration the resource heterogeneity of a multi-cluster system. The heterogeneity of the system is considered both within and among the clusters in terms of computing speed and number of processors. The performance evaluation of proposed policy shows that it considerably reduces the mean waiting time of the incoming jobs. The mean response is also reduced at higher values of service rates.

REFERENCES

(Periodical style)

[1] K.C. Huang, P.C. Shih, Y.C. Chung," Effective Processor Allocation for Moldable Jobs with Application Speedup Model. Smart Innovation", Systems and Technologies Volume 21, pp 563-572, 2013.

[2] A.I.D. Bucur, D.H.J. Epema, "An Evaluation of Processor Co-Allocation for Different System Configurations and Job Structures", *In Proceedings of the 14th Symposium on Computer Architecture and High Performance Computing*, 195– 203, 2002.

[3] A.I.D. Bucur, "Performance analysis of processor co-allocation in multicluster systems", *PhD Thesis, Delft University of Technology, Delft, The Netherlands*, 2004.

[4] S.P. Dandamudi, Z. Zhou, "Performance of adaptive space-sharing policies in dedicated heterogeneous cluster systems", Centre for Parallel and Distributed Computing, School of Computer Science, Carleton University, Ottawa, Ont., Canada K1S 5B6. Future Generation Computer Systems, 895-906. DOI = 10.1016/j.future, 2004.

[5] D. England, J. B. Weissman, "Costs and Benefits of Load Sharing in the Computational Grid", *In Job Scheduling Strategies for Parallel Processing*, 160-175, 2005..

[6] D.H.J. Epema, O.O. Sonmez, H. Mohamed, "On the Benefit of Processor Coallocation in Multicluster Grid Systems", *IEEE transactions on parallel and distributed systems, vol. 21, no. 6 (*June 2010), 778-789, 2010.

[7] D. Feitelson, L. Rudolph, "Parallel job scheduling: Issues and approaches", In *Job Scheduling Strategies for Parallel Processing*, 1-18, 1995.

[8] K.-C. Huang, H.-Y. Chang, "An Integrated Processor Allocation and Job Scheduling Approach to Workload Management on Computing Grid", *In the Proceedings of the 2006 International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'06), Las Vegas, USA, 703-709*, 2006.

[9] K.C. Huang, P.C. Shih, Y.C. Chung." Towards feasible and effective load sharing in a heterogeneous computational grid", *In the Proceedings of the second international conference on Advances in grid and pervasive computing, ed, 229-240*, 2007.

[10] K.C. Huang, P.C. Shih, Y.C. Chung, "Adaptive Processor Allocation for Moldable Jobs in Computational Grid", At *10th International Journal of Grid and High Performance Computing, 1(1), (*March 2009), 10-21, 2009.

[11] W.L. Jones, "Improving Parallel Job Scheduling Performance In Multi-Clusters Through Selective Job Co-Allocation", *A P.hd Dissertation presented to the Graduate School of Clemson University*, December 2005.

[12] J. Ngubiri, V.Mario, " Using the Greedy Approach to Schedule Jobs in a Multi-cluster System", In Proceeding of the International Conference on Parallel and Distributed Processing Techniques and Applications, 663 – 668, 2006.

[13] J. Ngubiri, V. Mario, "Group-wise Performance Evaluation of Processor Co-allocation in Multi-cluster Systems", *In Proceedings of the 13th Workshop on Job Scheduling Strategies for Parallel Processing*, 24 – 36, 2007.