Global Journal of Computer Science and Technology
Volume 12 Issue 5 Version 1.0 March 2012
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals Inc. (USA)
Online ISSN: 0975-4172 & Print ISSN: 0975-4350

Handling of Congestion in Cluster Computing Environment Using Mobile Agent Approach

By P.K. Suri & Sumit Mittal
M.M. University, Mullana, Ambala, Haryana

Abstract - Computer networks have experienced an explosive growth over the past few years and with that growth have come severe congestion problems. Congestion must be prevented in order to maintain good network performance. In this paper, we proposed a cluster based framework to control congestion over network using mobile agent. The cluster implementation involves the designing of a server which manages the configuring, resetting of cluster. Our framework handles - the generation of application mobile code, its distribution to appropriate client, efficient handling of results, so generated and communicated by a number of client nodes and recording of execution time of application. The client node receives and executes the mobile code that defines the distributed job submitted by server and replies the results back. We have also the analyzed the performance of the developed system emphasizing the tradeoff between communication and computation overhead. The effectiveness of proposed framework is analyzed using JDK 1.5.

Keywords : Cluster Computing, Mobile Agent, Congestion.

GJCST Classification: C.2.5

HANDLING OF CONGESTION IN CLUSTER COMPUTING ENVIRONMENT USING MOBILE AGENT APPROACH

Handling of Congestion in Cluster Computing Environment Using Mobile Agent Approach

P.K. Suri [α] & Sumit Mittal [σ]

Abstract - Computer networks have experienced an explosive growth over the past few years and with that growth have come severe congestion problems. Congestion must be prevented in order to maintain good network performance. In this paper, we proposed a cluster based framework to control congestion over network using mobile agent. The cluster implementation involves the designing of a server which manages the configuring, resetting of cluster. Our framework handles - the generation of application mobile code, its distribution to appropriate client, efficient handling of results, so generated and communicated by a number of client nodes and recording of execution time of application. The client node receives and executes the mobile code that defines the distributed job submitted by server and replies the results back. We have also the analyzed the performance of the developed system emphasizing the tradeoff between communication and computation overhead. The effectiveness of proposed framework is analyzed using JDK 1.5.

Keywords : Cluster Computing, Mobile Agent, Congestion.

I. Introduction

Mobile agent technology plays a vital role in the management of cluster computing. Mobile agents are autonomous programs that travel from computer to computer under their own control. Their abilities to cope with system heterogeneity and to deploy user-customized procedures at remote sites are well suited for cluster computing environments[12]. By interacting with a remote host after migrating to it, an agent can perform complex operations on remote data without transferring them, because the agent can carry the application logic to where it is needed.

Cluster computing harnesses the combined computing power of multiple microprocessors in a parallel configuration. Cluster computers are a set of commodity PC's dedicated to a network designed to capture their cumulative processing power for running parallel-processing applications. Clustere computers are specifically designed to take large programs and sets of data and subdivide them into component parts, thereby allowing the individual nodes of the cluster to process their own individual chunks of the program.

Author [α] : Dean (R&D) & Chairman (CSE/IT/MCA), HCTM Technical Campus, Kaithal, Haryana, 136 027, India
E-mail : pksuritf25@yahoo.com
Author [σ] : M.M. Institute of Computer Technology & Business Management, M.M. University, Mullana, Ambala, Haryana, 133 207, India E-mail : sumit_amb@yahoo.com

Literature Survey : A number of research efforts in the area of improving the performance of distributed applications in a cluster computing environment have emerged[2][7][8]. Martin, Vahdat, Culler and Anderson[9] revealed that the communication latency, overhead, and bandwidth can be independently varied to observe the effects on a wide range of applications. They showed that applications demonstrate strong sensitivity to overhead, slowing down, when overhead is increased. Lange, D.B[10] discussed the impact of cluster size and underlying network on the performance of distributed applications.

Our research effort (i) evaluates the performance of application as a function of problem size, network parameters and cluster size (ii) provides features for monitoring of cluster. The developed MCLUSTER model concentrates on reducing the communication overhead by using the mobility of code and bridging all system characteristics.

II. Architecture of Cluster Computing

There are many cluster configurations, but a simple architecture is shown in the Figure 1.

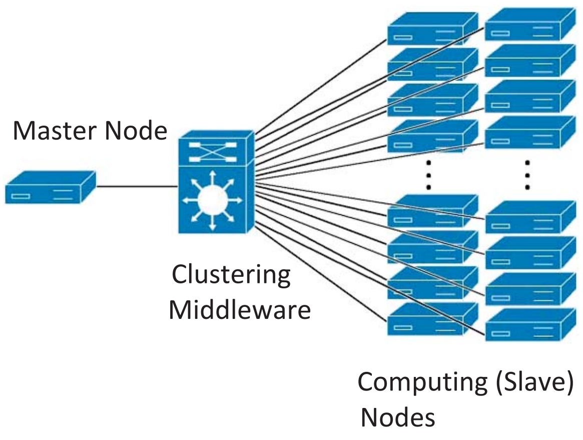

Figure 1. Cluster Computing Architecture

As shown in Figure 1, node participation in the cluster falls into master or head node and computing or slave nodes. The master node is the unique server in cluster systems. It is responsible for running the file system and also serves as the key system for clustering middleware to route processes, duties and monitor the status of each slave node. A slave node within a cluster provides the cluster a computing and data storage capability. These nodes are derived from fully operational, standalone computers that are typically marketed as desktop or server systems that are off-the-shelf commodity systems.

III. Design of Cluster Computing

The framework consists personal computers, high speed communication network, sequential and parallel applications as shown in Figure 2.

PC's are connected to the network using Ethernet Network Interface Card. Cluster middleware is implemented in JAVA so that middleware can provide the single system image of the cluster to any computer with different OS platforms, once the JAVA virtual machine is installed. JVM makes it easier to implement, migrate and execute the mobile code at remote computer in the cluster. The user is guided through the creation and management of cluster via GUI. It frees the user from identifying the network topology of the framework of cluster. The framework has been designed in such a way that incremental changes to it can easily enhance the generality and usability of cluster.

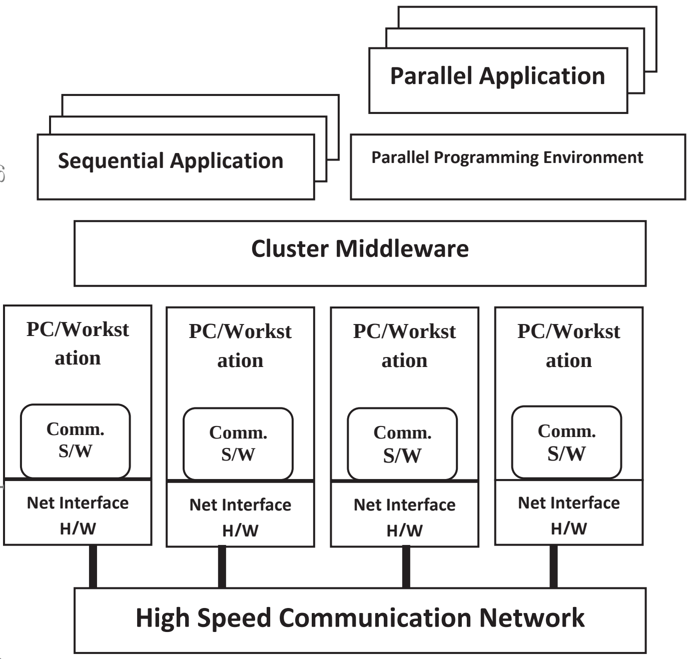

Figure 2. Cluster Computing Framework

IV. Cluster Middleware Architecture

Cluster middleware allows the user to develop & process the parallel applications on clusters and achieve good performance. In cluster configuration, there are two types of nodes: a server node named MCLUSTER, client nodes as shown in the Figure 3.

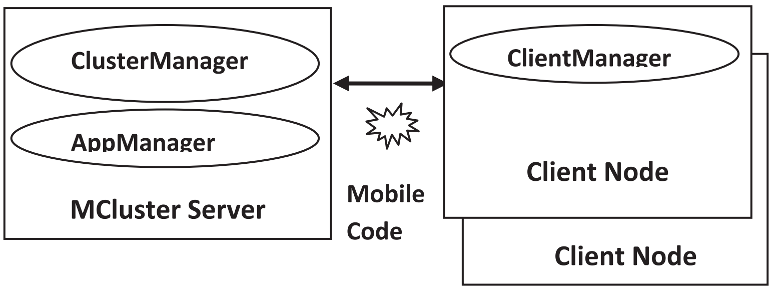

Fig. 3 Cluster middleware

MCLUSTER and client nodes are interacting with each other through message passing and any parallel application is executed using mobile code which contains input data/application code.

V. ClusterManager

ClusterManager initiates the cluster and controls all nodes of the cluster. ClusterManager periodically broadcast invitation packets to all nodes in the network as shown in the Figure 4. All those nodes on which ClientManager is activated and are not being part of the cluster, will display a frame on the user screen after receiving the invitation packet. If client node accepts the invitation, then, IP address of that node will be automatically communicated to MCLUSTER as show in Figure 4. MCLUSTER maintains a database of all these addresses.

VI. AppManager

AppManager coordinates all functions related to application description, distribution and execution. AppManager provides a GUI from which user can easily select a particular application, provides the input data and specify the parameters such as number of nodes to be used for the execution of application. Once the application is selected, then the AppManager will divides the data range into the number of nodes and distribute the mobile code to the selected client nodes as shown in Figure 4.

VII. ClientManager

ClientManager listens to the requests from MCLUSTER related to cluster membership, application execution and service those requests. In lieu of invitation packet, ClientManager running on a node will communicate its IP address to the server. On receiving the mobile code from server it load and execute the mobile code and reply the results back.

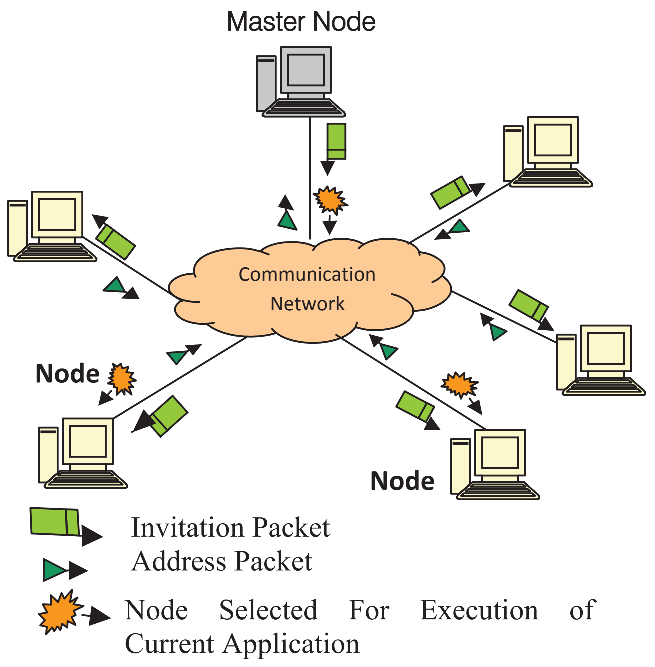

Figure 4 Communication between Master node and client node

VIII. Implementation of Our Framework

In our research effort, series application is selected for validating the framework. It basically involves generating a sequence of numbers between an initial value & a final value. MCLUSTER begins a timer before the execution of application and stops it after the collection of entire series from client nodes as shown in the figure 5. The effect of number of client nodes and data range on the execution time of application is represented by Figure 6 & 7.

The performance of cluster, during the execution of distributed applications not having significant amount of data (such as between 1-20000) deteriorate as indicated by Overhead point in Figure 6. At these data ranges the communication overhead between MCLUSTER Server and Client nodes overwhelm the advantage of distributed processing power obtained from client nodes. When data range is extensive then the time consumed by an application so executed in a distributed manner is several times smaller than the execution time of the same application processed on a single node. The MCLUSTER model is designed in such a way that even for a large problem size (such as between 1-5000000) and average cluster size (up to N=11) performance of distributed application will not deteriorate as shown in Figure 7.

Communication Overhead Analysis : When the scalability of cluster increases, the communication links near the server are congested due to large transmission of data, thereby degrading the performance of cluster. Communication overhead includes the overhead due to exchange of data between nodes and message delay caused by network congestion. Figure 8 is represented by the result so generated after execution of the application Series for a data range 1-100000 on a cluster of 11 client nodes.

On the basis of assumption that external network load is low and constant, the communication overhead for MCLUSTER model can be parameterized as a simple linear function of number of bytes transmitted and number of client nodes selected as represented in Figure 8.

$$\text{Coverhead} = T + C * N$$

Where = Startup time involving Partitioning time, Time spent during selection of client nodes.
C = Cost per byte transmitted.
N = Number of nodes selected for a particular execution.

IX. Conclusion

Cluster computing undoubtedly is gaining importance as a substitute for very expensive parallel computers. As depicted in our research work, Cluster computing by appropriately combining the computational processing power of autonomous computers can significantly improve the performance of distributed applications. Cluster Computing on the other hand can decline the performance of a problem if a proper check is not applied at the problem size as well as cluster size. Since the improper values of these two parameters will lead to the network congestion thereby resulting in the overwhelming of computation load by communication load. However this Network Congestion problem can be easily tackled by using the concept of Mobile Agent.

References Références Referencias

1. Baker, M. and R. Buya, 1999. Cluster computing: The commodity supercomputer. Software-Prentice, 29: 551-576.
2. Cluster Computing Overview, http://www.cs.wayne.edu/~haj/load/survey/overview.html.
3. Becker, D. and P. Merkey. The Beowulf Project. http://www.beowulf.org.
4. Message Passing Interface, MPI Forum-http://www.mpi-forum.org.
5. Flynn, M.J. Very High Speed Computing Systems. Proceedings. IEEE, 54: 1901.
6. Geist,A. and J. Schwidder, 1999. Managing multiple multi-user PC clusters. J. Parallel and Distributed Computing.
7. Cheung, L. and A.P. Reeves, 1992. High performance computing on a cluster of workstations. Proceedings of First Symposium on High - Performance Distributed Computing.

Global Journal of Computer Science and Technology Volume XII Issue V Version I March 2012

8. Jon, B.W. and A.S. Grimshaw, 1994. Network partitioning of data parallel computations. Proc. Third International Symposium. High Performance Distributed Computing (HPDC '94), April 2-5, 1994, San Francisco, CA, USA. IEEE Computer Society.
9. Lemeire, J. and E. Dirkx, 2002. Causes of blocking overhead in message-passing programs.10th Euro PVM/MPI 2002 Conference, Venice, Italy.
10. Martin, R., A. Vahdat, D. Culler and T. Anderson, 1997. Effects of communication latency,overhead and bandwidth in a cluster architecture. Proceedings 24th Annual International. Symposium Computer Architecture (ISCA), New York, USA, pp: 85-97.11. Lange, D.B. Mobile objects and Mobile Agents: The Future of Distributed Computing? General Magic, Inc. California.
11. Walker, B. and D. Steel, 1999. Implementing a full single system image unixware cluster: Middleware vs. underware. Proceedings: International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'99), Las Vegas, USA.
12. Ichiro Satoh. Configurable Network Processing for Mobile Agents on the Internet. Cluster computing, The Journal of Networks, Software Tools and Applications), vol. 7, no.1, pp.73-83, Kluwer, 2004.

Data Range	No. of Nodes	Time (Milliseconds)
1-10000	1	1718
	2	1469
	3	1282
1-20000	1	3000
	2	1734
	3	1890
1-50000	1	7016
	2	3734
	3	3125
1-100000	1	13687
	2	7015
	3	5375

Figure 5. Statistical Data of Application : Series

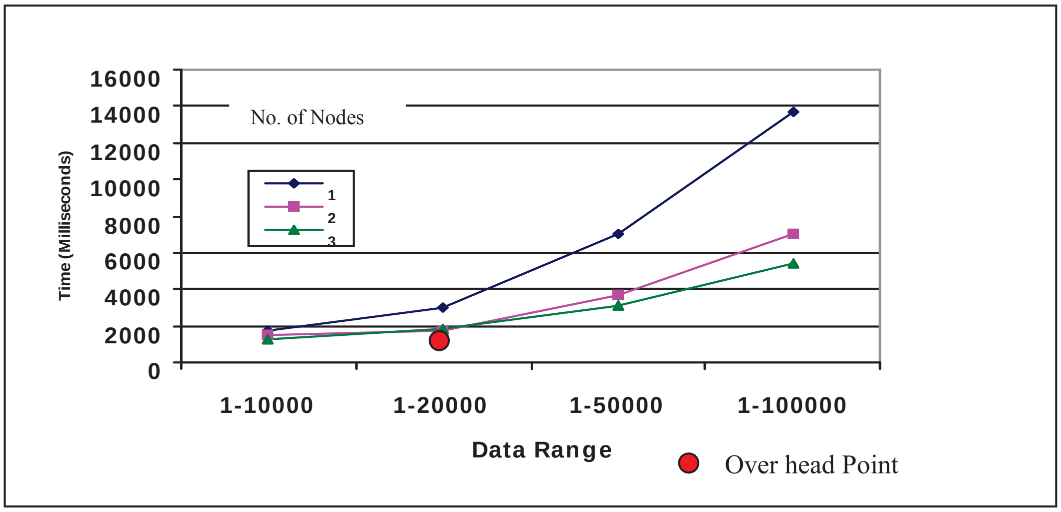

Figure 6. Effect of data range and number of nodes on execution time

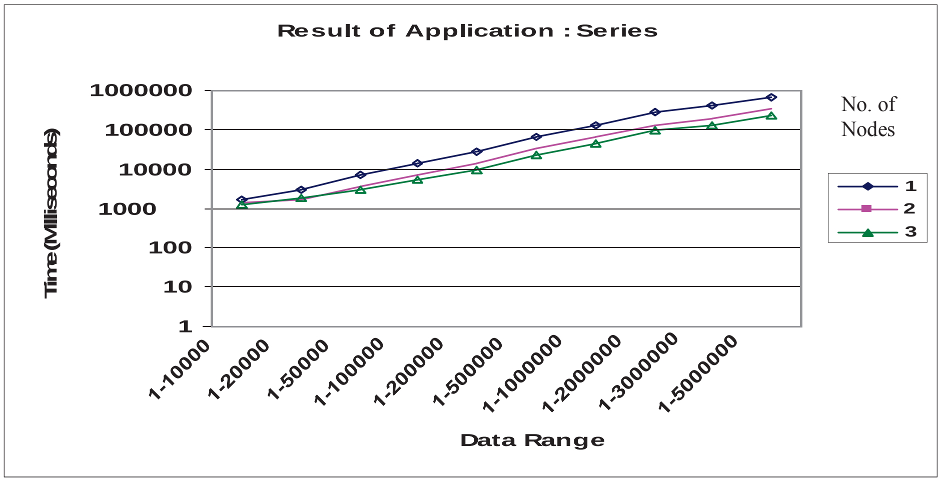

Figure 7. Effect of data range on execution time of application series

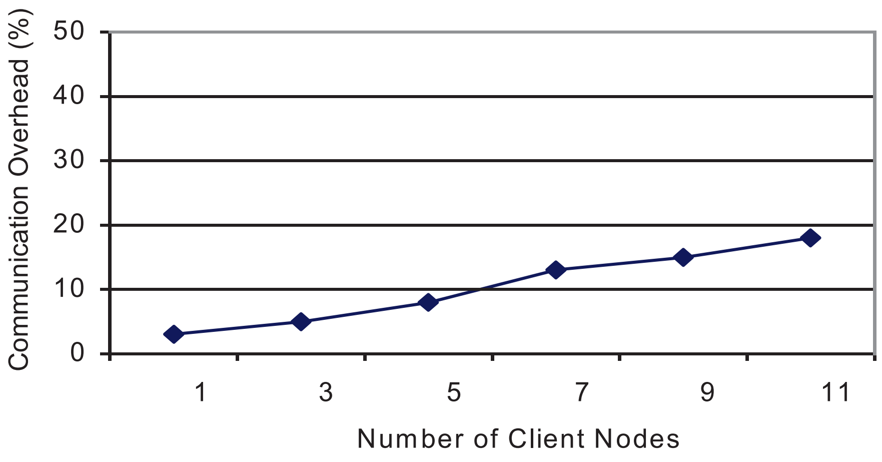

Figure 8. Communication overhead vs. cluster size

This page is intentionally left blank