Implementing Distributed Computing System For Parallel Processing

Ms. Pallavi S. Shendekar, Mr. Vijay S. Gulhane, Mr. Amit R. Gadekar

Abstract— In this paper we are using threshold algorithm for performance improvement, this is the part of the Load balancing algorithm, for implementing distributed computing system for parallel processing. Distributed computing is a form of parallel computing, but parallel computing is most commonly used to describe program parts running concurrently on multiple processors in the same computer. Both types of processing require dividing a program into parts that can run simultaneously, in distributed computing a program is dividing into parts that run simultaneously on multiple computers communicating over a network. In order to ensure good overall performance, Load balancing algorithm tries to balance the total system load by transparently transferring the workload from heavily loaded nodes to lightly loaded nodes.

Index Terms— Database centric, distributed computing, distributed file system, Load balancing algorithm, parallel computing, Threshold

1 INTRODUCTION

THE word distributed in terms such as "distributed system", "distributed programming", and "distributed algorithm" originally referred to computer networks where individual computers were physically distributed within some geographical area. The system may be characterized both as "parallel" and "distributed"; the processors in a typical distributed system run concurrently in parallel. Distributed computing deals with hardware and software systems containing more than one processing element or storage element, concurrent processes, or multiple programs, running under a loosely or tightly controlled administration. In distributed computing a program is divide into parts that run simultaneously on multiple computers communicating over a network. In parallel computing, all processors may have access to a shared memory to exchange information between processors whereas in distributed computing, each processor has its own private memory (distributed memory). Information is exchanged by passing messages between the processors. The application of several processors to a single task is an older idea with a relatively large literature. The advent of very large-scale integrated technology has made testing the idea realistic, and the fact that single processor systems are impending their maximum performance level has made it crucial. We shall show, however, that victorious use of parallel processing imposes rigorous performance necessities on algorithms, software, and architecture.

To estimate the speedup of a tightly coupled system on a single application, we use a model of parallel computation introduced by Ware. We define a as the fraction of work in the application that can be processed in parallel. Then we make a simplifying assumption of a two-state machine; that is, at any instant either all processors are operating or only one processor is operating. Consider the condition user having 10000 document to process and each having large data in this case project need to employee the system which will transfer processing over the network system and save output on the server or main system. So the implemented system will aimed at parallel processing of provided task.

In Distributed Computing approach, it is followed to assign a job to a processor if it is idle. The focus is now on how to optimize re-sources to decrease the energy consumption by volumes of computing equipments to deal with green and sustainability issues. Various hardware and software architectures are used for distributed computing. At a lower level, it is necessary to interconnect multiple CPUs with some sort of network, regardless of whether that network is printed onto a circuit board or made up of loosely-coupled devices and cables. At a higher level, it is necessary to interconnect processes running on those CPUs with some sort of communication system.

2 LITERATURE SURVEY

Various algorithm and models have been proposed, mostly heuristic in nature, as the optimal solution often requires future knowledge and is computationally intensive. The most widely approach for studying DLB algorithms is analytic modeling and simulation. For analytic modeling, the computer system is modeled as a queuing network with job arrivals and their resource consumptions following certain probabilistic patterns. Queuing network solution techniques are used to compute performance measures [2],[7],[8],[9]. Due to limitations of the solution techniques, simulation is often resorted to for approximate solutions [4],[5]. Some of the-source-initiated DLB algorithms are by Eager [6],[7],[8].

Various hardware and software architectures are used for distributed computing. At a lower level, it is necessary to inter-

- *Ms. Pallavi S. Shendekar is currently pursuing masters degree in information technology in Amravati University, India, E-mail: pshendekar@yahoo.com.*
- *Mr. Vijay S. Gulhane is currently woringas a assistant professor in* Department of CS*E at Sipna College Of Engineering Technology, Amravati.* He is working towards his Phd in Computer Science and Engineering *from Amravati University. E-mail:* v_gulhane@rediffmail.com
- *Mr. Amit R. Gadekar is currently woringas a assistant professor in* Department of CS*E at DES'sCOET Dhamangaon [Rly], Amravati, India, E-mail:amit.gadekar1@gmail.com*

connect multiple CPUs with some sort of network, regardless of whether that network is printed onto a circuit board or made up of loosely coupled devices and cables. At a higher level it is necessary to interconnect processes running on those CPUs with some sort of communication system.

2.1 Load Balancing

Load balancing is the processes of improving the performance of a parallel and distributed system through are distribution of load among the processors [3, 4]. It is a process of reassigning the total load to the individual nodes of the collective system to make resource utilization effective and to improve the response time of the job, simultaneously removing a condition in which some of the nodes are over loaded while some others are under loaded. A load balancing algorithm which is dynamic in nature does not consider the previous state or behavior of the system, that is, it depends on the present behavior of the system. The important things to consider while developing such algorithm are : estimation of load, comparison of load, stability of different system, performance of system, interaction between the nodes, nature of work to be transferred, selecting of nodes and many other ones [5] . This load considered can be in terms of CPU load, amount of memory used, delay or Network load.

2.2 Goals of Load Balancing

As given in [5], the goals of load balancing are:

1. To improve the performance substantially
2. To have a backup plan in case the system fails even partially
3. To maintain the system stability
4. To accommodate future modification in the system.

2.3 Types of Load Balancing Algorithms

All Depending on who initiated the process, load balancing algorithms can be of three categories as given in [5]:

1. **Sender Initiated:** If the load balancing algorithm is initialized by the sender
2. **Receiver Initiated:** If the load balancing algorithm is initiated by the receiver
3. **Symmetric:** It is the combination of both sender initiated and receiver initiated

Depending on the current state of the system, load balancing algorithms can be divided into 2 categories as given in [5]:

1. **Static:** It does not depend on the current state of the system. Prior knowledge of the system is needed
2. **Dynamic:** Decisions on load balancing are based on current state of the system. No prior knowledge is needed. So it is better than static approach.

3 System Architecture

As In the implemented architectures the main factors are the designing the distributed system and parallel processing through a specific problem domain. Here the problem domain is known and well defined, the environment in which the system run is also well defined. Basic aspect of distributed computing architecture is the method of communicating and coordinating work among concurrent processes. Through various message passing protocols, processes may communicate directly with one another, typically in a master/slave relationship. Alternatively, "database centric" architecture can enable distributed computing to be done without any form of direct inter-process communication, by utilizing a shared database.

3.1 Working Modules

Fig.1. shows the basic block diagram of the system architecture. Actual task processing needs a series of steps to be performed. These series of processes are simultaneously executed on different client machines. Basically here we are distributing the no. of files on t the network through the shared database.

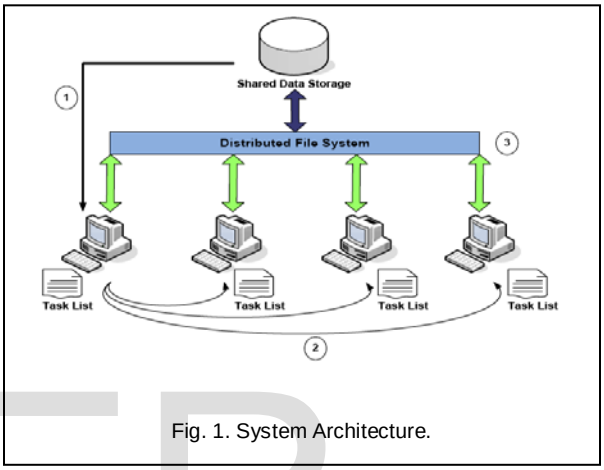

Fig. 1. System Architecture.

3.1.1 Master / Slave System

Parallel processing system built using server / client technology. Where master server act as a process manager system. In this system whenever network initialize or the first system user started in the network will check for active server in the network if no server running found then it will become host or master server and other were become slave system. This feature can be dynamic or static which means user can disable or enable this feature.

3.1.2 Task Assigning

Once the user provides input task on master server then master will analyze the task and divide it in to proper task and create its task list for client. Once all clients get the task list to process it will start processing. As implemented system will work on shared data storage it will reduce network processing and network traffic by removing data transfer processing.

Threshold algorithm

According to this algorithm, the processes are assigned immediately upon creation to hosts. Hosts for new processes are selected locally without sending remote messages. Each processor keeps a private copy of the system's load. The load of a processor can characterize by one of the three levels: Underloaded, Medium and Overloaded. Two threshold parameters tunder and tupper can be used to describe these levels.

Under loaded - load < tunder
Medium - tunder ≤ load ≤ tupper
Overloaded - load > tupper

Initially, all the processors are considered to be under loaded. When the load state of a processor exceeds a load level limit, then it sends messages regarding the new load state to all remote processors, regularly updating them as to the actual load state of the entire system. If the local state is not over-loaded then the process is allocated locally. Otherwise, a remote under loaded processor is selected, and if no such host exists, the process is also allocated locally. Thresholds algorithm have low inter process communication and a large number of local process allocations. The later decreases the overhead of remote process allocations and the overhead of remote memory accesses, which leads to improvement in performance. A disadvantage of the algorithm is that all processes are allocated locally when all remote processors are overloaded. A load on one overloaded processor can be much higher than on other overloaded processors, causing significant disturbance in load balancing, and increasing the execution time of an application.

3.1.3 Parallel Processing

After master distributes the task over the distributed network all clients will process simultaneously and send acknowledgement to the master server. Master server will also process the task and at the same time will check for the client processing status and monitor it on the screen.

3.1.4 Process Failure Detection System

The system will also manage failure of the client system at run time. Consider a condition if any of the client get failed due to any reason the remaining task should be processed by other system in the network. Master server will take care of this. It will continuously get acknowledgment from client time to time after each task completion. Once any client's connection get closed server will check work remain by specific client and then again divide this task and pass it to other client and client will process it.

3.1.5 Distributed File System

System sends data to client for processing but it will increase system overhead. So in this paper data to processed will kept on shared storage and accessed using distributed file system so that network protocol processing can be reduced.The main task is to start the server on specific port in listen mode now server is ready to get the request from the client. Now client can make request to the server by providing server address and the server port detail. Client send connect request. Server will get the connection request same as we get the ring on mobile for connection. After this server can accept or reject the connection and server accepting the request, a connection link gets established between server and client. Now server can send the data to the client and client get the data arrival ACK, after this client can read the data .Same thing happened with server and communication goes on. Finally any one of both can close the connection.

TABLE 1
PARAMETRIC COMPARISON OF LOAD BALANCING ALGORITHMS

Algorithm Parameters	Round Robin	Random	Local Queue	Central Queue	Central Manager	Threshold
Overload Rejection	No	No	Yes	Yes	No	No
Fault Tolerant	No	No	Yes	Yes	Yes	No
Forecasting Accuracy	More	More	Less	Less	More	More
Stability	Large	Large	Small	Small	Large	Large
Centralized/ Decentralized	D	D	D	C	C	D
Dynamic/ static	S	S	Dy	Dy	S	S
Cooperative	No	No	Yes	Yes	Yes	Yes
Process Migration	No	No	Yes	No	No	No
Resource Utilization	Less	Less	More	Less	Less	Less

4 REQUIREMENT ANALYSIS

Operating System : Windows XP
Development Tool : C# (.Net)
Database : SQL Server 2005

5 CONCLUSION

The implemented system is best for batch or mass execution. This parallel processing distributed computing Model can reduce over-heads and it makes the proper utilization of multiple systems rather than implementing supercomputing processor. This system reduces the risk of failure as it can use normal lower configuration PC system to complete the task and even input task is not dependent on the single system. But If not planned properly, a distributed system can decrease the overall reliability of computations if the unavailability of a node can cause disruption of the other nodes.

REFERENCES

[1] Gupta R., Chaube A.R., Singh S. (2011) International Journal of Advanced Research in Computer Science and Software Engineering,.
[2] Wang Y. and Morris R. (1985) *IEEE Trans. Computing*, 34(3), 204-217.
[3] Stone H.S. (1978) *IEEE Trans. Software Eng.*, 4(3).
[4] Stone H.S. (1977) *IEEE Trans of Software Engineering*, 3(1), 95-93.
[5] Sandeep Sharma, Sarabjit Singh, and Meenakshi Sharma, "Performance Analysis of Load Balancing Algorithms", academy of science, engineering and technology, issue 38, February 2008, pp. 269-272.
[6] Miron Livny, Myron Melman (1982) *The Computer Network Performance Symposium*, 47-55.
[7] Hsu C.H. and Liu J.W. (1986) The 6th International Confer-ence on Distributed Computing Systems, 216-223.
[8] Derek L. Eager, Edward D. Lazowska, John Zahorjan (1986) *IEEE Transactions on Software Engineering*, 12(5), 662-675.
[9] Eager D.L., Lazowska E.D. and Zahorjan J. (1986) *Perfor-mance Evaluation*, 6(1), 53-68.
[10] Chow Y.C. and Kohler W. (1979) *IEEE Transactions on Com-puters*, 28, 334-361.