Volume 3, Issue 10, October 2013 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Load Measurement Issues in Dynamic Load Balancing in Distributed Computing Environment

Priyesh Kanungo
Computer Centre
Devi Ahilya Vishwavidyalaya, Indore-452001, India

Abstract— In this paper, the issues in load measurement in dynamic load balancing (DLB) algorithms have been explored. We have analyzed various load indices used to estimate the load on the nodes in the system for load balancing algorithms. The simulation model has been used to address performance issues associated with load balancing. The architecture for implementation and performance evaluation of indices used for capturing and distributing load has been proposed.

Keywords: Dynamic Load Balancing, Process Migration, Load Indices, Threshold Level, Response Time, Process Age.

I. INTRODUCTION

Dynamic load balancing (DLB) technique distributes processing workload as evenly as possible among the nodes in a cluster. This helps in improving response time by minimizing job's execution time, minimizing communication overheads and maximizing resource utilization. [8]. DLB is realized through process migration and allows cluster of nodes to be used as a cost effective alternative to mainframe computing as well as parallel computing [2]. Dynamic load balancing is also used to balance load in a cluster of web servers deployed by websites for processing clients' requests [1].

Performance of load balancing is closely related to the process information made available to the load balancer, accuracy of the load measurement and the efficiency with which such information is used. Load balancing yields greater performance improvement when workload is heavy and unbalanced. The index used to measure the load in the system strongly affects the performance of load balancing algorithm. It may not be possible to include all load indices in calculating the load, as this requires collecting and exchanging huge amount of information. This may lead to excessive communication cost. Moreover, a poor load index may cause some process migrations which do not contribute to balancing the load in the system, thereby making the situation worse [4; 7]. Therefore, basic problem associated with DLB algorithms is to identify the parameters to be used for estimation of the load on various processors and making load management decisions dynamically to define the current state of the system. Effective load index measures will minimize the communication cost in the system and will allow the load balancer to take quick decisions about load distribution, thereby improving the effectiveness of the algorithm. In a heterogeneous environment, load indices must be adjusted to make them comparable [10].

II. LOAD INFORMATION MANAGEMENT

Transfer of processes from overloaded nodes to under-loaded nodes requires load balancer to collect load information from the nodes continuously. Load information management is therefore a critical issue in DLB. The load balancer must have a global knowledge of load on the nodes in the system at any instant. Although entire state of a node may be transferred, typically only selective information is exchanged to minimize the communication cost. [9]. Almost all the load balancing policies require some sort of load indices to measure the workload. Most fundamental issue is the selection of load index to be used in measurement of the load [6]. The choice of load index greatly affects the performance of DLB. We are considering following load indices:

A.Parameters for Static Load Balancing

In static algorithms, load balancing is performed at the compile time before the start of execution of processes. The following parameter may be used in static load balancing:

(a) Processor parameters including number of processors, speed of processors and ratio of speed of i^{th} processor with respect to a base processor.

(b) The program parameters including data size, number of loop iterations, work per iteration, data communication per iteration and execution time of each of the iteration on a base processor.

(c) Network parameters including network latency, network bandwidth and network topology.

On the other hand, DLB algorithms collect load information at run time. This information is based on some load index or combination of load indices. Parameters used for load measurement in DLB are being discussed in the following sections.

B. Processor Queue Length

Number of processes in the ready queue of a node acts as the load measurement index for that node. Using processor queue length, the value of workload can be determined easily. However this index will be effective only if all the processes have nearly same execution time. Using processor queue length, nodes having number of processes more than a threshold value will be overloaded nodes and nodes with number of processes less than the threshold will be the under-loaded nodes.

To decide whether node is overloaded or under-loaded, threshold level of the system is computed. Threshold level may be single or double. In single threshold level, a node accepts new processed if the workload is below this value. The problem with single threshold value is that even with marginal improvement in performance, excessive process transfer will take place causing processor thrashing. Therefore, double threshold level which divides space is d into three regions can be used.

Threshold level T is computed as follows:

$$T = \left(\sum_{i=1}^{n} p_i \right) / n \qquad (1)$$

n is the number of nodes in the system.
p_i *is the* number of processes on node *i*

In a heterogeneous environment, different nodes may have different processing power. The load is adjusted for processing power of the node.

C. Execution Time

If the processes have high variation in their execution times, then processor queue length will not be a suitable load index for measurement of processing load. The nodes with less number of processes may high workload due to high service time requirements of the processes in the queue. Therefore, for calculating exact processing load on a node, we have to find sum of execution times of the processes in the queue. But the problem here is regarding estimation of the execution times of the processes. Although the execution time is not known in advance, it can be estimated using some techniques that are based on the time already used by the process. These techniques are:

(a) If the execution pattern of a job is already known, we can find the suitability of a job for transfer. We can also collect the statistics regarding different program types and their average execution times. Based on this statistics, we can identify whether a program type is short or long.

(b) It has been found that processor time, main memory space and I/O requirement of a process can be predicted prior to its execution, using statistical averaging techniques. In the beginning, estimation is made on the basis of identification of the program being executed. We know that a program executes several computation bursts between its creation and termination. On the basis of these computation bursts, predictions are made for weighted mean calculation of resource requirements and actual usage of resource in its most recent stay in the ready queue of the scheduler. This is called exponential smoothening [5]. The service time of a process can be estimated using exponential smoothening formula. Using execution time of a process for calculating processing load has only theoretical significance, since it is not possible to estimate precise execution time in advance. However, this technique is suitable for benchmarking to compare the other implementable techniques.

D. Process Age

Processes that run for longer time are suitable for transfer rather than shorter processes, as the overhead of transferring a short process may override the benefits of load balancing. However input/output intensive jobs, e.g. interactive jobs which heavily access files on the local nodes, are not suitable for migration as the file access cost will be very high on remote nodes. The jobs that are processor bound are suitable for transfer.

Therefore, it is necessary to identify long processes on heavily loaded nodes that have to be transferred. This may be done by keeping track of age of the processes. It has been found that remaining time needed by a process is linearly related to age of the process. Age refers to the processing time used by the process so far. Workload on a processor is measured by finding the sum of the processing time already used by the processes. Processes longer than the average age of the processes in the system may be long processes. In addition to using these load indices for load balancing, combination of several load indices into a single index to represent a node's load is an interesting research issues that has to be investigated.

More accurate load state information of a node can also be computed by finding average of last few samples of load value on that node. For example:

(a) Last value: This is most simple strategy in which the previous load value from the node is used as its current state information.

(b) Arithmetic mean: Rather than taking the latest load value as current state of the node, arithmetic mean of last three or five most recent load values of that node can be taken as its current load.

(c) Weighted mean: Rather than simply taking arithmetic mean we can take weighted mean of three or five most recent load samples from that node.

In addition to load indices, the performance of load balancing algorithms is also affected by several other factors related to collection of load information. These factors are:

(a) Collection of processing load: Workload can be collected using job traces in trace driven simulation study. Workload can also be generated synthetically. Generally, Poisson's process arrival and exponentially distributed service times are used. Inter-arrival time distribution and service time distribution can also be modeled using Lognormal, Weibull or Pareto distribution if variance in distribution of inter-arrival time and service time is high. In peak times, arrival of processes is less bursty and service time distribution has a low variation.

(b) Load update interval: The collection of load information can be periodic or event based. A typical period is one second or longer while typical events are process creation, termination or migration. The intervals between load index updates should be chosen carefully for the stability of the system. If the interval in collecting load information is too long, the basic objective of load balancing is defeated. This results in poor performance as the load balancer is maintaining outdated information about the system load. On the contrary, collecting load information in short intervals may result in increased networking overheads and over reaction by the load balancing algorithm [3].

(c) Load inaccuracy: Accuracy of load information is necessary for making effective decisions in load balancing. Inaccuracy may be caused due to delay in dissemination of information. There is a time delay between measurement of load information and its use. This inaccuracy can be measured as the statistical mean of difference in queue length at arbitrary time t and $t+\Delta t$. The inaccuracy increases monotonically with increase in the delay of dissemination of the information.

(d) Load level: Load level is measured as the ratio of mean service time to the mean arrival time as:

Load level = mean service time / mean arrival time (2)

Load level is called moderate if it is around 50%. In this case inaccuracy is only moderate even if delay is high. Load level is said to be very busy if it is around 90%. In this case load index inaccuracy in more meaningful as it can cause high error rate in load measurement. Therefore, if nodes are busy, information dissemination delay should be small for transferring the information more accurately [11].

III. ALGORITHM DESCRIPTION

Our objective is to compare various load indices that are used to collect load information in the system. We will compare load balancing using three load indices viz. queue length, process age and execution time. The load balancing technique used here is the shortest algorithm in which process from a heavily loaded node are transferred to a node which is having minimum load.

We have considered Queue length, Process age, Processor execution time as the load indices.

Steps involved in the algorithm are:

1. **Collect the system information:** The information about various nodes in the system and list of all processes on the nodes, their arrival time and execution time are collected.
2. **Collect load information:** Load is calculated on each node in the system.
3. **Calculate the threshold value:** Calculate threshold value of the system to identify heavily loaded nodes.
4. **Identify a heavily loaded node:** Identify a source node and processes to be transferred.
5. **Choose the destination node:** Choose least loaded node in the system using the *shortest* algorithm.
6. **Transfer processes to destination node:** Transfer processes from heavily loaded node to the selected node.

IV. SIMULATION AND RESULT DISCUSSION

We have simulated the algorithm for a cluster of eight nodes and compared the load indices. The simulator was designed and implemented to evaluate load indices for different performance parameters. The simulator used artificial workload to carry out comparisons. Artificial workload provides greater flexibility as compared to real workload and can be easily reproduced. We assumed random process arrival and random service time distribution. Virtual processors are used to process the workload. The load balancer consists of two parts. One module called server module that collects load information and makes job placements. Another module called migration module is responsible for remote execution of processes. Table and Fig. 1 compare the results of simulation studies. We have made the following assumptions in our algorithm:

(a) The system consists of fixed number of non dedicated nodes with heterogeneous architecture.
(b) Fluctuations in bandwidth are negligible and communication delays are constant.
(c) Algorithm used for load balancing is *shortest*.
(d) Process transfer cost is proportional to the size of the process.

TABLE I: COMPUTATION OF MEAN RESPONSE TIME

	Mean Response Time of Processes			
Node Id.	Without DLB	Queue Length	Process Age	Exec. Time
3	11	15	18	21

5	18	19	21	20
4	22	23	21	21
7	27	26	23	21
8	32	29	27	23
6	40	32	29	25
2	48	35	32	28
1	59	40	35	30

Table I and Figure1 compare the three load indices on the basis of mean response time of the processes. The graph shows that DLB using any of the load indices is better than no load balancing at all. Among the three load indices, execution time as a load index gives better results. But we know that if is difficult to estimate execution time of the processes before actually executing them. However, it works as standard to compare other implementable algorithms. Process age as load indices gives better results than queue length.

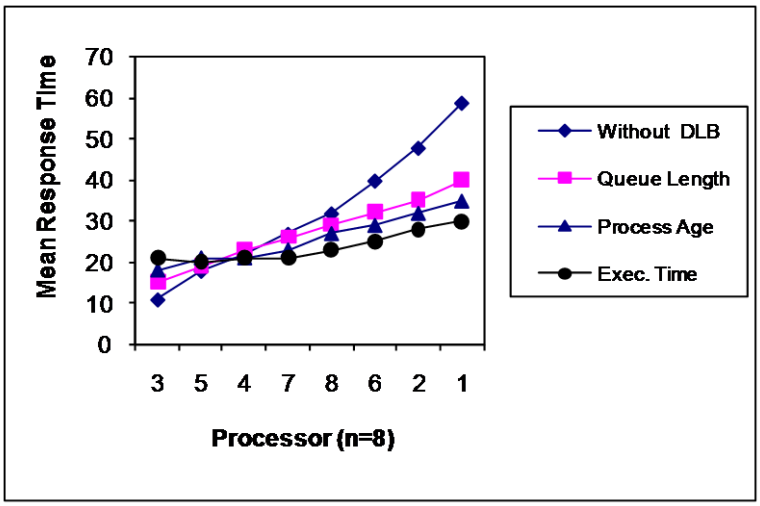

Fig. 1 Comparison of mean response time using different load indices (node numbers are in the ascending order of mean response time)

V. CONCLUSIONS

In this paper, we have investigated various issues in DLB and studied various parameters for effective load measurement in a computing cluster. We have compared various indices used to measure the load on the nodes. Results obtained are from simulation test instead of measurement from real system. The objective of the simulation was to study the effect of different load indices and on the performance of the DLB algorithms.

REFERENCES

[1] T.F. Abdelzaher, K.G. Shin and N. Bhatti, "Performance Guarantee for Web Server End Systems: A Control Theoretical Approach," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 13, No. 1, pp. 80-96, Jan. 2000.

[2] K Amiri. et al., "Dynamic Function Placement for Data Intensive Cluster Computing," *Proceedings of the 2000 USENIX Annual Technical Conference*,San Diego, CA, pp. 1-16, June 2000.

[3] M. Andreolini, M. Colajanni and R. Morselli, "Performance Study of Dispatching Algorithms in Multi-tier Web Architectures," *Performance Evaluation Review*, Vol. 30, No. 22, pp.10-20, Sept. 2002.

[4] T. Desell, , K.E. Maghraoui and A. C. Varel,, "Load Balancing of Autonomous Actors over Dynamic Networks," *Proceedings Hawaii International Conference on System Sciences*, Track 9, Vol. 9, Page 90268.1, 2004.

[5] M. Devarakonda and R.K. Iyer, "Predictability of Process Resource Usage: A Measurement based Study of Unix," *IEEE Transactions on Software Engineering*, Vol. 15, No. 12, pp. 1519-1586, Dec. 1989.

[6] H.D. Kartza, "A comparison of Load Sharing and JobScheduling in NOWs, International Journal of Simulation, Vol 4, No. 3, pp. 84 – 87, 2003.

[7] W. Kleiminger, E. Kalyvianaki and P. Pietzuch, "Balancing of Load in Stream Processing with Cloud," *IEEE International Conference on Data Engineering*, Germany, pp. 16-21, April 2011.

[8] H. Mehta, P. Kanungo and M. Chandwani, "Decentralized Content Aware Load Balancing Algorithm for Distributed Computing Environments," ACM International Conference and Workshop on Emerging Trends and Technology (ICWET 2011), Organized by Thakur College of Engineering and Technology, Mumbai, February 25-26, 2011.

[9] D.S. Milojicic et al., "Process Migration," *ACM Computing Surveys*, pp. 241-299, 2000.

[10] R. Sharma, P. Kanungo and M. Chandwani, "A Dynamic Load Balancing Method using Workstation Priority," *International Journal of Engineering Science and Technology (IJEST)*, Vol. 3, No. 4, pp. 2845-2851, April 2011.

[11] K. Shen, T. Yang and L. Chu, "Cluster Load Balancing for Fine-Grain Network Services," *Technical Report TRCS2002-02*, Department of Computer Science, UC, Santa Barbara, 2002.

[12] A. Tiwari and P. Kanungo, " A Model for Dynamic Load Balancing in Open Source Software for Distribute Computing Environment," CSI 6th international Conference of Software Engineering (CONSEG 2012), Indore, 6th - 7th Sept, 2012 (Proceeding Available on IEEE, Xplore).

