Volume 3, Issue 6, June 2013 ISSN: 2277 128X

International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: www.ijarcsse.com

Schemes for Dynamic Load Balancing – A Review

P. A. Tijare*, Dr. P. R. Deshmukh

**Sipna CoET, Amravati (MS)*
Maharashtra, India

Abstract* — In a distributed network of computing hosts, the performance of the system can depend basically how the work is efficiently divide across the participating nodes. Dynamic load balancing have better performing potential than static load balancing, but they are quite complex. The overheads involved in dynamic load balancing are much more. But one cannot neglect their benefits. The major factors that are strongly considered for the performance are load index, queue-length-based indices. Load balancing is very effective when a large portion of the workload is immobile. All hosts, along with lightly loaded nodes get benefit from load balancing. Similarly, all types of jobs get improvements in their response times, with larger jobs benefiting more. System instability is possible, but can be easily avoided. In a distributed system, the random arrival of number of tasks at each processor is likely to bring about uneven processor loads. There are various approaches to provide the dynamic load balancing.

***Keywords*— Load Balancing, Migration, Priority, Scheduling, Hosts.**

I. Introduction

Today, the use and access of fast processors and also multi-processors has vastly improved and are mostly used in cases that are very time-consuming and time is a valuable source. Thus, in multi-processor systems, scheduling on a number of processors is of great interest for such purposes [1]. Parallel computers perform their calculations by executing different computational tasks on a number of processors concurrently. The processors within a parallel computer generally exchange information during the execution of the parallel code. This exchange of information occurs either in the form of explicit messages sent by one processor to another or different parallel processors sharing a specified common memory resource within the parallel computer. The core idea of load balancing is to use multiple hosts to replace only one host, thereby it can enhance the computing power and reliability of the host at a low cost. In this way, the problem about a large number of concurrent accesses can be solved. Load-balancing system is a cluster system composed of multiple hosts. Each individual host, which can provide services independently without any external assistance of other hosts, has equivalent status in the system.The reliability of load-balancing is relatively higher than that of a single machine. When one or a few hosts in the system fail, rather than being interrupted, the system will adjust scheduling scheme rapidly and assign a new service request to a normal working host, meanwhile, shift the unaccomplished task of the failure host to other hosts. And looking from the outside, it seems the entire system works correctly. Some of the major benefits of parallel computing systems are information sharing among distributed users, resource sharing, better price/performance ratio, shorter response time, higher throughput, higher reliability, extensibility and incremental growth. Load balancing on multi computers is a challenge due to the autonomy of the processors and the inter processor communication overhead incurred in the collection of state information, communication delays, redistribution of load etc. Parallel and distributed computing environment is inherently best choice for solving/running distributed and parallel program applications. In such type of applications, a large process/task is divided and then distributed among multiple hosts for parallel computation [2].

II. Schemes for Dynamic Load Balancing

Load balancing algorithms can be divided into two major categories: static load balancing algorithm and dynamic load balancing algorithm. In static load balancing algorithm, on the basis of the time needed to complete any given task, tasks are assigned to processors during the compile time and their relation is determined. No decision regarding a shifting of a task from one processor to another during execution time. But in dynamic load balancing algorithms (DLB), load status at any given moment is used to decide on task shifts between processors [2, 3, 4, 5]. Random, Central and Rendez-vous are among the existing load balancing algorithms which can be seen in [4, 6, 7]. In parallel and distributed systems processors are divided into three categories according to their workload level. Heavily loaded processor/Overloaded processor: which have a large number of tasks in waiting. Lightly loaded processor/Underloaded processor: which have a small number of tasks in waiting. Idle processors: which have no tasks to execute[2, 7].

The tradeoff between knowledge and overhead is illustrated in [8] with five different DLB schemes. The schemes presented vary in the amount of processing and communication overhead and in the degree of knowledge used in making balancing decisions. The load balancing overhead includes the communication costs of acquiring load information and of informing processors of load migration decisions, and the processing costs of evaluating load information to determine task transfers.

Sender Initiated Diffusion (SID) is a highly distributed local approach which makes use of near-neighbor load information to apportion surplus load from heavily loaded processors to under loaded neighbors in the system. Global balancing is achieved as tasks from heavily loaded neighborhoods diffuse into lightly loaded areas in the system. Receiver Initiated Diffusion (RID) is the converse of the SID strategy, where under loaded processors requisition load from heavily loaded neighbors. Hierarchical Balancing Method (HBM) is an asynchronous, global, approach which organizes the system into a hierarchy of subsystems. Load balancing is initiated at the lowest levels in the hierarchy with small subsets of processors and ascends to the highest level which encompasses the entire system. This scheme centralizes the balancing process at different levels of the tree with increasing degrees of knowledge at higher levels. Gradient Model (GM) employs a gradient map of the proximities of underloaded processors in the system to guide the migration of tasks between overloaded and underloaded processors. Dimension Exchange Method (DEM) is a global, fully synchronous, approach. Load balancing is performed in an iterative fashion by "folding" an N processor system into log N dimensions and balancing one dimension at a time. In current load-balancing systems, the widely used scheduling algorithm is the Polling scheduling algorithm, and there are a lot of improved algorithms. Among these there is a bi-driven algorithm which is a combination of Receiver-Initiated and Sender-Initiated adaptive load balancing algorithms and in this algorithm, Receiver-Initiated and Sender-Initiated both can transfer task [9]. The basic idea is to set a threshold for the entire system, and all the nodes are divided into heavy-load and light-load nodes according to threshold. When the load of a node exceeds the threshold, it would try to transfer task to the light-node. Which node should be selected is depending on the load state in the related node. Then the task is added to the task queue to be implemented.

Dynamic Schedule algorithm inherits the virtue of bi-driven algorithm, ensuring no excessively busy nodes as well as idle nodes in the system. In contrary to polling algorithm, the dynamic schedule algorithm determinate the threshold dynamically based on prediction and feedback, and employ GM approach to reduce efficiency. It is an accomplished fact that the algorithm raises system overhead by introducing mechanism such as host status report, calculate the threshold and select additional overhead of the host. Under heavy-load conditions, it is negligible that concurrent execution between operation on balancer and the task of host. On the other hand, the entire system follows the principle of light-load to report more than heavy-load. To minimize additional pressure of heavy load, it will not give heavy burden on the system [10]. The stability of an approximate linear model in load balancing system is analyzed in [11] and then the asymptotic stable condition via some classical control theories is proved. In this way, the relations between load balancing gain and scalability of system is found. The algorithm in [12] is designed with less time complexity and reducing the time of scheduler to give feasible and better than the common static algorithm.

Task scheduling including three parts is proposed in [13, 14]:

1) Collection of information: Making load threshold value which can measure load information of node as well as making method of information collection.

2) Migration making-decision: judging whether or not migrating one task to other computing node according to task and load of node.

3) Implementation of migration: The task which is suitable to migrate to other computing node should choose aide node to migrate.

A fault-tolerant and reliable load balancing Coordinator and Backup Automatic Election (CBAE) algorithm is proposed in [15] that work by assigning one node as a coordinator and another node as a backup. The CBAE is compared with a well known random election algorithm. Results shows that all nodes are balanced by using CBAE, whereas by using a random approach they are obviously imbalanced.

In [16], two dynamic load balancing schemes DGOS (Dynamic Global Optimal Scheme) and DNCOOPC (Dynamic Non Cooperative Scheme with Communication) for multi-user jobs in heterogeneous distributed systems are proposed. DGOS tries to minimize the expected response time of the entire system, whereas DNCOOPC tries to minimize the expected response time of the individual users. These dynamic schemes use the number of jobs of the users in the queue at each node as state information. The performance of the dynamic schemes with that of the static schemes GOS and NCOOPC are compared. It was observed that, at low communication overheads, both DGOS and DNCOOPC show superior performance over GOS and NCOOPC. Also, the performance of DNCOOPC which tries to minimize the expected response time of the individual users is very close to that of DGOS which tries to minimize the expected response time of the entire system. Furthermore, DNCOOPC provides almost equal expected response times for all the users and so is a fair load balancing scheme. It was also observed that, as the bias, exchange period and the overheads for communication increases, both DGOS and DNCOOPC yield similar performance to that of the static schemes.

A balanced overlay networks (BON) framework is presented in [17], that provides scalable, decentralized load balancing for distributed computing using large-scale pools of heterogeneous computers. BON encodes the information about each node's available computational resources in the structure of the links connecting the nodes in the network. This distributed encoding is self-organized, with each node managing its in-degree and local connectivity via random-walk sampling. Assignment of incoming jobs to nodes with the most free resources is accomplished by sampling the nodes via short random walks. A paper [18] presents a methodology for evaluating runtime systems for NPs. The presented methodology considers the workload of the system in terms of processing characteristics and traffic characteristics. To exercise the evaluated system with a wide range of traffic patterns, a mechanism for generating synthetic traffic traces for different scenarios is presented. The system model uses a queuing network abstraction to represent different runtime systems and allows for an analytical performance evaluation. In [19], the session-level load-balancing problem as a Markov decision problem is formulated. Then, approximate dynamic programming is used to obtain approximate load-balancing policies that are scalable with the problem instance. Extensive numerical experiments show that the policies have nearly optimal performance. A behavioral model is introduced in [20] for parallel applications with large requirements of network, CPU, and disk I/O resources. Each node in the cluster serves multiple processes in a timesharing fashion so that these processes can dynamically share the cluster resources. To facilitate load-balancing technique, the communication load imposed by these processes is measured. The model is particularly beneficial to clusters where resource demands of applications may not be known in advance. In [21], a delay-based redistribution scheme and an extension are proposed to coordinate the reallocation of distributed load for High Level Architecture (HLA) based virtual simulations. Their balancing elements are organized in a hierarchical structure, which is built according to the network topology of the environment used to deploy the distributed simulations. The proposed schemes rely on the measurement of simulation and resources, analysis, detection, redistribution, and migration.

III. Analysis of Dynamic Load Balancing

We observed from [6] that the Rendez-vous algorithm is convergent. On a 'n' processor system, at most log2(n) are needed to reach a steady state where each processor has got the average load of the system as shown in table 1. Communication cost is taken into consideration for accurately evaluating the behavior of redistribution.

TABLE I
Algorithm Convergence & Communication Cost

Algorithm	**Speed of Convergence**	**Communication Cost**	**Communication Policy**
Rendez-vous	~log2(n)	O(nlogn)cg	Global
Tiling	>n/2	O(n)cn	Local
X-Tiling	~log2(n)	O(logn)cu	Uniform
Sliding	<n	O(n)cn	Local

From [10], we compared the dynamic scheduler with polling scheduler and found that in light load conditions the efficiency of dynamic algorithm is similar or slightly lower that polling scheduling algorithm while in heavy load conditions the efficiency of dynamic scheduling algorithm is significantly greater. Figure 1 shows % efficiency of dynamic algorithm as compared to polling algorithm when different numbers of task are considered.

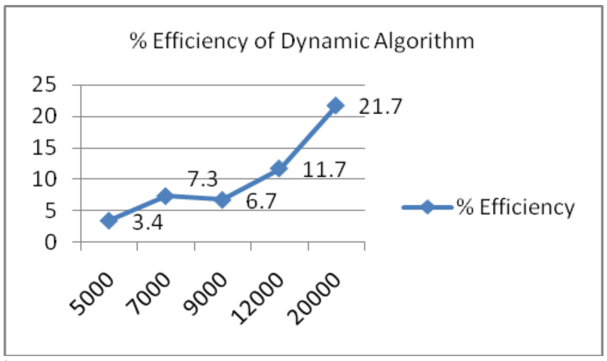

Fig 1: % Efficiency of Dynamic Algorithm than Polling Algorithm Under Heavy Load Condition

In [19] they developed two approximations to dynamic programming operator such that the state space will be reduced. We analyzed that developed approximations have good performance as compared to LB, LC and RR algorithm in terms of hyper exponential and uniform distribution as shown in figure 2 & 3.

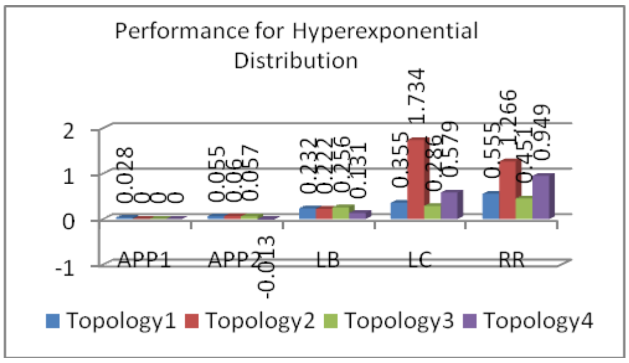

Fig 2: Performance for Hyper exponential Distribution

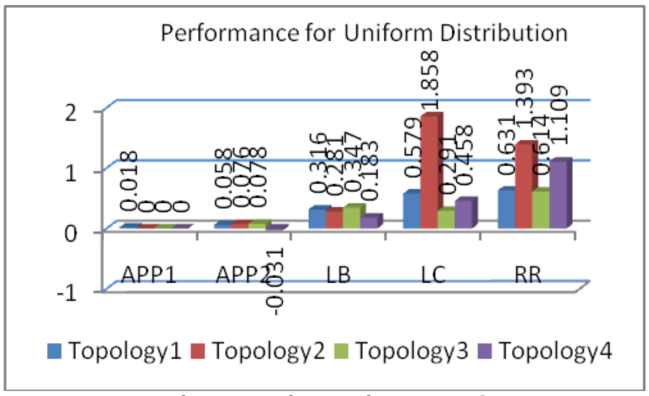

Fig 3: Performance for Uniform Distribution

From [20] we observed that COMM aware load balancer is significantly better than the CPU-aware, MEM-aware and I/O-aware load balancer in terms of mean slowdown performance when each parallel job consists of 16 tasks and message arrival rate is 0.5 No./ms. As shown in figure4

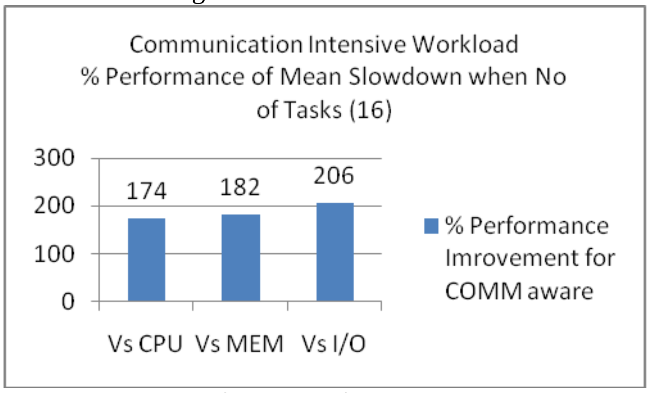

Fig 4: % Performance of Mean Slowdown

Also, for varying network bandwidth (10Mbps, 100Mbps & 1Gbps), the average percentage performance of COMM-aware with respect to mean turnaround time is better than other schemes as shown in figure 5.

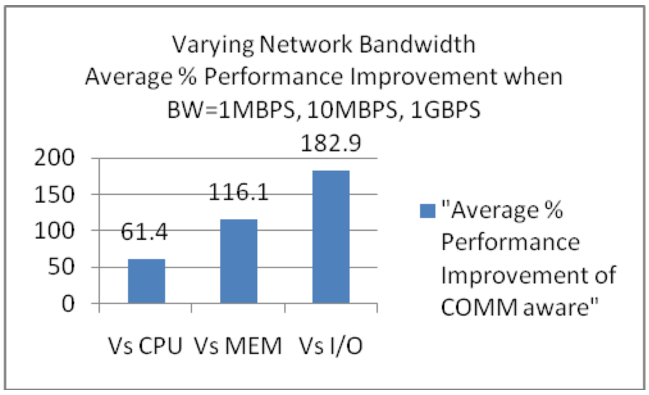

Fig 5: % Performance of Mean Turnaround Time

IV. Conclusion

It is said that in a system of multiple hosts the probability of one of the hosts being idle while other host has multiple jobs queued up can be very high [3]. Here load balancing is likely to improve performance Such imbalances in system load suggest that performance can be improved by either transferring jobs from the currently heavily loaded hosts to the lightly loaded ones or distributing load evenly/fairly among the hosts.

References

[1] R. Buyya and D. Abramson and J. Giddy and H. Stockinger, "Economic Models for Resource Management and Scheduling in Grid Computing", int Journal of Concurrency and Computation: Practice and Experience, Volume 14, Issue.13-15, pp. 1507-1542, Wiley Press, December 2002.

[2] Amit Chhabra, Gurvinder Singh, Sandeep Singh Waraich, Bhavneet Sidhu, and Gaurav Kumar "Qualitative Parametric Comparison of Load Balancing Algorithms in Parallel and Distributed Computing Environment" Proc. World Academy of Science, Engineering and Technology (PWASET) Vol 16, pp. 39-42, November 16, 2006,

ISSN 1307-6884.

[3] Miron Livny, Myron Melman, "Load balancing in homogeneous broadcast distributed systems", Proceedings of the Computer Network Performance Symposium, College Park, Maryland, United States, pp.47-55, April 13-14, 1982.

[4] A. Osman, H. Ammar, "Dynamic load balancing strategies for parallel computers," Scientific Annals Journal of Cuza University, International Symposium on Parallel and Distributed Computing (ISPDC), vol. 11, pp. 110–120, Iasi, Romania July 17-20, 2000.

[5] M. Wu, "On runtime parallel scheduling for processor load balancing," IEEE Transactions on Parallel and Distributed Systems Press Piscataway, vol. 8, pp. 173-186, February 1997.

[6] C. Fonlupt, P. Marquet, J. Dekeyser, "Analysis of synchronous dynamic load balancing algorithms," (PARCO'95) Parallel Computing: State-of-the Art Perspective Advances in Parallel Computing, Elsevier Science Publishers, France, September 1995.

[7] C. Fonlupt, P. Marquet, J. Dekeyser, "Data-parallel load balancing strategies," West Team, High Performance Computing, Laboratoire d'Informatique Fondamentale de Lille, Université de Lille 1, France, December 17, 1996, ISSN 0249-6399.

[8] Marc H. Willebeek-LeMair, "Strategies for Dynamic Load Balancing on Highly Parallel Computers IEEE Transactions on Parallel and Distributed Systems Vol 4. No 9, September 1993

[9] Eager D.L., Lazowska E.D. , and Zahorjan J., "A Comparison of Receiver-Initiated and Sender-Initiated Adaptive Load Sharing", Performance Evaluation, Elsevier Science Publishers, Amsterdam, Holland, Vol. 6. Issue 1, pp. 53-68, March 1986, ISSN: 0166-5316.

[10] Jie Chang, Wen'an Zhou, Junde Song, Zhiqi Lin, "Scheduling Algorithm of Load Balancing Based on Dynamic Policies", Sixth International Conference on Networking and Services (ICNS), IEEE Xplore Digital Library, Cancun,
ISBN: 978-1-4244-5927-8, May 6, 2010.

[11] Qingyang Meng, Jianzhong Qiao, Jun Liu, and Shukuan Lin. "A Dynamic Load Balancing Method Based on Stability Analysis" International Symposium on Computer Science and Computational Technology (ISCSCT), IEEE, Shanghai, China, Vol.1, pp.404-408, Dec 20-22, 2008.

[12] Wenzheng Li, Hongyan Shi, "Dynamic Load Balancing Algorithm Based on FCFS", Fourth International Conference on Innovative Computing, Information and Control (ICICIC), IEEE, Kaohsiung, Taiwan, pp.1528 – 1531, Dec. 7-9, 2009.

[13] Sun Nian, Liang Guangmin, "Dynamic Load Balancing Algorithm for MPI Parallel Computing", International Conference on New Trends in Information and Service Science (NISS), IEEE, Gyeongju, Korea, pp.95–99, June 30 - July 2, 2009.

[14] Yongzhi Zhu, Jing Guo, Yanling Wang, "Study on Dynamic Load Balancing Algorithm Based on MPICH", World Congress on Software Engineering, IEEE, Xiamen, China, May 19-21, 2009.

[15] Tarek Helmy, Fahd S. Al-Otaibi, "Dynamic Load-Balancing Based on a Coordinator and Backup Automatic Election in Distributed Systems", International Journal of Computing & Information Sciences, Vol. 9, No. 1, pp.19 -27, ISSN: 1708-0460 (print) - 1708-0479 (online), April 2011.

[16] Satish Penmatsa and Anthony T. Chronopoulos, "Dynamic Multi-User Load Balancing in Distributed Systems", 21st IEEE Parallel and Distributed Processing Symposium (IPDPS), Long Beach, California USA pp.1-10, March 26-30, 2007.

[17] Jesse S.A. Bridgewater, P. Oscar Boykin, Vwani P. Roychowdhury, "Balanced Overlay Networks (BON): An Overlay Technology for Decentralized Load Balancing", IEEE Transactions on Parallel and Distributed Systems, Vol.18, No.8, August 2007.

[18] Xin Huang,Tilman Wolf, "Evaluating Dynamic Task Mapping in Network Processor Runtime Systems", IEEE Transactions on Parallel and Distributed Systems, Vol.19, No.8, August 2008.

[19] Dennis Roubos and Sandjai Bhulai, "Session-Level Load Balancing for High-Dimensional Systems", IEEE Transactions on Automatic Control, Vol.54, No.8 August 2009.

[20] Xiao Qin, Hong Jiang, Adam Manzanares, Xiaojun Ruan, Shu Yin, "Communication-Aware Load Balancing for Parallel Applications on Clusters", IEEE Transaction on Computers, Vol.59, No.1, January 2010.

[21] Robson Eduardo De Grande, Azzedine Boukerche, and Hussam Mohamed Soliman Ramadan, "Measuring Communication Delay for Dynamic Balancing Strategies of Distributed Virtual Simulations", IEEE Transactions on Instrumentation and Measurement, Vol.60, No.11, November 2011.