Load Balancing Approaches: Recent Computing Trends

Varsha Thakur
Research scholar Pt.Ravishankar shukla University, Raipur (C.G), India

Sanjay Kumar
Associate Professor Pt.Ravishankar Shukla University, Raipur (C.G), India

ABSTRACT
This paper presents thorough survey of work addressing on load balancing in recent computing trends. There are many issues whose solutions lead to the need for load balancing. The objective of load balancing is to increase the performance of parallel and distributed system by distributing the load among the processors. Load balancing is a major factor for achieving high performance. It affects the execution time significantly by expediting it. Load imbalance is a well-known problem in the areas involving parallelism. However, offering load balancing is a difficult and challenging task. Various algorithms have been proposed for load balancing. These algorithms have distinguished features and each uses different mechanisms. Various Load balancing algorithms like biased sampling, honey bee, active clustering, and join idle queue have been studied.

General Terms
Parallel and distributed computing.

Keywords
Load Balancing, Cloud Computing, CloudSim

1. INTRODUCTION
High-performance computing (HPC) was once restricted to institutions that could afford the significantly expensive and dedicated supercomputers of the time. There was a need for HPC in small scale and at a lower cost which lead to cluster computing. The emergence of cluster platforms was driven by a number of academic projects, such as Beowulf [1], Berkeley NOW [2], and HPVM [3]. The popularity of the Internet and the availability of powerful computers and high-speed network technologies have changed the way computers are used. Grid computing originated in academia in the mid-1990s with an aim to facilitate users to remotely utilize idle computing power within other computing centers when the local one is busy [1].Initially, it only referred to a compute grid and had a rather limited audience. However, after years of development the grid gained momentum and came to mean an effective way for coordinated resource sharing and problem solving in dynamic, multi-institutional virtual organizations. Cloud computing, is a kind of computing model that came into existence around the end of 2007. It provides a pool of computing resources which the users can access through Internet. The basic principle of cloud computing is to shift the computing done from the local computer into the network [4]. Load balancing is a term which basically comes from electrical engineering. It is a technique used by electric power stations to store excess electrical power when demand is low and release that power when demand increases.

2. LOAD BALANCING
A Parallel computer will have number of processors and interconnected resources which can work independently or in cooperation with each other. Each processor and resource has their own workload which represents an amount of work to be performed and every one may have different processing capability. To minimize the execution time, workload has to be evenly distributed over all resources and processors based on their processing speed.

2.1 Taxonomy of Load Balancing:
Static load balancing distributes the processes to processors at compile time, while dynamic load balancing binds processes at run time. Static load balancing algorithms may be either deterministic or probabilistic. Deterministic algorithms use the information about the properties of nodes and the characteristics of the processes. Probabilistic load balancing uses information regarding static attributes of the system such as number of nodes, the processing capability of each node, the network topology and so on. Dynamic load balancing is further categorized as centralized and distributed. Centralized load balancing technique store global information at a central location and use this information to balance the load. Centralized load balancing uses the computing and storage resources of one or more processors while in distributed load balancing, the state information is distributed among the processors that are responsible in managing their own resources or allocating tasks residing in their queues to other processors. In non-cooperative approach each processor has autonomy over its own resource while in cooperative load balancing, processes work together towards a common system-wide global balance. Dynamic load balancing is further classified depending on who initiated the load balancing such as sender initiated, receiver initiated and symmetric. In sender initiated, the overloaded nodes transfer one or more of their tasks to several under loaded nodes. In receiver initiated, under loaded nodes request tasks to be sent to them from nodes with higher load. In the symmetric approach, both the under loaded as well as the over loaded nodes may initiate load transfers [4].

3. STRATEGIES AND POLICY OF LOAD BALANCING
Strategies of load balancing can be centralized or de-centralized. In centralized strategies, the load balancer is located on one master workstation node and all decisions are made there while in a de-centralized strategies, the load balancer is divided among all the workstations.

3.1 Policies in Load Balancing
Transfer Policy selects a job for transferring from a local node to a remote nodes, Selection Policy specifies the processors involved in the load exchange. Location Policy selects a destination node for a transferred task is referred to as location policy. Information Policy specifies what workload information to be collected, when it is to be collected and from where .The information exchange policy can obey one of the following policies: Periodic policies in which all process broadcast their state information at certain time intervals. Setting time for intervals are very important because if this time is very small it causes high overhead on the system and if it is very large, it can decrease the accuracy of the system Data-Demand policies collects the state of other nodes only when it becomes overloaded or under loaded. State-change-

driven policies give information about their state whenever their state changes by a certain amount. Triggering policy determines the appropriate period to start a load balancing operation.[2].

4. ALGORITHM

Depending on who initiated the process, If the load balancing algorithm is initialized by the sender then it is Sender Initiated. If the load balancing algorithm is initiated by the receiver then it is Receiver Initiated and if it is the combination of both sender and receiver then Symmetric. Depending on the current state of the system it may be static or dynamic. In static ,it doesn't depend on the current state of the system but Prior information of the system is needed while in dynamic the load balancing are based on current state of the system.

4.1 Static Load balancing

Static load balancing algorithms allocate the tasks of a parallel program to processor based on either the load at the time nodes are allocated to some task, or based on an average load. The decisions related to load balance are made at compile time when resource requirements are estimated. Some static algorithms are Round Robin and Randomized Algorithms. Round Robin and Randomized Algorithms are divided evenly between all processors. Each new process is assigned to new processor in round robin order. The process allocation order is maintained on each processor locally independent of allocations from other processors. Round Robin [5] and Randomized schemes work well with number of processes larger t h a n number of processors. Threshold a l g o r i t h m assigned immediately upon creation to hosts. Hosts for new processes are selected locally without sending remote

Messages. Each processor keeps a copy of the system's load. The load of a processor can characterize by under loaded,medium and overloaded. Recursive bisection recursively divides the problem into sub-problems of equal work while minimizing message passing. Simulated annealing or genetic algorithms is a mixture allocation procedure including optimization techniques.

4.2 Dynamic Load Balancing

Dynamic load balancing algorithms make changes to the distribution of work among processors at run-time, they use recent load information when making decisions. Multicomputer with dynamic load balancing reallocates resources at runtime and determine when and whose tasks can be migrated. However, this comes at the additional cost of collecting and maintaining load information, so it is important to keep these overheads within reasonable limits. Central Queue Algorithm works on the principle of dynamic distribution. Each new job coming at the queue manager is inserted into the queue. Then, whenever a request for an activity is received by the queue manager, it removes the first activity from the queue and sends it to the requester. When a processor load falls under the threshold, the local load manager sends a request for a new activity to the central load manager. The central load manager answers the request immediately if a ready activity is found in the process-request queue, or queues the request until a new activity arrives. Local Queue Algorithm is static allocation of all new job with process migration initiated by a host when its load falls under threshold limit, is a user defined parameter of the algorithm.

5. REVIEW

Load-balancing problems arise in many applications but, most importantly, they play a special role in the operation of parallel and distributed computing systems. Load-balancing deals with partitioning a program into smaller tasks that can be executed concurrently and mapping each of these tasks to a computational resource such a processor. Hwakyung R gives an efficient dynamic load balancing using the dimension exchange method for [5] balancing quantized load on hypercube multiprocessor. In this Dynamic load balancing on hypercube is consider with emphasis on quantized load. Result in difference in assigned loads to processors as large as log n unit after balancing for size n a reduction to ½ log n is done by using slam II Balancing method.

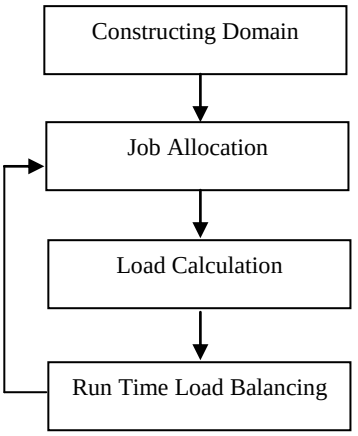

Fig 1. Basic Process for balancing load

Cybenko .G. [6] study diffusion schemes for dynamic load balancing on message passing multiprocessor under which dynamic schemes converge and their rates of convergence for arbitrary topologies. These results use the eigen structure of the iteration matrices that arise in dynamic load balancing. Author completely analyzes the hypercube network by explicitly computing the eigen structure of its node adjacency matrix. Using a realistic model of interprocessor communications, diffusion approach is used to load balancing O. O Olakanmi1 and O.A Fakolujo [7] discuss processing-elements stealing (PEs-S) technique to enforce load balancing in multiprocessor architecture. In case of extreme communication costs or very strong affinity of process for the processors they are assigned to, work stealing will achieve the optimal performance as a result of the fact that stolen works will be transported from its initial processing element to the new processing element. Affinity[8] is measure of cost of moving the task and be generated by compiler on runtime. Using data affinity determine the task to be migrated from heavy loaded to idle processor .

Wentao Wang [9] discuss a design Of load balancing model for multiprocessor in that they analyzed the basic reason for overhead occurring in load balancing is the load migration.They define the four possible states of the node, and discusses the implementation rule of the algorithm in the process of task execution and completion, it is needed to check the task completion situations periodically and

Communicate with other nodes. It should be determined whether the task is migrated or not, and the source Author discuss efficient work stealing for Multicore Event-Driven Systems [10] in this event coloring approach is adapted .

Event coloring is a promising approach it simply and progressively inject support for the safe, parallel execution of multiple event handlers through the use of annotations. It works on stealing algorithm to dynamically balance the

execution of event handlers on the available cores. Xiaozhong

Geng[11] study the Dynamic Load Balancing Scheduling Model Based on Multi-core Processor and represents the dynamic load balancing scheduling problem as a quintuple o f E,T,L,S,C, which gives a formal description about all factors that affect multi-core processors load balancing . Youngho Ahn Won-Jin Kim[12] uses probabilistic information on the expected execution time the child processes for each parent process. The trade-off between load balancing effect of each load balancing unit and the cost is taken into account. They developed a novel method that efficiently distributes workloads of applications into multiple cores with asymmetric memory architecture and runs them in parallel. By collecting precise information about how frequently the process creates child processes and how long the child processes execute by system calls. Jin Sun, Avinash Kodi, Ahmed Louri, and Janet M. Wang proposes [13] a new design framework for multi-core system that includes device wear-out impact. Based on device fractional NBTI model. Enric Musoll proposed hardware-based stateless load-balancing schemes for homogeneous multicore architectures in terms of their power and thermal behavior The different cores in the processor can be switched on and off by usings power gating Round Robin (RRB). It is the best one to distribute the temperature, but fails at minimizing the power .Author [14] study a thermal friendly load balancing technique for multicore processor in this they provides low overheads in performance and energy with respect to highest performance. The scheduling techniques evaluated, however, are again suitable for implementation at the OS level,a temperature sensor for each of the cores is required, which may not be feasible in processors with a large number of cores. Author investigates

the effect of power gating the idle cores, a technique that is very effective in reducing the overall power consumption of the processor .Author[15] have developed a dynamic load balancing library that allows parallel code to be adapted to heterogeneous systems for a wide variety of problems. The strategy was applied to a Dynamic Programming Algorithm, the Resource Allocation Problem. The library has been implemented in a way that does require minimal changing in the code of existing programs.

Table 1. Approaches used for load balancing

S No	Author	Research	Approach
1.	Hwakyung R	quantize d load on hypercube	Dimension Exchange Method
2.	O Olakanmi	Macro Pipeline Multiprocessor	ELements stealing
3.	Fabien G	Workstealin g for Multicore	Event coloring
4.	Alejandro	Heterogeneous Multicore/Mul t iGPU	load balancing library
5.	Albert Y.	Observations	Genetic algorithm
6.	Xiaozhong	Scheduling Model Based on Multi-core	Scheduling

6. ISSUES IN DESIGNING OF LOAD BALANCING ALGORITHM

Load Estimation Policy: Determine how to estimate work load of particular node.

Process Transfer Policy: Determines whether to execute process locally or remotely.

State Information Exchange Policy: Determines how to exchange the system load information among nodes.

Location Policy: Determines which node a process will select for transfer.

Priority Assignment Policy: Determines to the priority of execution of local and remote processes at a particular node.

Migration Limiting Policy: Determines total number of times a process can migrate from one node to other.

7. LOAD BALANCING METRICES

In recent computing, load balancing is required to distribute the dynamic local workload evenly across all the nodes. It helps to achieve a high user satisfaction and resource

Utilization ratio by ensuring an efficient and fair allocation of every computing resource. Proper load balancing aids in minimizing resource consumption, implementing fail-over, enabling scalability, avoiding bottlenecks and over-provisioning etc [16].

In this paper various performance metrics has been considered for existing load balancing techniques such as:
Throughput is used to calculate the no. of tasks whose execution has been completed. Overhead Associated determines the amount of overhead involved while implementing a load-balancing algorithm. It includes overhead due to movement of tasks, inter-processor and inter-process communication. Fault Tolerance is the ability of an algorithm to perform uniform load balancing in case of link failure. The load balancing should be a good fault-tolerant technique. Migration time is the time to migrate the jobs or resources from one node to other. It should be minimized in order to enhance the performance of the system. Response Time is the amount of time taken to respond by a particular load balancing algorithm in a distributed system. Resource Utilization is used to check the utilization of resources. Scalability is the ability of an algorithm to scale according to the requirement.

Performance is used to check the efficiency of the system.

8. LOAD BALANCING AGORITHMS

Honey bee foraging: The main idea behind the algorithm is derived from the behavior of honey bees for finding and reaping food. M. Randles et al. [16] proposed a decentralized honeybee-based load balancing technique that is a nature-inspired algorithm for self-organization. In this case the servers are grouped under virtual servers (VS), each VS having its own virtual service queues. Each Server processing a request from its queue calculates a profit or reward, which is analogous to the quality that the bees show in their waggle dance. If this profit was high, then the server stays at the current virtual server otherwise then the server returns to the forage. The algorithm performs as the system diversity increases. But it has a big disadvantage that it does not increase the throughput as the system size increases.

Biased Random Sampling: M. Randles et al. [2] investigated a distributed and scalable load balancing approach that uses

random sampling of the system domain to achieve self-organization thus balancing the load across all nodes of the system. Here a virtual graph is constructed, with the connectivity of each node (a server is treated as a node). Representing the load on the server. Each server is symbolized as a node in the graph, with each in degree directed to the free resources of the server. The load balancing scheme used here is fully decentralized, thus making it apt for large network

Systems like that in a cloud. The performance is degraded with an increase in population diversity.

Active Clustering: Active Clustering works on the principle of grouping similar nodes together and working on these groups. The performance of the system is enhanced with high resources thereby in-creasing the throughput by using these resources effectively. It is degraded with an increase in system diversity [2].Join-Idle-Queue: This algorithm provides large-scale load balancing with distributed dispatchers by, first load balancing idle processors across dispatchers for the availability of idle processors at each dispatcher and then, assigning jobs to processors to reduce average queue length at each processor. Y. Lua et al.[3] proposed a Join-Idle-Queue load balancing algorithm for dynamically scalable web services. It effectively reduces the system load, incurs no communication overhead at job arrivals and does not increase actual response time. It can perform close to optimal when used for web services. However, it cannot be used for today's dynamic-content web services due to the scalability and reliability. Min-Min Algorithm: It begins with a set of all unassigned tasks. First of all, minimum completion time for all tasks is found. Then among these minimum times the minimum value is selected which is the minimum time among all the tasks on any resources. Then according to that minimum time, the task is scheduled on the corresponding machine. Then the execution time for all other tasks is updated on that machine by adding the execution time of the assigned task to the execution times of other tasks on that machine and assigned task is removed from the list of the tasks that are to be assigned to the machines. Then again the same procedure is followed until all the tasks are assigned on the resources. But this approach has a major drawback that it can lead to starvation [18].Comparisons of various load balancing algorithms is shown below.

Table 2: Comparison of load balancing algorithms

Parameters	Honeybee Scheduling	Biased random Sampling	Active clustering	Join Idle Queue	Min - min
Throughput	No	No	No	No	Yes
Overhead	No	Yes	Yes	Yes	Yes
Fault Tolerance	No	No	No	No	No
Migration Time	No	No	Yes	No	No
Response Time	No	No	No	Yes	Yes
Resource Utilization	Yes	Yes	Yes	No	Yes

9. CONCLUSION

Load balancing is a process of reassigning the total load to the individual core of the collective processors to make resource utilization effective and to improve the response time of the job. Load balancing is required to increase the efficiency of parallel and distributed computing, load balancing is required. The field of parallel and distributed computing is rapidly growing due to the varied advantages and diverse application areas. Load balancing is one of the challenges faced in multiprocessors and multicores. In this paper, a number of Load Balancing techniques and algorithms have been surveyed, each having different aspects of balancing. In many cases, it was difficult to compare them with each other directly as each one of them had different assumptions and employed various mechanisms to achieve the goal. According to the study, these techniques have different strengths and drawbacks some are general and robust, but the randomness result in convergence, the fault tolerance and rejection rate are not considered. Centralized load balancing algorithms suffer from scalability issue. A load balancing technique can hardly satisfy all requirements, but each technique is designed to provide the maximum possible requirements, according to the scenarios. In the years to come, as parallel and distributed computing grow, multiprocessors and multicomputer will continue to flourish, and a load balancing would be required with more modifications as it will be very complicated and will have a tremendous amount of information to be maintained. Satisfying most of the requirements would result in maximizing throughput, minimizing execution time and increase in performance. Different algorithms have been discussed in the review; each of them varies based on certain specific criteria. More efforts are needed to formulate an efficient Load Balancing Algorithm for processors based on different parameters like queue length, migration cost , CPU utilization information based on the event that change the load and job migration with optimum resource utilization with increased performance. Comparisons are done on various algorithms of load balancing.

10. REFERENCES

[1] SadaShiv N. and Dilip K.S.M., "Cluster, Grid and Cloud Computing: A Detailed Comparison", International Conference on Computer Science & Education, Singapore, Aug, 2011, pp 477-482.

[2] http://www.nist.gov/itl/csd/cloud-102511.cfm.

[3] Rajath Y.S.,"A Novel way of Improving CPU Utilization In Cloud", thesis.

[4] Mohiuddin A., Abu S. M., Mustaq A., Md. Mahmudul H. R., "An Advanced Survey on Cloud Computing and State-of-the-art Research Issues". IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 1, No 1, Jan 2012, pp601-608.

[5] Wilkinson B., Allen .M., 2004 Parallel Programming Techniques & Applications Using Networked Workstations & Parallel Computers 2nd ed., Pearson Education Inc.

[6] Cybenko .G.,1989 Dynamic Load Balancing for Distributed Memory Multiprocessors Journal of Parallel and Distributed computing 7, pp 279-301.

[7] Olakanmi1 and O.A Fakolujo 2011," Load Balancing in the Macro Pipeline Multiprocessor System using Processing Elements Stealing Technique", Ubiquitous

Computing and communication journal, pp 28.

[8] Stephens "the importance of locality in scheduling and load balancing for multiprocessor".

[9] Wentao Wang, Xiaozhong Geng Qing Wang, 2011, "Design of a Dynamic Load Balancing Model for Multiprocessor Systems", IEEE computer pp 641-645.

[10] Fabien G, Sylvain G, Renaud L. B. Fabien M, Gilles V Quéma,2010 "Efficient Workstealing for Multicor Event-Driven Systems "International Conference on Distributed Computing Systems IEEE ;pp 516- 525.

[11] Xiaozhong Geng, Gaochao Xu, Yuan Zhang, "Dynamic Load Balancing Scheduling model Based on Multi-core Processor", Fifth International Conference on Frontier of Computer Science and Technology , pp 398 -403.

[12] Youngho. A & Won-J. K 2010 ."A novel load balancing method for multicore with NUMA", ISOCC 2010. pp 412-415 .

[13] Jin Sun_, Avinash Kodi, Ahmed Louri_, and Janet M. Wang, "NBTI Aware Workload Balancing in Multi-core Systems".

[14] Musoll .E, 2008." A thermal-friendly load-balancing technique for multi-core processors" in International Symposium on Quality Electronic Design , pp549 -552.

[15] Alejandro .A, Robert C, Vicente B, and Francisco, 2010," A Dynamic Load Balancing on Heterogeneous Multicore/MultiGPU". Systems ieee, pp 467- 477.

[16] Sharma M. and Sharma P.," Performance Evaluation of Adaptive Virtual Machine Load Balancing Algorithm", (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 3, No.2, 2012 pp 1-3

[17] Jasmin James and Dr. Bhupendra Verma, "Efficient VM load balancing algorithm for a cloud computing environment", International Journal on Computer Science and Engineering (IJCSE), 09 Sep 2012.

[18] R. Buyya, R. Ranjan, and R. N. Calheiros, "Modeling And Simulation Of Scalable Cloud Computing Environments And The Cloudsim Toolkit: Challenges And Opportunities," Proc. Of The 7th High Performance Computing and Simulation Conference (HPCS 09), IEEE Computer Society, June 2009.

[19] CloudSim: A Framework for Modeling and Simulation of Cloud Computing Infrastructures and Services, The Cloud Computing and Distributed Systems (CLOUDS) Laboratory, University ofMelbourne, (2011) available from: http://www.cloudbus.org/cloudsim

[20] Rodrigo N. Calheiros, Rajiv Ranjan, Anton Beloglazov, César A. F. De Rose, and Rajkumar Buyya CloudSim: "A Toolkit for Modeling and Simulation of Cloud Computing Environments and Evaluation of Resource Provisioning" cloudsim.pdf 2011.

[21] Bhathiya, Wickremasinghe."Cloud Analyst: A Cloud Sim-based Visual Modeller for Analysing Cloud Computing Environments and Applications", 2010, IEEE.