A Comparative Study of Load Balancing Algorithms: Static Approaches vs. Real-time Adaptation

Amrita Soni *[1], Dr. Mayank Pathak[2]

[1]M. Tech Scholar, Department of Computer Science and Engineering, Technocrats Institute of Technology, Anandnagar, Bhopal, Madhya Pradesh 462021

[2]Professor, Department of Computer Science and Engineering, Technocrats Institute of Technology, Anandnagar, Bhopal, Madhya Pradesh 462021

Corresponding Author-

amritasoni83@gmail.com

Abstract: Pay-per-use cloud computing offers scalable IT resources over the internet, making it a major paradigm in distributed computing. Because of its adaptability, it has been quickly adopted by various industries, which has accelerated the growth of data centers. In cloud computing, load balancing is essential for improving system performance, cutting down on execution times, and making the best use of available resources. This study investigates the effects of several load balancing strategies on distributed systems, with a focus on dynamic and static algorithms. Static load balancing depends on a predetermined distribution and makes no adjustments during runtime, whereas dynamic load balancing, which can be distributed or non-distributed, adjusts in real-time to system changes. The study also compares various methods, emphasizing their advantages and disadvantages. To highlight developments in load balancing techniques, notably those utilizing metaheuristic algorithms, job scheduling in Infrastructure as a Service (IaaS), and server consolidation, important findings from recent studies are also discussed. The analysis emphasizes how crucial effective load balancing is to maintaining system resilience, making the best use of available resources, and enhancing service quality.

Keywords: Cloud computing, load balancing, dynamic load balancing, static load balancing, distributed systems

1. Introduction

A new paradigm for large-scale distributed computing is emerging: cloud computing. Large data centres now house computing and data instead of desktop and portable PCs. It offers pay-per-use scalable IT resources—such as services and apps—as well as the infrastructure they run on via the Internet, allowing for rapid and simple capacity adjustments. It assists any organization in avoiding the capital costs of software and hardware and helps to adapt to changes in demand. Therefore, cloud computing is a framework that makes it possible for users to access a shared pool of computing resources (such as networks, servers, storage, apps, and services) on an appropriate, on-demand basis. Quick provisioning and deprovisioning of these resources requires little management work or communication with service providers. This aids in promoting availability even more. The industry has embraced cloud computing widely as a result of its exponential growth, and data centres are expanding quickly [1].

The technology makes use of the Pay-Per-Use model, and numerous of its services are frequently found in well-known IT firms like Google, Microsoft, IBM, and so forth. With this approach, customers can buy the services they need in accordance with their demands; it works similarly to a metered service, or as subscriptions. The Software as a Service (SaaS) distribution model makes extensive use of this kind of approach. Figure 1 below gives an overview of cloud computing. To manage the cloud environment, all cloud entities collaborate with one another. For instance, cloud auditors serve as the cloud's police, making sure that CSPs provide high-quality, reliable services. In order to deliver services to clients, cloud carriers ensure a reliable connection (cloud users).

In a private cloud, the data centre is housed within the company's network; in a public cloud, it is located online and is dependent upon Cloud Service Providers (CSPs); in a hybrid cloud, it may be located in both.

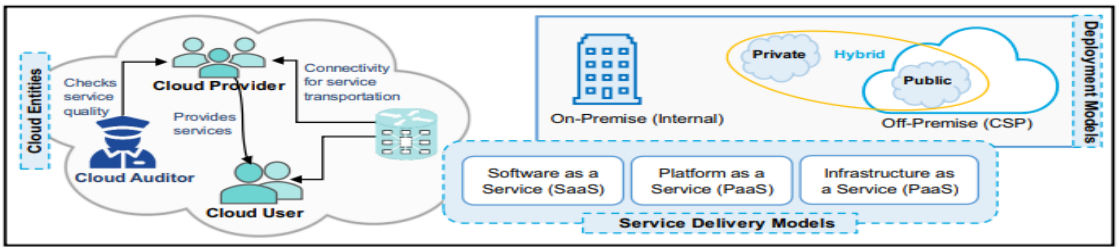

Figure 1: Overview of Cloud Computing [2]

For dynamic resource provisioning, the cloud's resources are both diverse and adaptable. For optimal performance, distributed or hierarchical environments employ DLB approaches. Through load balancing in Platform as a Service, user processes on virtual machines (VMs) operating in the cloud are distributed efficiently. The load balancer selects a suitable virtual machine (VM) from the list of running VMs and routes user requests to that VM for processing. The system's present condition and variations in resource availability are taken into account by the dynamic scheduled load balancers. For cloud service providers and users alike, the load balancing with optimal cost scheduling algorithm technique is utilized with the least amount of execution time and expense [3]. Through effective work scheduling and adjusted resource allocation procedures, the upgraded and efficient scheduling algorithms offer better load balance and improved strategies. System load balancing is accomplished by shifting only extra work from overloaded virtual machines (VMs) using the task-based system load balancing approach employing particle swarm optimization (TBLSBPSO). The optimization methodology decreases the amount of time needed for the load balancing procedure, gets rid of virtual machine downtime, and improves cloud users' quality of service [4]. Higher level dataset fragmentation, in which the datasets are split into a greater number of dataset elements of equal or varying size, is used in load balancing. Additionally, it gives each computer unit a task based on a tiny dataset. This technique's primary benefit is optimal resource utilization. In addition to preventing overload, the load balancing technique maximizes throughput, minimizes reaction times, and optimizes resource utilization. For load balancing, a number of fault tolerance strategies are employed, including replication, job migration, self-healing, and static load balancing. The load balancing method takes into account various elements, including node selection, node interaction, performance, stability, and load estimation and comparison. The payload flow is governed by an algorithm that uses safety levels based on available bandwidth and machines. The dynamic, on-demand balance of resources accessible to the resources for the execution of the project is provided by the load balancing algorithm [5].

2. Types of Load Balancing

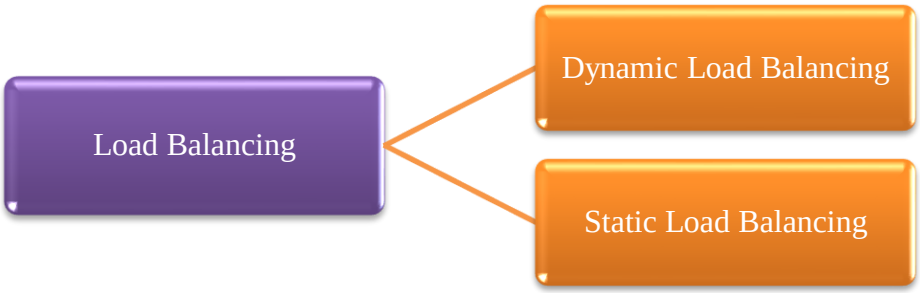

Figure 2: Types of Load Balancing

A. Dynamic Load balancing

There are two methods for implementing dynamic load balancing in a distributed system: distributed and non-distributed is shown in figure 2. In the distributed one, all of the system's nodes share the load balancing responsibility and each one executes the dynamic load balancing algorithm. There are two ways in which nodes can interact with one another to accomplish load balancing: cooperative and non-cooperative. In the first, the nodes collaborate to accomplish a shared goal, such speeding up response times generally. In the second type, every node works independently to achieve a local objective, such speeding up a local task's response time.

Distributed dynamic load balancing techniques typically produce more messages than their non-distributed counterparts since every node in the system must communicate with every other node. One advantage of this is that, should one or more nodes in the system fail, the load balancing process won't stop entirely; instead, it will only have a little impact on the system's performance. When every node in the system needs to communicate its status to every other node in the system, distributed dynamic load balancing can put a great deal of strain on the system. When the majority of the nodes operate independently and interact with one another sparingly, it is more beneficial. In the non-distributed kind, load balancing is handled by one node or by a collection of nodes. There are two types of non-distributed dynamic load balancing algorithms: semi-distributed and centralized. In the first type, the central node—the only node in the entire system—is the only one that uses the load balancing algorithm. The entire system's load balancing is the exclusive responsibility of this node. Only the center node communicates with the other nodes. The system's nodes are divided into clusters in a semi-distributed configuration, with centralized load balancing for each cluster. The task of load balancing within each cluster is handled by the central node, which is chosen using the proper election procedure. Therefore, the central nodes of each cluster handle the system's overall load balancing. In comparison to the semi-decentralized scenario, centralized dynamic load balancing significantly reduces the total number of interactions within the system, requiring fewer messages to reach a conclusion. On the other hand, centralized techniques may result in a bottleneck at the central node, and if the central node crashes, the load balancing procedure is rendered ineffective. As a result, tiny networks are best suited for this approach [6].

B. Static Load balancing

The traffic is distributed equally among the servers in a static mechanism. To ensure that the choice to move the load is independent of the system's current state, this algorithm necessitates prior knowledge of the resources available to the system. Static algorithms work well in systems with little load fluctuation [7]. In cloud computing strategies, the static loads usually fall under two categories. The first is the arrival of the first task, and the second is the initial availability of physical equipment. Each work will be scheduled, and then the resources will be updated. The following heuristics are some of the ones that are provided in static strategy: Min-Min, Min-Max, TABU, GA, MCT, OLB, MET, and Switching Algorithm [8]. Static algorithms are suitable for low-variation load systems. The traffic is distributed equally among the servers in a static mechanism. Prior knowledge of the system resources is required for this algorithm. Because the processors' performance is predetermined at the start of execution, the choice to move the load is independent of the system's current condition. Static load balancing techniques do have a disadvantage, though, in that tasks cannot be transferred during execution to another machine for load balancing; instead, they are assigned to the processor or machines immediately after they are produced [9].

A comparison of dynamic and static load balancing strategies in distributed systems is shown in the accompanying table 1. The text delineates the principal distinctions between the execution techniques, node participation, communication overhead, system resilience, bottlenecks, performance modifications, appropriateness for system dimensions, and illustrations of algorithms employed in both dynamic and static load balancing methodologies. By highlighting the advantages and disadvantages of each strategy, the comparison aids in determining the best load balancing method given the specifications and features of the system.

Table 1: Comparative Analysis of Dynamic and Static Load Balancing

Criteria	Dynamic Load Balancing	Static Load Balancing

Nature	Can be distributed or non-distributed	Typically non-distributed
Execution	Algorithms executed during runtime based on current system status	Algorithms executed based on prior knowledge and do not depend on the current system state
Node Involvement	Distributed: All nodes participate; Non-distributed: Single or group of nodes participate	Load is evenly divided among servers without runtime intervention
Interaction among Nodes	Distributed: Cooperative or non-cooperative; Non-distributed: Centralized or semi-distributed	No interaction during execution
Communication Overhead	High in distributed; Lower in non-distributed	Low
System Resilience	High in distributed; failure of some nodes does not halt the process	Low; failure of the central node in centralized algorithms halts the process
Bottleneck	Distributed: No single point of failure; Non-distributed: Potential bottleneck at central node	Possible bottleneck due to reliance on initial task distribution
Performance Adjustment	Adjusts load dynamically during runtime	Does not adjust; relies on pre-determined load distribution
System Size Suitability	Suitable for large and dynamic systems	Suitable for smaller systems with low load variation
Algorithm Examples	Centralized, Semi-distributed, Cooperative, Non-cooperative	OLB, MET, MCT, GA, Min-Min, Min-Max, TABU, A*
Task Migration	Possible during execution	Not possible; tasks remain assigned to initial processor

3. Analysis of Key Findings from Past Studies

A comparison of several research projects on load balancing in cloud computing is shown in this table. Important details like the main topic of each study, the load balancing strategies that were addressed, the evaluation procedures, the desired Quality of Service (QoS) criteria, and any suggested fixes or innovative algorithms are all included in the table 2. The comparison attempts to draw attention to the variations and developments in load balancing techniques, specifically with regard to effectiveness, resource usage, and fault tolerance.

Table 2: Comparative Analysis of Key Findings from Past Studies

Reference	Primary Focus	Load Balancing Techniques	Evaluation Methods	Targeted QoS Parameters	Proposed Solutions or

					Novel Algorithms
Shahid et al. (2020) [10]	Challenges in load balancing and need for fault tolerance (FT) metrics	Various load balancing algorithms	Critical study and analysis of existing techniques	Throughput, performance, resource usage, scalability, fault tolerance	Proposes a novel load balancing algorithm that incorporates FT metrics
Syed et al. (2024) [11]	Systematic review of static and dynamic load balancing, with focus on metaheuristic algorithms	Static and dynamic load balancing, metaheuristic-based algorithms	Comparative analysis using CloudSim simulation tool	Response time, time complexity, adaptability, targeted QoS parameters	Identifies metaheuristic-based dynamic algorithms as optimal for handling increased traffic
Shafiq et al. (2021) [12]	Task scheduling and load balancing in IaaS cloud model	Dynamic load balancing in IaaS	Resource utilization, execution time, makespan	Resource utilization, execution time, SLA adherence, priority of VMs	Proposes a dynamic load balancing algorithm achieving 78% resource utilization
Ala'Anzy & Othman (2019) [13]	Server consolidation for load balancing in data centers	Load balancing via server consolidation	Meta-analysis of existing algorithms	Energy consumption, resource utilization, network traffic, reliability	Presents a taxonomy and classification for load balancing and server consolidation algorithms
Ghomi et al. (2017) [14]	Comprehensive review of task scheduling and load balancing categories	Various categories including Hadoop MapReduce, Agent-based, etc.	Literature review	Response time, cost, throughput, performance, resource utilization	Provides a new classification of load balancing algorithms and discusses open issues

Maurya & Sinha (2022) [15]	Analytical review of load balancing techniques	Memory, computation, and network load balancing	Proposal for load balancing with decision theory	System stability, load optimization, decision-making	Introduces a decision theory-based load balancing proposal
Elmagzoub et al. (2021) [16]	Survey of swarm intelligence-based load balancing techniques	Genetic Algorithm, BAT, Ant Colony, Grey Wolf, etc.	Performance analysis based on response time, processing time, etc.	Response time, processing time, quality parameters	Surveys and analyzes swarm intelligence algorithms for optimal load distribution

4. Conclusion

Examining load balancing strategies in cloud computing shows how important a role these strategies play in improving system performance, resource efficiency, and overall effectiveness. Large and dynamic systems benefit from dynamic load balancing because of their capacity to adjust in real-time. Distributed techniques that guarantee system resilience even in the event of node failures further highlight this benefit. However, there can be difficulties due to the intricacy and additional communication overhead. However, static load balancing lacks the flexibility to adjust during runtime, which could result in less-than-ideal performance in dynamic contexts. It is simpler and better suited for smaller systems with little load variance. The comparative study of various methods highlights how crucial it is to choose the right load balancing approach in accordance with the specifications and features of the system. In addition, new research indicates that sophisticated algorithms—like metaheuristic-based dynamic load balancing—are becoming more and more popular as viable ways to manage intricate cloud settings. All things considered, effective load balancing is necessary to preserve fault tolerance, guarantee high-quality service delivery, and maximize the utilization of cloud resources in an increasingly demanding digital environment.

References

[1] Sidhu AK, Kinger S. Analysis of load balancing techniques in cloud computing. International Journal of computers & technology. 2013 Mar;4(2):737-41.

[2] Lowe D, Galhotra B. An overview of pricing models for using cloud services with analysis on pay-per-use model. International Journal of Engineering & Technology. 2018 Jul;7(3.12):248-54.

[3] Ashalatha R, Agarkhed J. Dynamic load balancing methods for resource optimization in cloud computing environment. In2015 Annual IEEE India Conference (INDICON) 2015 Dec 17 (pp. 1-6). IEEE.

[4] Ramezani F, Lu J, Hussain FK. Task-based system load balancing in cloud computing using particle swarm optimization. International journal of parallel programming. 2014 Oct;42:739-54.

[5] Sran N, Kaur N. Zero proof authentication and efficient load balancing algorithm for dynamic cloud environment. Int J Adv Res Comput Sci Softw Eng. 2013 Jul;3(7):128-2277.

[6] Kumar YR, Priya MM, Chatrapati KS. Effective distributed dynamic load balancing for the clouds. International Journal of Engineering Research & Technology (IJERT). 2013 Feb;2(2):1-6.

[7] Desai T, Prajapati J. A survey of various load balancing techniques and challenges in cloud computing. International Journal of Scientific & Technology Research. 2013 Nov 25;2(11):158-61.

[8] Mishra SK, Sahoo B, Parida PP. Load balancing in cloud computing: a big picture. Journal of King Saud University-Computer and Information Sciences. 2020 Feb 1;32(2):149-58.

[9] Shah N, Farik M. Static load balancing algorithms in cloud computing: challenges & solutions. International Journal of Scientific & Technology Research. 2015 Oct;4(10):365-7.
[10] Shahid MA, Islam N, Alam MM, Su'ud MM, Musa S. A comprehensive study of load balancing approaches in the cloud computing environment and a novel fault tolerance approach. IEEE Access. 2020 Jul 14;8:130500-26.
[11] Syed D, Muhammad G, Rizvi S. Systematic Review: Load Balancing in Cloud Computing by Using Metaheuristic Based Dynamic Algorithms. Intelligent Automation & Soft Computing. 2024 Mar 1;39(3).
[12] Shafiq DA, Jhanjhi NZ, Abdullah A, Alzain MA. A load balancing algorithm for the data centres to optimize cloud computing applications. IEEE Access. 2021 Mar 10;9:41731-44.
[13] Ala'Anzy M, Othman M. Load balancing and server consolidation in cloud computing environments: a meta-study. IEEE Access. 2019 Sep 30;7:141868-87.
[14] Ghomi EJ, Rahmani AM, Qader NN. Load-balancing algorithms in cloud computing: A survey. Journal of Network and Computer Applications. 2017 Jun 15;88:50-71.
[15] Maurya SK, Sinha G. Load balancing in cloud computing: an analytical review and proposal. Indonesian Journal of Electrical Engineering and Computer Science. 2022 Jun;26(3):1530-7.
[16] Elmagzoub MA, Syed D, Shaikh A, Islam N, Alghamdi A, Rizwan S. A survey of swarm intelligence based load balancing techniques in cloud computing environment. Electronics. 2021 Nov 8;10(21):2718.