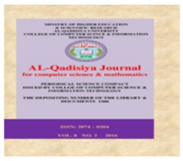
Available online at www.qu.edu.iq/journalcm
JOURNAL OF AL-QADISIYAH FOR COMPUTER SCIENCE AND MATHEMATICS
ISSN:2521-3504(online) ISSN:2074-0204(print)



Adaptivity In Distributed Load Balance Approach in Cloud Computing

Fadheela S. Abu Almash[a], Azhar H. Nsaif [b] , Maytham S. Jabor [c]

a. Ministry of higher education and scientific research, Baghdad, Iraq, Fadheelasabri63@gmail.com
b. Computer Science Department / College of Science / Mustansiriyah University, Baghdad, Iraq, bahar299947@gmail.com
c. Instituto ITACA. Universitat Politècnica de València, maytham.jebory@gmail.com

ARTICLE INFO

Article history:
Received: 03/04/2024
Rrevised form: 06/05/2024
Accepted : 01/06/2024
Available online: 30 /06/2024

Keywords:

ACSIM framework, Cloud Computing, MAPE-K Algorithm, Load Balancing, ACSIM framework.

ABSTRACT

Abstract: Cloud computing has supplanted conventional computing environments. The demand for better-optimized workload allocation and resource efficiency is increasing as ACSIM, a distributed framework-based service, continues to grow. This paper suggests the ACSIM framework technique to manage an adaptive algorithm based on the Apply MAPE-K algorithm with the execution of the MAPE-K loop. The evaluation phase is applied as real work instead of storing previous data; most importantly, the actual results of our framework can be evaluated. This method produces the best infrastructure, application, and platform results, respectively. We investigated the suggested approach using the cloud, and the results demonstrate gains in throughput maximization and reaction time reduction. The optimization of resource utilization and job responsiveness can pose a challenge in ACSIM, given the task of managing resources and scheduling jobs. The Throttled Load Balancing Algorithm is a viable method for effectively handling and processing multimedia data in cloud-based settings, thereby enhancing the performance and responsiveness of mobile applications. Therefore, handling distributed time allocation for each device within the Mobile Cloud is crucial. The response time of Node is being reduced due to the distribution of load across multiple servers, which is the objective of the Load Balancing Algorithm. The findings demonstrate the analytical efficacy of time division by utilizing various virtual machines. the management time was filtered so that it became from Start (MS) (821.2013), Processing Time (MS) (821.201), and 0.000803 Response Time (MS). In the case of balancing, we observe the start (MS) (507.9036), the processing time (MS) (507.92045), and the response time (MS) (0.00113). Consequently, the utilization of the Load Balance Algorithm confers a tangible benefit. In the context of Mobile Cloud environments, load-balancing algorithms MSC.

https://doi.org/10.29304/jqcsm.2024.16.21537

1.Introduction

With the evolution of the computing paradigm, cloud computing has become an all-encompassing, internet-based computing solution for global corporations. Using this method, customers can provision, release, and use virtualized resources as a metered service on an as-needed basis for a predetermined cost. These advantages have helped to make cloud computing widespread. All of its services are delivered via various as-a-service models, such as “Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS), and Software-as-a-Service (SaaS)” [1]. The cloud may be seen as the height of TCP/IP-based internet development under the moniker ARPANET (Advanced Research Projects Agency Network) [2]. Using several

∗Corresponding author

Email addresses:

Communicated by ‘sub etitor’

scalable and virtualized resources, cloud computing technology provides a variety of services to all users. The primary goal of the cloud is to deliver services at the lowest cost and highest quality, wherever they are on the globe. Every client request must be handled effectively, with minimum time and resources waste, to have the capacity to allow many customers across the world share cloud resources and provide them with high-quality service promptly.

Therefore, load-balancing methods, the lynchpin of every cloud service provider's success, are in high demand [3]. The main goal of any cloud-based load-balancing algorithm is to promptly offer the client the required services, with all transactions between the client and the cloud service provider proceeding without a hitch Users and cloud service providers will sign Service Level Agreements (SLAs) [4]. One common goal of process scheduling systems is load balancing. Allocating workflow jobs via load balancing ensures that all available clouds are used effectively in a geographically dispersed cloud architecture. Load balancing is a technique that can improve the Quality of Service (QoS) in response time, cost, throughput, and performance. As a result of the variety of servers in geographically distributed clouds, response times and energy costs can be reduced by employing geographical load balancing to distribute workloads [5]. It ensures that the burden is divided equally across the available system nodes or processors so that the current operation may be finished without interruption. Adaptively organized distribution by dynamically responding to imbalances in processing resources, cloud computing, synchronization, and communication. Most of today's cutting-edge load-balancing technologies are hacked together piecemeal, making them difficult to recycle and update. The ACSIM framework also suggests a new design approach that may be employed in creating the most popular multi-chip/multi-board parallel systems with hardware accelerators. ACSIM, a distributed framework to apply a load-balanced approach to the MAPE-K algorithm, is the contribution of this study. Two distinct implementations are used to assess this technique: a massively parallel micro-traffic simulation in a graph-based setting.

1.1. Cloud Computing

Cloud computing, an advanced technology, provides services to both commercial and public sectors, including immediate internet access to data, applications, and files [12]. Furthermore, it offers customers a wide range of cost-effective, adaptable, and interactive services. As technology continues to decrease hardware costs, it enhances companies globally. Several prominent technological companies, such as Microsoft and IBM, routinely utilize many of the capabilities provided by this technology. The technology functions based on a Pay-Per-Use business model [13]. Similar to metered services, which are also referred to as subscriptions, this method allows users to purchase the services they require in accordance with their needs. The Software as a Service (SaaS) distribution paradigm frequently makes advantage of this concept [14]. An overview of cloud computing is shown in Fig. 1 below. The management of the cloud environment is the shared goal of all cloud entities. As the cloud's version of the police, for example, cloud auditors ensure the integrity and high standard of services rendered by CSPs. Additionally, cloud carriers offer a dependable connection so that customers (cloud users) can benefit.
As opposed to the public cloud, which is exclusively accessible online and is managed by cloud service providers (CSPs), the hybrid cloud permits the location of its data center in a location intermediate to both. There are two main parts to any Cloud Computing setup: the user interface and the underlying infrastructure. The interface is user-facing and is accessed via network connections [15]. On the other hand, the back end is concerned with cloud service models. It consists of a Data Center, which is shown in Fig. 2 below, where multiple physical devices are kept (servers).

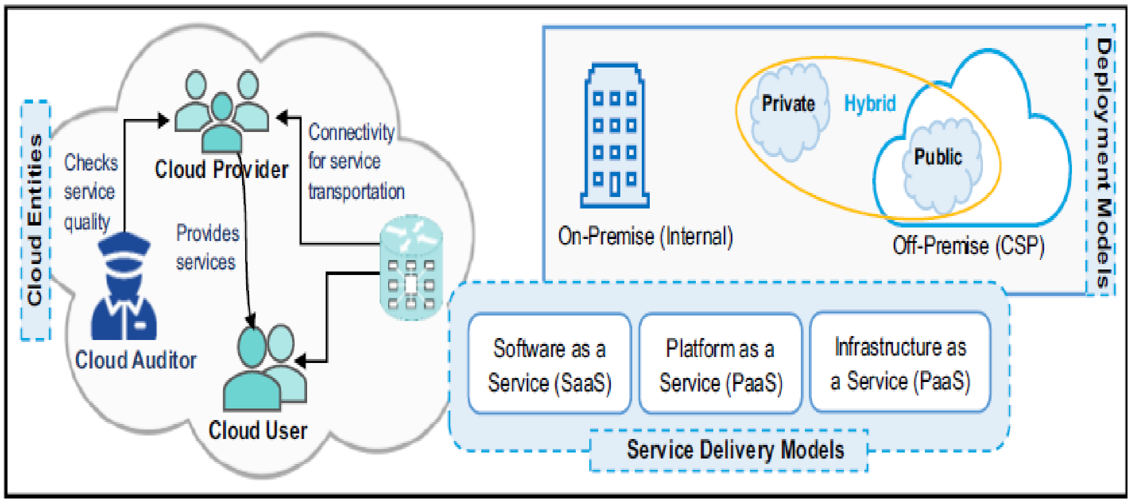


Fig 1. Cloud Computing Overview.

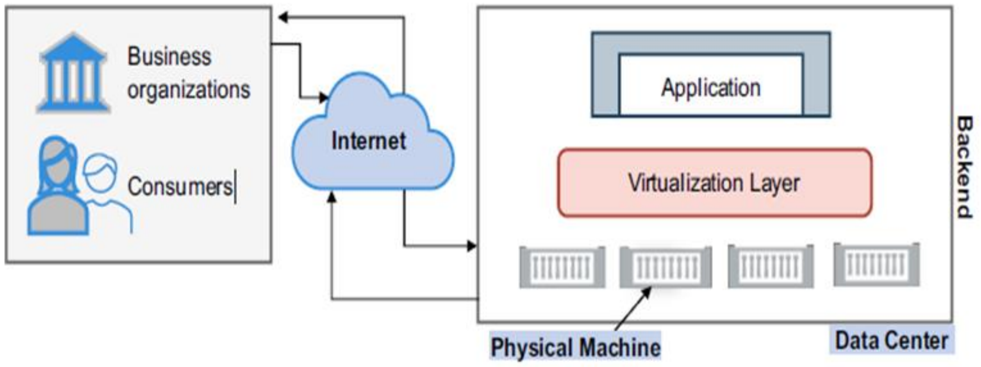


Fig 2. Architecture for Cloud Computing.

The application dynamically schedules incoming user requests, and virtualization assigns client resources. The virtualization technology balances the demand on the entire system, which manages dynamic resources in the cloud. Additionally, it supervises scheduling [16] and effective resource allocation. The user submits requests online, and virtual machines store them (VMs). Every delivery model requires CSPs to maintain QoS by guaranteeing that user-submitted bids may be processed and completed within a specific time frame [17]. A scheduling strategy (Data Broker), which is used to assign user tasks to the appropriate VMs, should be able to produce a workload that is evenly distributed among the machines and servers. Designing and creating a dynamic load balancer will enable efficient resource scheduling and utilization.

1.2. A Concept of Cloud Load Balancing

To process requests more quickly and send responses, a load-balancing method divides incoming traffic among available servers. Figure 3 illustrates the application of the traditional load-balancing architecture in a cloud environment. It involves the usual processes shown below: Fielding inquiries from a wide range of customers seeking assistance, a server monitoring daemon routinely monitors the server pool's load status, determines the optimal load size based on a client's load request, and determines which servers to use based on a load-balancing technique, algorithm, or heuristic. Millions of data packets are routed by sophisticated computer network systems each second. This large amount of data traffic needs to be appropriately distributed across the available servers in order to be handled without compromising the end-user experience. The goal of cloud computing designs is rapid elasticity, or the ability to grow or contract in response to demand. This implies that regardless of the program, it will eventually need to scale to fulfill user needs. Therefore, a mechanism that distributes the requests among many application instances must exist. A load balancer is provided [18].

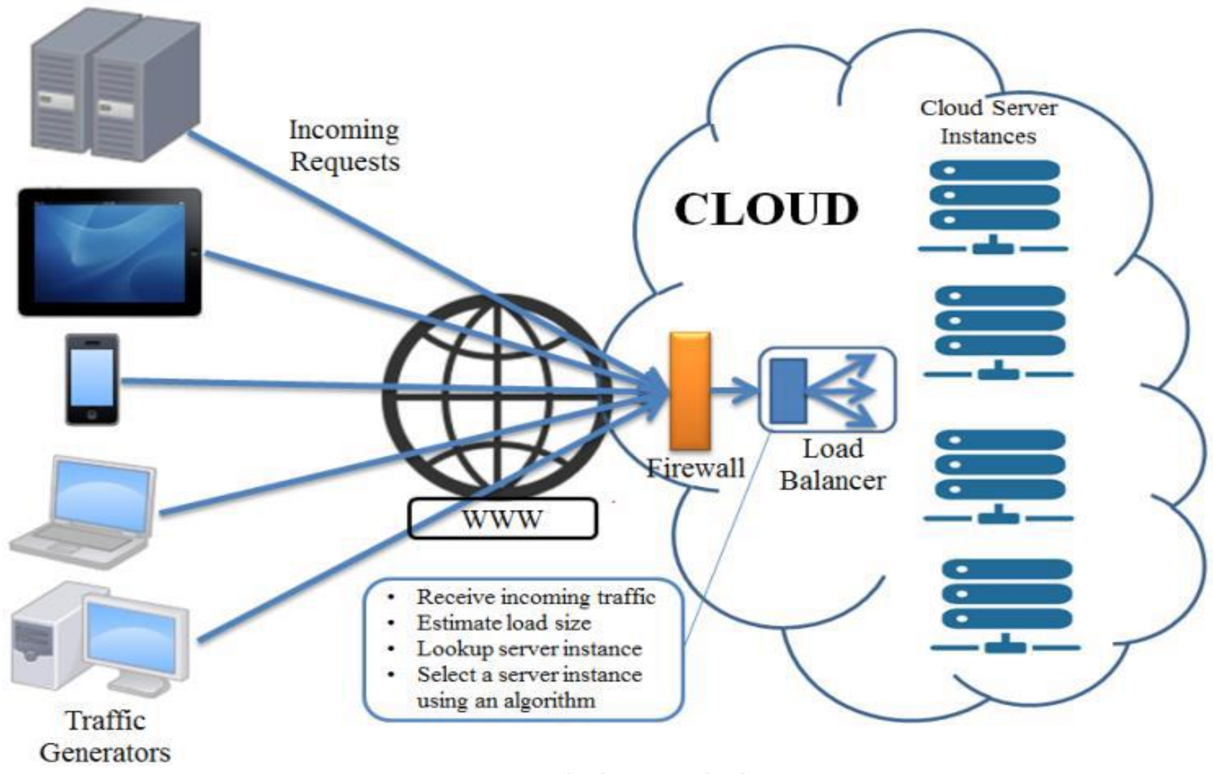


Fig.3: Load Balancing in Cloud

The existing state of the system served as the basis for the development of the load-balancing algorithm (LBA), which can be seen in Figure 4. The existing load balancing solutions can be grouped into six categories [19].

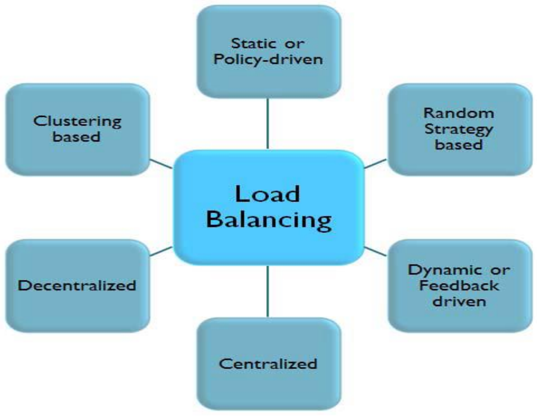


Fig 4. Classification of load balancing solutions [19]

1.2.1 Static Load Balancing

Static load balancing, also known as policy-driven load balancing, employs a predetermined collection of regulations and principles. The rules and policies are determined by different factors, including server capacity, availability, response time, resource usage, and fault tolerance. Typically, load-balancing decisions are made based on preset thresholds, maximum and minimum limitations, and other constraints [19]. Static load balancing assigns tasks to workstations depending on load distribution at compile time, allowing the scheduler to determine where each job will be executed. Nevertheless, static load balancing methods had a drawback in that tasks were unable to be transferred during their execution to another computer for the purpose of load balancing [20].

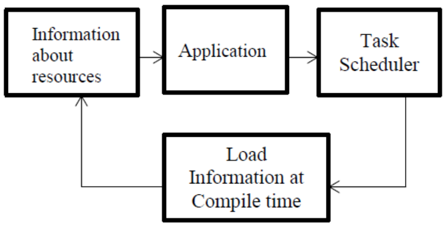


Fig 5. Working of Static Load Balancing Algorithm [20].

1.2.2 Dynamic Load Balancing

Unlike SLB algorithms, it distributes client requests across the available resources during runtime. As demonstrated in According to the dynamic information obtained from all resources in Figure 6, the LB allocates the request.

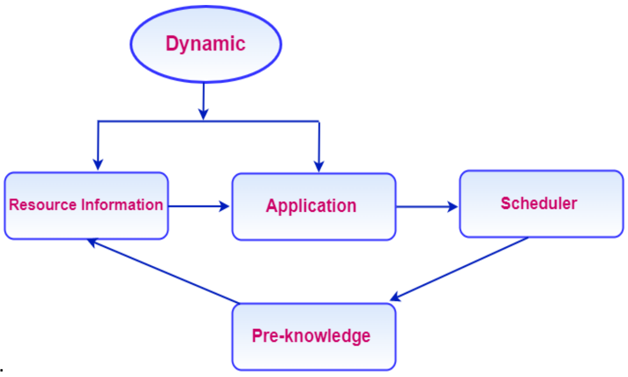


Fig 6. Dynamic load balancing

There are two distinct varieties of DLB algorithms, distributed and non-distributed varieties. The load balancing in a distributed DLB system is the task of the computer. Load balancing is a group effort that relies on everyone's contributions. In contrast, each resource is employed separately to complete the task in non-distributed algorithms. As a result of their interaction with all of the resources, distributed DLB methods frequently contributed to a greater amount of message overhead than non-distributed DLB methods [21]. Distributed algorithms perform better in failure scenarios because they only influence certain parts of the system rather than the entire system. Non-distributed algorithms can be further broken down into centralized and semi-centralized categories. A single machine handles the load-balancing process in a centralized algorithm. Semi-centralized architectures use server clusters with centralized load balancing [22].

2.Literature Review

In a Software Defined Network (SDN) context, Anis et al. [7] discuss the many load-balancing algorithms that may improve resource usage and linear service delivery among several customers. The authors have found a crowd-based method to satisfy users' needs to continue providing service. This is accomplished by evenly distributing the workload to achieve maximum resource usage and minimize idle time. In software-defined networks, the control and administration planes are separated from the data plane by an abstraction layer called the controller. They are both a part of a larger whole called an application layer.

- Hui et al. [8] An effective information duplication technique is required to lessen the effort and improve the system's functionality. This paper develops a task loading model that considers the needs of energy-sensitive and latency-sensitive activities along with the overall load dynamics in the cloud, edge, and end layers to minimize long-term task delay and energy consumption. A dynamic duplicate deployment technique was created in the last stage to enhance access performance and load balancing amongst service nodes. As a result, the recommended method enhances load balancing, efficiency, and accessibility in a hierarchical cloud computing environment.
- Chunlin et al. [9] The geo-distributed cloud is a could deployed in different geographical locations. It is a promising solution to deal with the data explosion. Google’s 13 data centers are deployed in 4 continents over 8 countries. In order to enhance the efficiency of utilizing a geographically distributed cloud, user requests are tailored to suit the various data centers. Geo-distributed cloud computing offers superior processing capability and storage space compared to standard cloud computing, ensuring a higher quality of service. Therefore, it is crucial to investigate the scheduling technique for workflow applications in order to enhance the efficiency of geo-distributed clouds.
- LINJIE et al. [10], In order to reduce long-term task latency and energy consumption, this study develops a task loading model that takes into account the demands of energy-sensitive and latency-sensitive jobs as well as the overall load dynamics in the cloud, edge, and end layers. To choose the optimal edge server or cloud server for loading, a method called TOLBO is suggested, which is based on deep reinforcement learning (DRL). The approach outperforms competing algorithms in terms of energy utilization on cloud edge nodes, according to simulation data. Concurrently, it has the potential to drastically cut down on end device energy consumption, average latency, and job throw rate.

- Santosh T. Waghmode, Bankat M. Patil. [11]. The technologies of distributed computing, server virtualization, and network storage have converged to form the cloud computing platform. The fundamental goal of cloud computing is to deliver a variety of IT solutions by organizing and configuring a collection of computing resources, enhancing the burden for users, and enabling them to concentrate on their core operations. Cloud computing encompasses the attribute of scalability. The primary objective is to introduce a novel load-balancing method capable of evenly distributing incoming requests from users worldwide, who are located in various locations, and are seeking data from faraway data sources. This approach will integrate efficient scheduling with cloud-based methodologies. A novel approach for load balancing was devised to guarantee fast responsiveness of cloud environments, while also optimizing the utilization of cloud resources and accelerating job processing durations. Elastic load balancing automatically distributes incoming traffic from applications over several targets, such as Amazon EC2 instances, network addresses, and other entities. The results indicate that the suggested methodology performs effectively in a dynamic cloud setting, where user requests arrive in a planned sequence and steadily expand in length. When comparing the suggested strategy with the current approach, the algorithm is capable of responding to large-scale requests.

3. Mape-K Algorithm

The Karnaugh map, often known as KM or K-map, is a technique used to streamline Boolean algebra statements. Maurice Karnaugh developed the concept in 1953 as an improvement upon Edward W.'s original idea. The Karnaugh map exploits people' capacity to recognize patterns, hence minimizing the requirement for complex mathematics. Additionally, it allows for the quick detection and resolution of possible race circumstances [23].

MAPE-K loops to handle this intricacy [10]. Assess and evaluate Closed feedback loops called Plan Execute-Knowledge (MAPE-K) loops are capable of managing the intricacies of self-adaptation. In a more recent work, [11] provided frameworks for applying MAPE-K control loops to diverse distributed apps. Most adaptive optimization algorithms are implemented ad hoc in simulations and cannot be recycled from earlier iterations. As a general approach, we suggest using a MAPE-K control loop, which will enable the implementation and upkeep of traditional adaptivity schemes [24].

When addressing digital circuits and practical difficulties, it is sometimes necessary to identify expressions that include the fewest variables possible. Boolean expressions of 3 or 4 variables may be efficiently minimized using K-map, without the need for any Boolean algebra theories. The K-map can be represented in two forms: Sum of Products (SOP) and Product of Sums (POS), depending on the problem's requirements. A K-map is a tabular representation that provides more comprehensive information compared to a truth table. The K-map is populated with binary values of 0's and 1's, and subsequently solved by forming groups [26].

The k-means algorithm's fundamental principle is to use the average value of the data samples in each cluster subset as the cluster's representative point and iteratively refine it. The procedure separates the data set into various groups in order to assess clustering. Each cluster can arise when the energy criteria function reaches its ideal value. compact and class-independent. During the iterative process, each class makes use of the optimal cluster set, which is reached by continuously moving the items in the clustering set. The average value of the items is expressed. The cluster formed through the application of the k-means algorithm has a high level of similarity among its constituent items, while simultaneously demonstrating a significant degree of dissimilarity when compared to objects belonging to other clusters.

3.1 The simulation method

A comparison can be made with different load-balancing algorithms and methods in cloud computing settings using simulation tools such as the ACSIM framework. To create a simulated cloud environment, Virtual machines, servers, and network components are used, through which performance analysis tools can be used to balance different loads in this simulated environment.

Overall, while the ACSIM framework was not explicitly designed for cloud computing load balancing, the effectiveness of load-balancing algorithms in cloud computing settings could be assessed using its simulation and analysis tools. However, the accuracy and applicability of the simulation results would depend on how well the simulated environment reflects the actual cloud environment being analyzed.

The simulation method of adaptivity in distributed agent simulation will address computational cost reduction, execution time, and imbalance through the dynamic exchange of ACSIM distribution across multiple computational units and the optimum use of resources. Also, it shows the type of simulation geometry required in a clear and reusable way. Figure (7) shows the general scheme of the simulation method, which includes six steps.

Step 1: Initial ACSIM

ACSIM is a distributed framework-based Python programing language figure. The work on this framework is carried out dynamically by implementing a local server. The admin manager is generated in this sage based on the users within the range (of 50 – 5000). Also, four types of admin managers are developed, two types of intruders and the other two regular ones on which the balancing process is carried out.

Step 2: Adjust the ACSIM setting.

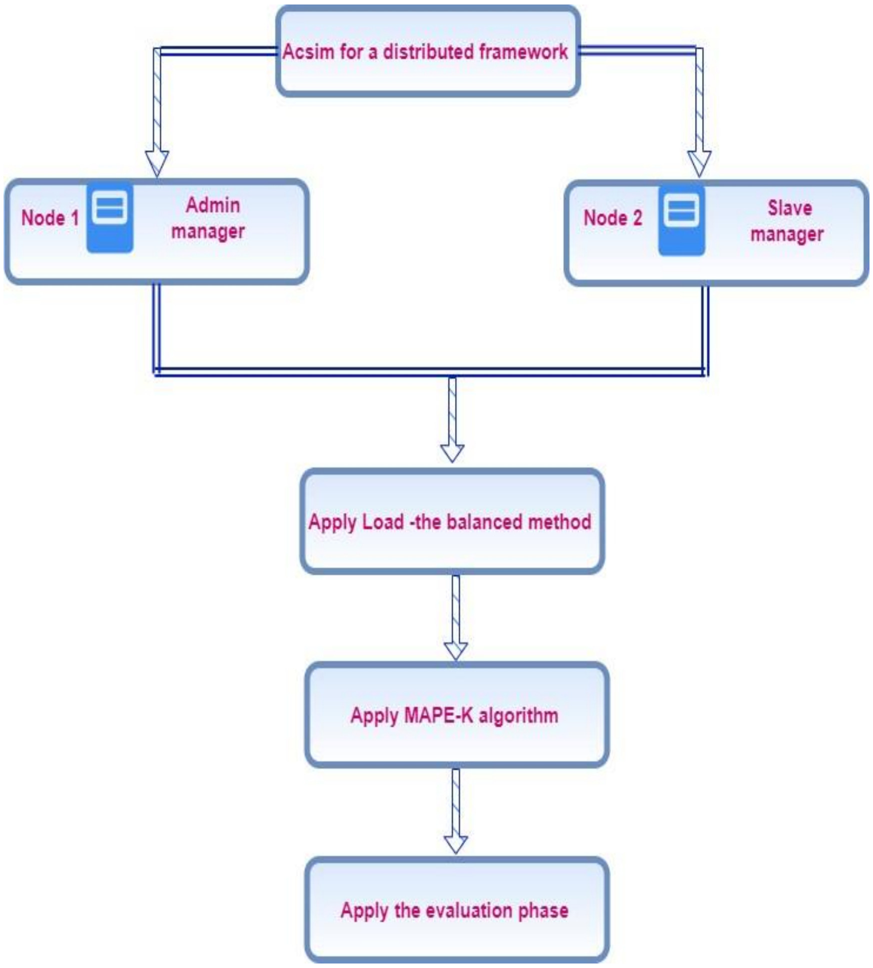
ACSIM settings and the client's movement are adjusted at this step. So, for example, when the latter moves in three directions from its position, and when moving to the new campaign, three new activities are selected, and so on. And these movements are all random.

Step 3: Executing the method.

In this step, the framework within which the client moves is determined.

Step 4: Get rid of intruders

This step includes that the user identifies the intruder client (irregular) , the intruder's movements are processed and converted into regular (for example, the intruder's strength is 10 movements and then weakens, where the intruder is killed and turned into a common node).



Acsim for a distributed framework
Node 1
Admin manager
Node 2
Slave manager
Apply Load -the balanced method
Apply MAPE-K algorithm
Apply the evaluation phase

Fig 7. The simulation method diagram

Step 5: Map-k apply.

In this step, the balancing of the two regular categories and attempt to make them both equal. For example, one category includes 50 admin managers, while the other contains 20 Slave mangers, they are summed and divided by 2, and the result is 35 nodes for each category), And as figure 8 shows.

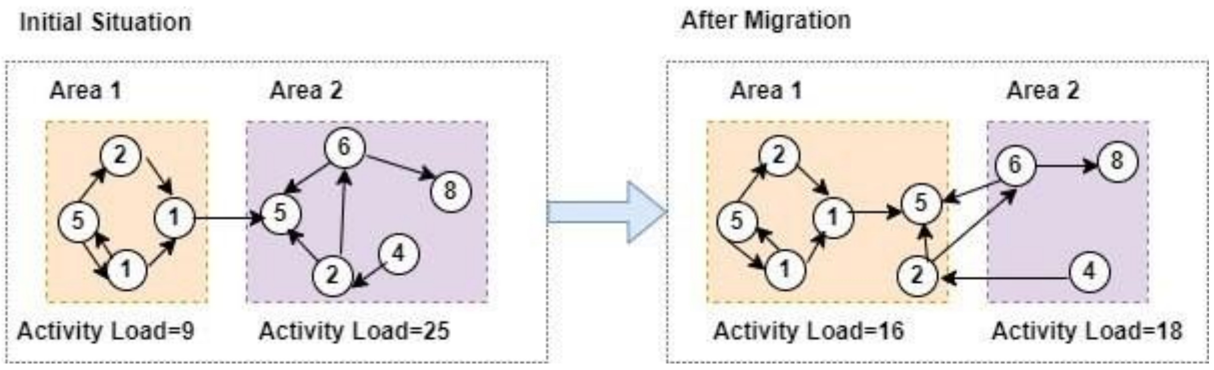


Fig 8. load balancing

Step 6: Evaluation

In the last step, the evaluation is done as a real action instead of storing previous data. The evaluation here is achieved based on accurate results depending on two categories, regular and intruder, where the intruder category reduces to zero while the regular is balanced and distributed. The research revealed that the utilization of ACSIM led to a notable enhancement in the efficacy and capacity of the MAPE-K Algorithm in both of its implementations. The load-balancing methodology implemented in ACSIM ensured equitable distribution of workload among the nodes, thereby mitigating the likelihood of performance deterioration resulting from excessive loading. The research findings suggest that the utilization of ACSIM has the potential to enhance the efficiency of autonomic computing systems that necessitate load balancing. According to the study, the implementation of ACSIM (Autonomic Computing System Integrator and Modeler) resulted in a notable enhancement in the performance and scalability of the MAPE-K Algorithm. The load-balancing methodology implemented in ACSIM facilitated the equitable allocation of workload among nodes, thereby mitigating the likelihood of performance-related complications arising from excessive loading. The study's conclusion suggests that ACSIM could be a valuable tool for optimizing the performance of autonomic computing systems that require load balancing.

3.2. Load Balancing Algorithm

Load and load performance are the primary metrics used to evaluate the impact of load balancing. The CPU queue index and CPU usage are two metrics used to assess the level of workload. Performance is the duration it takes for a user to obtain a response. Several parameters are inputted into the load-balancing method, including the configuration of the virtual machine (VM), the arrival time, and the characteristics of the cloudlet application (such as length, completion time, and expected completion time) [26].

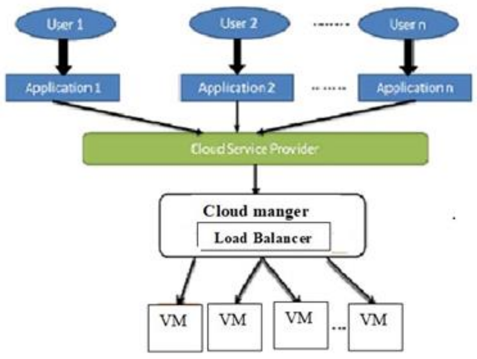


Fig 9. The load balancer's architecture in cloud computing.

The following formula is used to computing the expected response time for task [26]:

Expected Response Time (ERT) = TC – TA + TT …………(1)

where:
TC: The Complete of Time.
TA: The Arrival of Time.
TD: Transfer of Time (delay time).

The Datacenter Broker algorithm is responsible for performing load balancing, and its efficiency directly affects the processing time in environment of data center. The communication delay has been eliminated completely. Calculate the anticipated duration of task completion:

The scheduling strategy may be either Timesharing-Space sharing or Space Sharing-Timesharing. Next, the calculation is performed using the formula provided in equations (2), (3):

$$eft(p) = est + \frac{ri}{capacity * cores(p)} \quad \ldots\ldots\ldots\ldots\ldots\ldots (2)$$

Formula (3) used to compute the capacity parameter [9]:

$$capacity = \sum_{i=1}^{np} \frac{cap(i)}{np} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (3)$$

Else when the scheduling policy is Space Share-Timeshare or Timeshare-Timeshare, then the calculation according to the formula which defined in (4), (5):

$$eft(p) = ct + \frac{ri}{capacity * cores(p)} \quad \ldots\ldots\ldots\ldots\ldots\ldots (4)$$

Formula (5) used to compute the capacity parameter [10]:

$$capacity = \frac{\sum_{i=1}^{np} cap(i)}{max(\sum_{j=1}^{cloudlets} cores(j), np} \quad \ldots\ldots\ldots\ldots\ldots\ldots(5)$$

For each formula in (2), (3), (4) and (5), that and for Cloudlet p:
eft(p): is the expected completion time.
est: The arrival time.
rl: The total number of instructions that are to be executed on a processor.
capacity: average processing power (in MIPS) of a core.
ct: current simulation time.
Cores (p): refers to the number of cores required.
np: actual number of core that the host is considered.
Cap: processing power of the core [26].
Therefore, in accordance with these equations and algorithms, the system as a whole has the ability to make things faster and more efficient.

4.Results of The Experiment

In this experiment, the following variables were used to create a cloud: 30 cloudlets were considered a task, and information was obtained via multimedia streaming (multiple files).
The results of related work are shown in the tables below: the reaction time is shown in the first table when Load Balancing adaptive is not utilized, and the same results are shown in the second table when Load Balancing adaptive is. Both tables are shown below. Case study of the associated work performed for a few servers. Show Table 1, and Table 2.

Table 1. Without load balancing approach

No.	Start (MS)	Process Time (MS)	Response Time (MS)
1	821.2013	821.201	0.000803
2	126.9081	126.9197	0.010624
3	299.9316	299.9335	0.011637
4	448.5049	448.5185	0.01162
5	764.0457	764.0622	0.0145

Table 2. With load balancing

No.	Start (MS)	Process Time (MS)	Response Time (MS)
1	507.9036	507.92045	0.00113
2	411.9347	411.94165	0.01105
3	262.845	262.83391	0.01107
4	448.5049	448.5203	0.011102
5	746.0457	746.0622	0.011566

The findings from the previous tables (reference 25) indicate that there exist significant disparities in response time between the utilization of Throttled Load Balancing and the absence of the algorithm, particularly in the context of multiple mobile devices. Therefore, it is crucial to handle distributed time allocation for each device within the Mobile Cloud effectively. The goal of the load balancing algorithm is to distribute the load among numerous servers, which reduces the response time of Node. The findings demonstrate the analytical efficacy of time division through the utilization of multiple virtual machines. The experimental findings suggest that the Algorithm exhibits a degree of success. Consequently, the utilization of Load Balance Algorithm confers a tangible benefit. In the context of Mobile Cloud environments, load balancing algorithms aim to prevent delays and minimize time waste across both types of research. Results show the logical performance of time division via multiple VMs. The results of the experiment indicate that the algorithm could be considered successfully.

5. Conclusion

Mobile Cloud Computing (ACSIM) is a concept that combines the advantages of cloud computing and smartphone. The use of ACSIM allows mobile devices to offload data-intensive and computationally complex operations to the cloud, leading to improved processing efficiency thanks to the superior computing power and storge capacity of cloud servers. ACSIM can improve the efficiency of mobile applications by offloading the processing and storge of data to the cloud. When the quantity of mobile apps rises, cloud computing power becomes significant. In order to enhance the performance of this apps, the computer resources will also be effectively utilized to manage their resources. Performance will improve as a result of jobs being more responsive. As a result, it is challenging to plan tasks in a way that both maximizes resource usage and job responsiveness.

References

[1] B. Ranjan Parida, Amiya Kumar Rath, Hitesh Mohapatra," Binary Self Adaptive Salp Swarm Optimization-Based Dynamic Load Balancing in Cloud Computing", International Journal of Information Technology and Web Engineering (IJITWE) 17.1 (2022): 1-25.

[2] Srinivasa Rao Gundu, Charan Arur Panem, Anuradha Thimmapuram, "Real-Time Cloud-Based Load Balance Algorithms and an Analysis", SN Computer Science 1.4 (2020): 1-9.

[3] Walaa Saber, Walid Moussa, Atef M. Ghuniem, Rawya Rizk," Hybrid load balance based on genetic Algorithm in cloud Environment", Vol. 11, No. 3, June 2021, pp. 2477~2489 ISSN: 2088-8708, DOI: 10.11591/ijece.v11i3.pp2477-2489.

[4] Nicola Sfondrini, Gianmario Motta, "SLA-aware broker for Public Cloud", In 2017 IEEE/ACM 25th international symposium on quality of service (IWQoS), Vilanovai la Geltru; 2017, pp. 1–5.

[5] Ma Chen, Yuhong Chi, "Evaluation Test and Improvement of Load Balancing Algorithms of Nginx." IEEE *Access* 10 (2022): 14311-14324.

[6] Rajagopalan S, "An Overview of Server Load Balancing." International Journal of Trend in Research and Development 7.2 (2020): 231-232.

[7] Anish Ghosh, Mrs T. Manoranjitham, "A study on load balancing techniques in SDN ", International Journal of Engineering & Technology, 7 (2.4) (2018) 174-177

[8] Hui-Ching Hsieh, Mao-Lun Chiang," The Incremental Load Balance Cloud Algorithm by Using Dynamic Data Deployment", https://doi.org/10.1007/s10723-019-09474-2.

[9] Chunlin Li, Jianhang Tang, Tao Ma, Xihao Yang, Youlong Luo, "Load balance-based workflow job scheduling algorithm in the distributed cloud", Journal of Network and Computer Applications (2020).

[10] Yan, Linjie, et al. "A Task Offloading Algorithm With Cloud Edge Jointly Load Balance Optimization Based on Deep Reinforcement Learning for Unmanned Surface Vehicles." IEEE Access 10 (2022): 16566-16576..

[11] Santosh T. Waghmode, Bankat M. Patil, "Adaptive Load Balancing Using RR and ALB: Resource Provisioning in Cloud" ISSN: 2321-8169 Volume: 11 Issue: 7 DOI: https://doi.org/10.17762/ijritcc.v11i7.7940 Article Received: 08 May 2023 Revised: 26 June 2023 Accepted: 12 July 2023

[12] Agarwal, Mohit, and Gur Mauj Saran Srivastava. "Cloud computing: A paradigm shift in the way of computing." International Journal of Modern Education & Computer Science 9.12 (2017).

[13] Abdalla, Peshraw Ahmed, and Asaf Varol. "Advantages to disadvantages of cloud computing for small-sized business." 2019 7th International Symposium on Digital Forensics and Security (ISDFS). IEEE, 2019.

[14] Lowe, D., and B. Galhotra. "An overview of pricing models for cloud services with analysis on a pay-per-use model." International Journal of Engineering & Technology 7.3.12 (2018): 248-254.

[15] Odun-Ayo, Isaac, et al. "Cloud computing architecture: A critical analysis." 2018 18th international conference on computational science and applications (ICCSA). IEEE, 2018.

[16] Gupta, Indrajeet, Madhu Sudan Kumar, and Prasanta K. Jana. "Efficient workflow scheduling algorithm for cloud

computing system: a dynamic priority-based approach." Arabian Journal for Science and Engineering 43.12 (2018): 7945-7960.

[17] Adhikari, Mainak, and Tarachand Amgoth. "Heuristic-based load-balancing algorithm for IaaS cloud." Future Generation Computer Systems 81 (2018): 156-165.

[18] Rajat, Dr Sanjeev Kumar, "Performance Comparison of Load Balancing Architectures in Cloud Computing Environment.", Volume 8 • Issue 2 March 2017 – Sept. 2017 pp.. 186-193.

[19] Mazedur Rahman, Samira Iqbal, and Jerry Gao. " Load Balancer as a Service in Cloud Computing." Linked Open Data-Applications, 978-1-4799-3616-8/14 $31.00 © 2014 IEEE DOI 10.1109/SOSE.2014.31.

[20] Jaimeel M Shah , Sharnil Pandya , Narayan Joshi, "Load Balancing in cloud computing: Methodological Survey on different types of algorithm", 978-1-5090-4257-9/17/$31.00 ©2017 IEEE .

[21] Kumar, Pawan, and Rakesh Kumar. "Issues and challenges of load balancing techniques in cloud computing: A survey." *ACM Computing Surveys (CSUR)* 51.6 (2019): 1-35.

[22] Puthal, Deepak, et al. "Secure and sustainable load balancing of edge data centres in fog computing." IEEE Communications Magazine 56.5 (2018): 60-65.

[23] Ebadifard, Fatemeh, and Seyed Morteza Babamir. "A PSO-based task scheduling algorithm improved using a load-balancing technique for the cloud computing environment." Concurrency and Computation: Practice and Experience 30.12 (2018): e4368.

[24] Negin Najafizadegan, Eslam Nazemi, Vahid Khajehvand, "A MAPE-K Loop Based Model for Virtual Machine Consolidation in Cloud Data Centers", Journal of Computer & Robotics 13(2), 2020 33-60.

[25] Stig Bosman, Toon Bogaert, Wim Casteel, Siegfried Merceli, Joachim Denil, and Peter Hellinckx," Adaptivity in distributed agent-based", 2020 Antwerp,

[26] Karim Q. Hussein, "A Multimedia Information Time Balance Management in Mobile Cloud Environment Supported by Case Study", DOI: https://doi.org/10.3991/ijim.v16i19.33615, VOL. 16 NO. 19 (2022)