SLUJST
Sule Lamido University Journal of Science & Technology Vol. 7 No. 2 [December, 2023], pp. 52-67
https://doi.org/10.56471/slujst.v7i.475
Online ISSN: 2736-0903
Print ISSN: 2736-089X
www.slujst.edu.ng

An Improved Crow Search Load Balancing Algorithm Using Threshold-Based Strategy for Virtual Machine Allocation

Mathias Pama Garkuwa[1*], Rabi Mustapha[2], Khalid Haruna[3]
[1,2,3]Department of Computer Science, Kaduna State University, Kaduna State, Nigeria.,
mathiasgarkuwa@gmail.com[1*], rabichubu@kasu.edu.ng[2], khalid.haruna@kasu.edu.ng[3]

[*]Corresponding Author

Abstract

Background: *Load balancing is the allocation (distribution) of workload among a set of computational elements (CEs). In large-scale distributed computing systems, CEs are physically or virtually distant from each other. Communication among CEs is associated with delays that can significantly alter the expected performance of load-balancing policies. This is a prominent problem in computing systems for which the individual units are connected by means of a shared communication medium such as the Internet, ad-hoc networks, wireless Local Area Networks.* ***Aim:*** *This study is aimed at providing an improved Crow search load balancing algorithm (ICSLBA) to enhance performance, simulate and evaluate the proposed algorithm using some performance metrics.* ***Method****: the existing Crow Search Algorithm is improved by introducing a Threshold-based strategy and simulated using a cloud simulator with an aim to enhance resource allocation in a cloud computing environment.* ***Results****: The results achieved through experimentation have shown that; the proposed ICSLBA has out-performed the CSLBA in terms of improving Average Makespan Time (AMT), Average Waiting Time (AWT), and Average Data Center Processing Time (ADCPT). Hence, the proposed ICSLBA is the optimal load-balancing strategy.*

Keywords: *Cloud computing, Crow Search, Virtual Machine, Load balancing Algorithms, Threshold-Based Strategy*

1. Introduction

With the rapid development of processing and storage technologies and the success of the internet, computer resources have become cheaper, more powerful, and more accessible than they were before. This technological trend has enabled a new realization of a computing model called cloud computing. Cloud computing provides simple, on-demand network access to a shared pool of virtual computing resources such as Networks, Servers, Storage, Applications, and Services that can be instantly supplied and released with minimum management effort or service provider involvement" (Singh et al., 2019). Cloud computing enables users and entrepreneurs to use cloud resources without installing or creating infrastructures, as well as to access their applications and data from any location through the Internet at any time (Thanka et al., 2019).

The cloud concept is an evolutionary approach in computing environment that combines the resources of different computers to function as a single entity. These resources (hardware and software) are shared between different users through the internet and managed by the provider. Cloud Computing integrates a number of computing concepts such as Service Oriented Architecture (SOA), Web 2.0, virtualization, and other technologies that rely on the Internet. It combines them as a paradigm to deliver on-demand resources (e.g., infrastructure, platform, software, etc.) to customers. The term “cloud” computing is metaphorically used to describe the virtualized nature of services that cloud computing provides. The term cloud was first used by telecommunication companies, describing the virtual private network services (VPN) provided in the 1990s. However, the concept dates back to the 1960s when John McCarthy introduced it, whereby later in 1966 Douglas Parkhill characterized the characteristics of the technology and set the first known architecture of the technology in his book "The Challenge of the Computing Utilities” (Swati & Ankur, 2015).

Sule Lamido University Journal of Science & Technology Vol. 7 No. 2 [December, 2023], pp. 52-67
https://doi.org/10.56471/slujst.v7i.475
Online ISSN: 2736-0903
Print ISSN: 2736-089X
www.slujst.edu.ng

Over the years many companies such as Amazon (Amazon Web Services), Google, and IBM have researched and invested billions in the field of cloud computing. These companies provide many cloud computing-based services such as Infrastructure as a Service (IaaS), Software as a Service (SaaS), and, Platform as a Service (PaaS) (Sova, 2016). With the invention of the internet and Moore's law, Cloud computing technology grew, many companies started providing cloud computing services depending on the area and the nature of the service, despite the increase of service providers in the field, the number of individuals requesting the services increases rapidly, and that poses the question of performance decline (Abey & Cyril, 2019). The introduction of the load balancing algorithm has simplified many problems such as the information lost due to lack of bandwidth and queue.

Due to unique features of cost-effectiveness, stability, and scalability. The primary goal of cloud computing is to implement economies of scale by enabling access to high processing power and extensive storage capacity. Cloud service providers maintain and manage several data centers in different geographical regions. This implies that there should be a system in place to ensure that workloads and resources are shared efficiently. Therefore, cloud service providers implemented load balancing as a service to facilitate workload dispersion across various data centers, servers, PCs, and networks. The performance of a cloud computing environment is mostly determined by the scheduling method employed inside it. Scheduling in the cloud is a procedure that maps a collection of received workloads to a collection of virtual machines capable of executing them, or alternatively, assigning virtual machines with available resources to satisfy user needs. Cloud assets are more often than not, underutilized as a result of poor mapping between jobs and data center resources. Resource optimization must be accomplished by effective scheduling, which would boost the performance efficiency of the cloud environment. (Pradeep, 2021).

The load balancing refers to, the technique that strives to balance the workload across virtual machines (VM) and eliminates the status of some of the nodes or VMs being overloaded while some others are less loaded or idle. Resources scheduling and load balancing in cloud computing environment is one of the most important challenges for experts and researchers who have attempted in their research to investigate the problems that occur during running (executing) applications (Abhijit et al., 2016).

2. Materials and Methodologies

Cloud computing has long been a popular topic of study. Numerous studies have been conducted on topics related to cloud computing which include: Performance of Cloud computing technology, Cloud Security, Data backup, Data segregation and Recovery, Safe architecture, and a wide range of other topics. Many researchers in the field of computer science have argued about how to improve and gauge performance. This has led to several algorithms employed, that have proven to be successful in many applications for maintaining performance of cloud computing systems. However; each research utilizes a distinct set of criteria to assess performance efficiency (Niloofar & Reza, 2013).

Saravanan et al. (2023) study introduces Improved wild horse optimization algorithm with Levy flight theory (IWHOLF-TSC) approach. The proposed approach was designed to address the issues of long scheduling times, high costs and high virtual machine load in cloud computing task scheduling. The technique is empirically examined and compared with others algorithm such as the Ant colony algorithm (ACO), Particle Swarm Optimization (PSO), and Whale Optimization algorithm (WOA). To model the cloud computing environment, the Cloud-Sim simulator was employed. The findings reveal that the algorithm performs well across a wide range of job quantity situations. Nevertheless, the end result is more efficient at task scheduling and less expensive than other approaches. Even though future improvement is welcome.

Singh et al. (2023) presented a scheduling technique for Big Data applications based on Docker Swarm and Microservice architecture. Bearing in mind that Big Data applications need considerable resources and settings in order to store, process, and analyze a dispersed collection of data. Containerization with cloud computing is a viable solution for accommodating massive data requirements, but requires a precise and effective load-balancing technique. The findings indicate that rising workloads for big data applications may be efficiently controlled by leveraging microservices in containerized systems and Docker Swarm is used to efficiently achieve load balancing. The study

Sule Lamido University Journal of Science & Technology Vol. 7 No. 2 [December, 2023], pp. 52-67
https://doi.org/10.56471/slujst.v7i.475
Online ISSN: 2736-0903
Print ISSN: 2736-089X
www.slujst.edu.ng

focuses on enhancing performance amid greater workloads caused by big data processing and scaling services to increase efficiency.

The study Kumar, (2023) proposes a novel algorithm for resource allocation in a fog computing environment using a hybrid crow search mechanism and a multi-objective population-based evolutionary mechanism. The proposal considers two goals: the security hit rate and the success rate. The study uses evolutionary method to figure out how to schedule and divide resources in a fog computing environment. The study result effectiveness was the contrast between hybrid CSA, original CSA and Genetic Algorithm (GA). The study's outcome shows how well the proposed algorithm accomplishes its objectives.

Le & Tran (2022), presented an overview of existing and frequently used load-balancing technologies, as well as research strategies for improving cloud computing performance. The proposed improved Throttled algorithm is to increase task processing and response time, which was done by modeling with the Cloud Analyst GUI tool. Simulation results were compared with algorithms such as Equally Load, Round Robin, and Throttled modified Algorithm. The results revealed that the enhanced Throttled Algorithm (ITA) reduced data center costs, enhanced load balancing, and accelerated task and request processing times.

Cloud computing suffers from request overload. load balancing task in a cloud setting has become an essential part of spreading resources from a data center. Jena et al. (2022) suggested a unique mechanism for dynamic load balancing across virtual machines using the hybridization of modified Particle swarm optimization (MPSO) and an improved Q-learning algorithm. With the goal of improving machine performance by balancing the load among the VMs. The simulation results obtained with CloudSim 3.0.3 reveal that the suggested approach outperforms the other algorithms examined. However, leaves room for dynamic load balancing among the dependent tasks.

The study of Krishnadoss et al., (2021**)** presented a proposed efficient hybridized scheduling algorithm that replicates the parasitic behavior of the cuckoo and food gathering habit of the crow bird, named Cuckoo Crow Search Algorithm (CCSA) for improvising the task scheduling process. The proposed work was set to address the task scheduling issue in cloud computing, and reduce Makespan and cost. Results obtained using hybrid CCSA were compared with MO-ACO, ACO, and Min-Min algorithms. It shows that the proposed hybrid CCSA enhanced the system performance compared with other algorithms.

Mishra and Panda (2020) recognized that cloud computing offers a well-liked outcome for providing easy access to resources from an external source. And virtualization idea was extremely beneficial, which resulted in a wide range of applications. For cloud computing facilities to effectively offer service, various resources, such as a virtual machine, CPU, and memory, among others, must be allocated correctly to reduce energy usage anytime. However, a need for Cloud simulation and modeling tool is a necessity. CloudSim and Cloud report tools were used to investigate and simulate the cloud environment for better understanding. CloudSim was an appropriate key for modeling the cloud as well as extending in and out of the infrastructure requirement. Cloud reports became the appropriate solution for pricing infrastructure, resource usage, and power consumption in updated environments.

Mishra et al. (2020) outlined several homogeneous and heterogeneous load-balancing approaches in cloud computing systems. The study used performance metrics to describe diverse system performances, suggested a classification system for load balancing algorithms, and estimated energy usage in the cloud environment. To evaluate the performance of heuristic-based algorithms, the CloudSim simulator was employed. However, the suggested algorithms have not yet been tested in a real-world cloud deployment. However, understanding the methodologies is vital for future study.

To really understand cloud computing inefficiency challenges, the study of Mohit and Bharat (2020) emphasizes the load-balancing issues by carrying out a comparison analysis of several load-balancing strategies. Summarized information about load balancing techniques highlighted some specified criteria used in evaluating various algorithms such as throughput, response time, fault tolerance, etc. which are the basis for determining the effectiveness of algorithms. Haven discussed the benefits and drawbacks of several load-balancing strategies. The study conclusion revealed a number of predefined methods that may be coupled to improve load balancing.

SLUJST
Sule Lamido University Journal of Science & Technology Vol. 7 No. 2 [December, 2023], pp. 52-67
https://doi.org/10.56471/slujst.v7i.475
Online ISSN: 2736-0903
Print ISSN: 2736-089X
www.slujst.edu.ng

Singh et al. (2019) defined task scheduling as a vital aspect of computer science that enables the computer system to execute various tasks and perform related activities with accuracy and efficiency. The proposed Crow Search-based load balancing algorithm (CSLBA) concentrates on allocating the best suitable resources using various parameters such as average Makespan time (AMT), average waiting time (AWT), and average data center processing time (ADCPT). The findings using the CloudSim simulator offer a comparison with different algorithms, which demonstrated that the CSLBA is the best scheduling strategy when compared to others. In conclusion, it was recommended that to achieve a more efficient result, the proposed work Can be combined with other load-balancing meta-heuristic scheduling techniques.

A new load-balancing technique for cloud computing that leverages a water flow-like algorithm to improve the system's overall response time was suggested in the study of Alazzam et al., (2019). The suggested algorithm was assessed using metrics such as job migration, average response time, and throughput. The acquired results were compared to those produced by the Genetic Algorithm (GA), Round Robin (RR), and Min Min algorithms. The results indicated that the proposed WFLBA has a better performance than the compared algorithms.

The Artificial Bee Colony algorithm (ABC) and Particle Swarm Optimization (PSO) were hybridized to create the Artificial Bee and Particle Swarm (ABPS) algorithm. The goal of the proposed ABPS algorithm was to optimize task scheduling in a cloud environment by minimizing Makespan, cost, and resource use. The CloudSim simulation tool was used to evaluate the proposed ABPS algorithm against the ABC and PSO algorithms. The suggested algorithm performs better than the ABC and PSO algorithms (Thanka et al., 2019).

Kumar and Kumar (2019), conducted a thorough survey study on several load-balancing methods while taking different parameters. Load-balancing strategies were grouped into various groups: General Load-Balancing Techniques, Natural Phenomena–Based Load Balancing, Task-Based Load Balancing, and Agent-Based Load Balancing. From each category, advantages, disadvantages, concepts, (techniques), and challenges were discussed. The study indicated that; there are several load-balancing algorithms available that can integrate the majority of load-balancing parameters and provide higher resource use with less response time. However, there is a need to enhance the approaches for system performance and resource utilization in the future. The study helped researchers uncover research challenges and existing load-balancing approaches.

Sambangi and Gondi (2019) conducted a survey on the need to optimize performance in terms of execution time, response time, and resource utilization in the cloud. To achieve optimization of the aforementioned factors, an efficient Load balancing is required. The study highlights the recent research regarding the application of Load Balancing techniques for task allocation such as resource allocation (RA) strategies, cloud task scheduling centered on Load Balancing, dynamic Resource Allocation schemes, and cloud resource provisioning scheduling heuristics. Proffers a literature survey on various Load balancing techniques for task allocation in a Cloud environment. And finally, compared Load Balancing performance for task allocation methods based on task completion time. It suggested for more enhancement in the performance of the task allocation approaches.

Parida and Panchal (2018) identified load balancing to be the biggest obstacle in cloud computing, where the number of VMs assigned to user requests is either over-loaded or under-loaded. The study focuses on several algorithms that are modified versions of the throttled load balancing method. These algorithms are developed to produce better results and to address issues that have arisen in prior ones. They suggested that, future research might lead to the development of throttled load-balancing algorithms that produce better results.

Dhari and Arif (2017) in an effort to improve performance and stability of load balancing system. The Proposed Load Balancing Decision Algorithm (LBDA) was used to balance load between virtual machines in a data center, and reduce Makespan and Response time. The algorithm was based on three stages, first, to calculate the VM capacity and VMs' load states (Under-loaded VM, Balanced VM, High-Balance VM, and Overloaded). Second, to calculate the time required to execute the task in each VM. The results indicated that the proposed LBDA is more efficient than the existing Max- Min, Shortest Job First, and Round Robin algorithms.

Sule Lamido University Journal of Science & Technology Vol. 7 No. 2 [December, 2023], pp. 52-67
https://doi.org/10.56471/slujst.v7i.475
Online ISSN: 2736-0903
Print ISSN: 2736-089X
www.slujst.edu.ng

The study of Ghomi et al. (2017) undertook a thorough examination of load balancing by categorizing the findings into seven distinct categories: Natural phenomena-based load balancing techniques, Agent-based load balancing techniques, General load balancing techniques, Application-oriented load balancing techniques, Network-aware task scheduling, and load balancing, Workflow specific scheduling algorithms. Various concerns and challenges in present load-balancing approaches as well as future directions, were highlighted, with a focus on Hadoop MapReduce and energy efficiency in cloud load balancing.

A systematic literature study of load-balancing approaches in cloud systems was undertaken using 726 paper search queries. An in-depth review of 15 original papers discovered evidence confirming load balancing as an ascending process that creates a new paradigm by improving cloud performance and influencing resource usage, by ensuring that the distribution of each input request is efficient and equitable. In order to create future strategies for load balancing that are more effective, some of the problems with current algorithms are addressed. However, that does not make the research free from other problems. To be a genuinely on-demand solution, load balancing techniques in computer environments still need to be improved in terms of reducing associated overhead, service response time and improving performance etc. It finished by reviewing the main goal of load balancing, current difficulties, unresolved problems, Methodologies, and Algorithms in cloud systems (Milani & Navimipour, 2016).

Shao and Chen (2016) concentrated on the creation of a great method named: The load balancing based on data correlation (LBSDC) for implementing cloud-based load balancing using data correlation. The data correlation idea seeks to find two important features; the selection of the migration unit and the selection of the migration server. The data correlation factor also aids in defining the link between communication overhead and the virtual server. The experimental framework compares the migration approach used by data correlation load balancing to that of a single virtual machine. According to the experimental findings, data correlation load balancing increases resource usage, decreases average response time, and lowers communication overhead.

The study presents an organized and thorough overview assessment on load balancing methods in cloud computing, by comparing several load balancing strategies, such as Round Robin (RR), Min-Min, Max-Min, Ant colony, Carton, Honey bee, etc. And equally considered performance metrics like Fairness, Throughput, Fault tolerance, Response time and resource utilization. The study cited the limitations of existing research on each cloud computing technique for failing to address relevant concerns like fairness, high throughput, and equality. Future work will focus on resolving the aforementioned issue and utilizing a hybrid approach to improve system performance and security (Aslam & Shah, 2015).

This study aims propose an improved approach for an effective task execution method that will improve cloud-based load balancing. The concept will evolve a new algorithm and logically merge current implementations and generate a new one by minimizing their functional behaviors. The study presents a group strategy for load balancing that enhances task performance, distributes the load well, and processes the load quickly (Patel et al., 2014).

2.1 Crow Search Load Balancing Algorithm (CSLBA)

Askarzadeh A. (2016) describe Crow as intelligent birds that exhibit intellect and have good knowledge of assessment which gave them the capable of memorize faces, use tools, and communicate in sophisticated ways when confronted with danger. Their cleverness makes it possible for them to hide and retrieve food across seasons.

Crow keeps a watch on other bird species, search for the locations where these birds conceal their food, and then take it when they get the chance. Whenever a crow steals, it takes a protection major by moving away to newer locations so as to avoid becoming a victim of theft in future. Been theft themselves, they used their experience to clearly read the intention of other thieves and protect themselves from being victimized. It is a heavy task finding source of food shield by a crow, because they move behind each other to get better food sources. However, in the course, if a crow discovered is been tailing by another, the crow attempts to prank that crow by heading off to another location of the environment.

Their manner of acting similar with an optimization process is demonstrated by their ability to hide excess food in a particular spot of the environment and later recalled the location of the hidden food when the need arises. Based on the Crow behaviour, a nature-based algorithm is developed.

Sule Lamido University Journal of Science & Technology Vol. 7 No. 2 [December, 2023], pp. 52-67
https://doi.org/10.56471/slujst.v7i.475
Online ISSN: 2736-0903
Print ISSN: 2736-089X
www.slujst.edu.ng

From optimization point of view: The Crows are **searchers,** the environment is the **search space**, each position of the environment is corresponding to a **Feasible solution**, the quality of food source is **Objective(fitness) function** of the problem.

Crow Search Algorithm and it shares these four principles:

- Crows live in groups.
- Crows are capable to recollect the position of their hiding food places.
- Crows pursue one another and take one another's food whenever they have the chance.
- Crows protect their location from intruders with a chance between [0, 1].

Assume that, there is a number crows in a **D**-dimensional environment. Let **N** be their flock size, and the location of the crow *i* at time (iteration) iter, in the search space represented by a vector indicated by $x^{i,iter}$ (i =1, 2,...N; $iter$ = 1,2,…….$iter_{max}$) where $x^{i,iter}$ [$x_1^{i,iter}$, $x_2^{i,iter}$,. . . . $x_d^{i,iter}$] and $iter_{max}$ represent the maximum number of iterations. Each crow recalls the location of its hiding food from its memory. At iteration *iter*, *Crow i* hiding place could be given by $m^{i,iter}$. For now, this is the best spot (location) *crow i* had so far found. In fact, each crow memorizes the spot it deems to be the best site. Crows forage throughout their surroundings in search of better food sources.

Let's assume that at iteration iter, *crow j* visits its hiding place $m^{j,iter}$. However, during the same iteration, *Crow i* pursues *Crow j* to find its hiding place. In this situation two outcomes are possible:

1. *Crow j* is unaware that he is being followed by *Crow i.* As a result, the *Crow i* will approach the hiding place of *Crow j*. In this case, the new position of the *Crow i* is revealed with the help of a random number with a uniform distribution between 0 and 1 and the flight length of the *Crow i* at iteration as it is shown in equation (i). In this case, the new position of *crow i* is obtained as follows:

$$x^{i,iter+1} = x^{i,iter} + ri \times fl^{iter} \times \left(m^{j,iter} - x^{i,iter}\right) \quad (1)$$

   Where ri represent random number with uniform distribution between 0 and 1, fl^{iter} represents flight distance of crow i at iteration iter.

   Figure 1 demonstrates this state's design and the impact of fl on the search capacity. Small fl values result in a local search (near $x^{i,iter}$ neighborhood) and large fl values result in global search (far from $x^{i,iter}$ neighborhood). Figure 1a demonstrates that if the value of fl is less than 1, the next location of *crow i* is on the dash line between $x^{i,iter}$ and $m^{j,iter}$
2. *Crow j* knows that he is being followed by Crow i. As a result, in order to protect its hidden food location from being stolen, *Crow j* will fool *Crow i* by going to another different position in the search space. Figure 1b demonstrates that if the value of fl is greater than 1, the next location of crow i is on the dash line which might surpass $m^{j,iter}$
   Now, both scenarios 1 and 2 might be expressed collectively as follows:

$$x^{i,iter+1} = \begin{cases} x^{i,iter} + ri \times fl^{i,iter} \times \left(m^{j,iter} - x^{i,iter}\right) & rj \geq AP^{j,iter} \\ \text{a random position} & \text{otherwise} \end{cases} \quad (2)$$

   Where rj represent random number with uniform distribution between 0 and 1, $fl^{i,iter}$ represents distance the crow flies and $AP^{j,iter}$ represent the Awareness Probability of crow j at iteration iter.

These two scenarios are demonstrated below:

Sule Lamido University Journal of Science & Technology Vol. 7 No. 2 [December, 2023], pp. 52-67
https://doi.org/10.56471/slujst.v7i.475
Online ISSN: 2736-0903
Print ISSN: 2736-089X
www.slujst.edu.ng

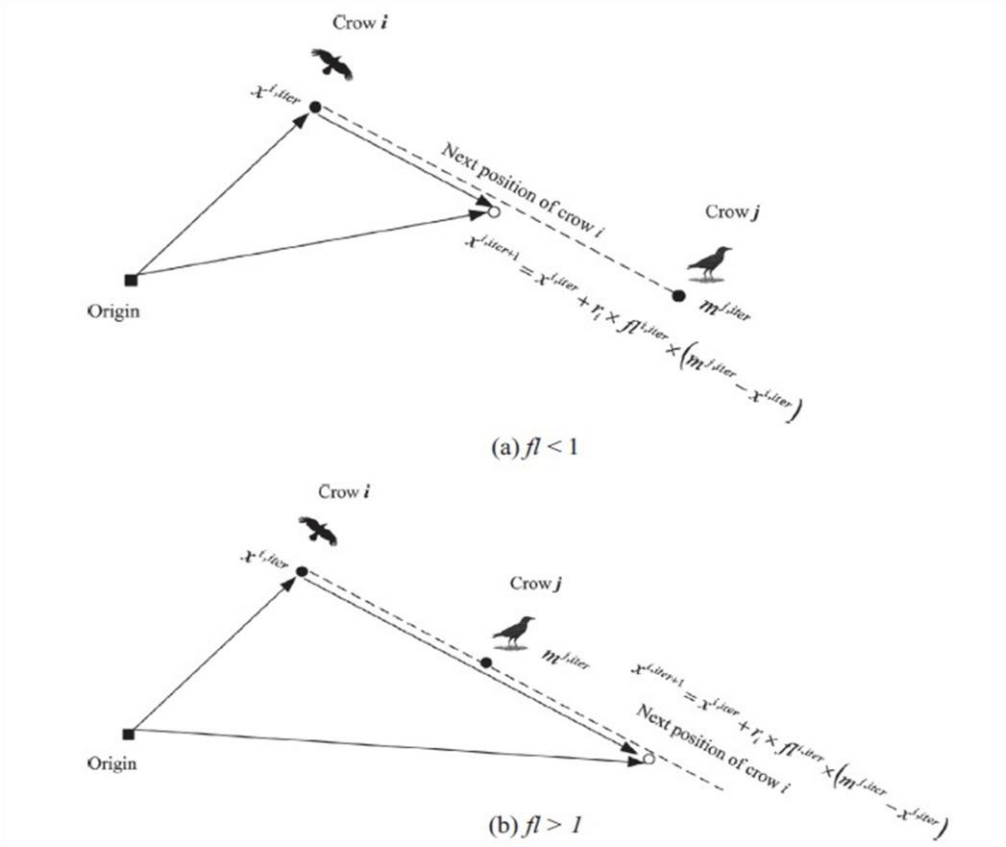

Figure 1 Flowchart of scenarios in CSA (a) fl < 1 and (b) fl > 1. (Askarzadeh, 2016)

2.1.1 Crow Search Load Balancing Algorithm

INPUT: n number of crows in the population
OUTPUT: Optimal crow position

Begin:
Step 1: INPUT: n number of crows in the population
Step 2: itr_{max} Maximum number of iterations
Step 3: OUTPUT: Optimal crow position
Step 4: Initialize position of crows
Step 5: Initialize crows' memory
Step 6: WHILE iter < itermax DO
Step 7: FOR $Crow_i$ in crows DO
Step 8: choose a random crow.
Step 9: determine a value of awareness probability AP
Step 10: Update $x^{i,\,iter}+1$ using Eq. (1)
Step 11: END FOR
Step 12: Check solution boundaries.
Step 13: Calculate the fitness of each crow
Step 14: Update crows' memory using Eq.(ii)
Step 15: END WHILE

Sule Lamido University Journal of Science & Technology Vol. 7 No. 2 [December, 2023], pp. 52-67
https://doi.org/10.56471/slujst.v7i.475
Online ISSN: 2736-0903
Print ISSN: 2736-089X
www.slujst.edu.ng

2.2 The Principle of Threshold Strategy

The need to manage the applications in cloud computing creates the challenge of on-demand resource provisioning and allocation in response to dynamic workloads change. Even though a number of strategies have been presented in effort to overcome resource allocation problems in cloud computing. This threshold-based strategy is viable for cloud computing, with an objective to dynamically assign virtual resources across cloud computing applications depending on load fluctuations. And it is also use to optimize the decision of resource reallocation (instead of a scheme that allocate resources needed to meet peak demands which often result in failing to satisfy peak load requests or underutilization of computing resources). Effort to address the problem of dynamic workload change, the proposed threshold-based strategy uses virtual machine as the minimum resource allocation unit. However, it monitors and anticipate the resource demands of cloud applications in order to alter virtual resources in accordance with those demands.

When a user starts an application, a virtual machine that satisfies the minimum resource requirement for the application is allocated. When workload of the application increases, an adjusted virtual machine is allocated for this application. This approach does not need to shutdown the existing virtual machine for resource reallocation, unlike some existing resource allocation schemes. Time delay is an issue in cloud computing. If the time delay between two resource re-allocation operations is either too long or short, the system may either or not be able to respond to load changes in a timely manner. Such a delayed response may waste virtual resources or hinder the performance of specific applications. On the other hand, if the time delay between two allocation events is too short, there will be excessive overhead for resource allocation. Therefore, reallocating virtual resources for cloud applications requires computational time and physical resources, the effectiveness of the resource allocation strategy depends on knowing when to reallocate the virtual resources.

In the proposed Threshold-based strategy, on time delay between two successive allocation events is configured to be adaptable to a cloud application's load change. If the load of an application varies gradually and steadily, a longer interval is used. When the load on an application varies quickly, a shorter interval is employed. However, an application's load may fluctuate from time to time over its lifespan. If the schedule task is based on current workload, the system may have to reallocate virtual resources often, resulting in significant overhead. To avoid the unnecessary overhead caused by the application's workload oscillation, a threshold is used to regulate the timing of resource reallocation.

2.3 Propose Improved Crow Search Load Balancing Pseudo code

Step 1: Set up the problem and its parameters. The problem, decision variables, and restrictions are all defined (flock size(N), the maximum number of iterations($iter_{max}$), flight length(fl), and Awareness Probability (AP)).

Step 2: Randomly initializing crows in D-dimensional search space with each Crow representing a viable solution to the problem.

Step 3: A fitness (objective) function is used to evaluate each Crow, and its value is put as an initial memory value. Each Crow stores its hiding place in its memory variable m^i.

Step 4: The Crow obtained its new position in the search space as shown in equation (i) and updates its position by selecting a random crow, i.e. generating a random value. if this value is greater than Awareness Probability `AP', then Crow *xi* will follow *xj* to know *mj*.

$$x^{i,iter+1} = \begin{cases} x^{i,iter} + ri \times fl^{i,iter} \times \left(m^{j,iter} - x^{i,iter}\right) & rj \geq AP^{j,iter} \\ \text{a random position} & \text{otherwise} \end{cases} \quad (3)$$

Where $AP^{j,\,iter}$ refers to Crow *j* awareness probability, iter refers to iteration number, r_i; r_j refers to random numbers, $fl^{i,iter}$ is the Crow i flight length to denote crow j memory.

Physical resources are assigned to virtual machines so that applications on these virtual machines are around their normal workloads. In essential, the system reserves some virtual resources to anticipate the sudden increase of the application's load (a similar approach is used in many of the Web applications).

Sule Lamido University Journal of Science & Technology Vol. 7 No. 2 [December, 2023], pp. 52-67
https://doi.org/10.56471/slujst.v7i.475
Online ISSN: 2736-0903
Print ISSN: 2736-089X
www.slujst.edu.ng

Step 5: Examine the viability of a new position. Each crow's new position is validated. If the crow's new position is viable, the crow's location is updated. Otherwise, the crow remains in its present place rather than moving to create a new position.

Step 6: Updating memory: The crow updates its memory with the new position as indicated in equation (4). If the new position is viable

$$m^{i,iter+1} = \begin{cases} x^{i,iter} + 1 \text{ if } fl(x^{i,iter+1}) \leq fl(m^{i,iter}) \\ m^{i,iter} \quad \text{otherwise} \end{cases} \quad (4)$$

Step 7: Assume the maximum workload of a virtual machine (virtual resource) is L_{max}, then equation (5) defines its normal workload Lnorm as:

$$L_{norm} = L_{max} \times \text{rate}_{norm} \quad (5)$$

Where **rate** $_{norm} \in [0,1]$ is called the normal workload rate.

Step 8: Equation (4) implement the threshold strategy as follows:

$$L_{threshold} = \text{rate}_{threshold} \times (1 - \text{rate}_{norm}) \times K \times L_{max} \quad (6)$$

Where **K** is the current number of virtual machines, **rate** $_{threshold}$ is a threshold rate ranging between 0 to 1. A resource reallocation takes place only if the application's load changes (up or down) is over $L_{threshold}$.

2.3.1 Proposed Improved Crow Search Load Balancing Algorithm (ICSLBA)

INPUT: n number of crows in the population
OUTPUT: Optimal crow position
Begin:
Step 1: INPUT: n number of crows in the population
Step 2: itr_{max} Maximum number of iterations
Step 3: OUTPUT: Optimal crow position
Step 4: Initialize position of crows
Step 5: Initialize crows' memory
Step 6: WHILE iter < $iter_{max}$ DO
Step 7: FOR Crow $_i$ in crows DO
Step 8: choose a random crow.
Step 9: determine a value of awareness probability AP
Step 10: Update $x^{i, iter}+1$ using Eq.(1)
Step 11: END FOR
Step 12: Check solution boundaries.
Step 13: Calculate the fitness of each crow
Step 14: normalVmLoad= Determine VmLoad according to Eq.(3)
Step 15: threshHoldVal = ComputeThresholdVal(normalVmLoad) according to Eq(4)
Step 16: IF threshHoldVal < normalVmLoad THEN
Step 17: schedule task to same vm
Step 18: ELSE
Step 19: check for next available vm for task scheduling
Step 20: Update crows' memory using Eq.(2)
Step 21: END WHILE
END

Sule Lamido University Journal of Science & Technology Vol. 7 No. 2 [December, 2023], pp. 52-67
https://doi.org/10.56471/slujst.v7i.475
Online ISSN: 2736-0903
Print ISSN: 2736-089X
www.slujst.edu.ng

3. Results and Discussion

The configuration analysis of the proposed Improved Crow Search Load Balancing Algorithm (ICSLBA), for experimentation is as follows: The User-based is composed of 6 different user-based distributed across 6 regions namely; Region 0 to region 5. Each of the user based has 60 requests per user. The Peak hours start time and Peak hours end time is 3 and 9 GMT respectively. The data size per requests (in bytes) is 100. The Average peak users and average off-peak users is 1000 and100 respectively. The application deployment configuration comprises 06 different data centers namely DC1 to DC6 each having 5 virtual machines (VMs) with an image size of 10000. The memory and bandwidth (BW) of each data center is 512 and 1000 respectively. Each of the data center has x86 architecture (Arch) and Linux as its operating system (OS). As shown in Table 1.

3.1 Parameter setting for Cloud Simulator

Table 1. Cloud Simulator parameter

Entity Type	Parameter	Value
User Base	User Base name	0 to 5
	Region	6
	Request per User	60
	Data size per request	100
	Peak Hours start (GMT)	3
	Peak Hours end (GMT)	**9**
	Average Peaks Users	1000
	Average Off-peak Users	100
Application Deployment Configuration	Data center	DC1 to DC6
	No. of VM	5
	Image size	10000
	Memory	512
	Bandwidth	1000

(Source: Singh et al., 2019)

In this experiment, the Crow search Load Balancing Algorithm (CSLBA) and the proposed Improved Crow search Load Balancing Algorithm (ICSLBA) are simulated in cloud environment using Cloud Analyst that runs on top of CloudSim via the configurations parameters as shown in Table 1. above. Three (3) service Broker policies namely: Closest Data Center Policy (CDCP), Optimize Response Time Policy (ORTP) and the Reconfigure Dynamically with Load Policy (RDLP) were used to determine the effect of the simulation parameters (Average Data Center processing Time (ADCPT), Average Makespan Time (AMT) and Average Waiting Time (AWT)).

3.2 Comparative Analysis of Average Make span Time (AMT)

One of the objects of this experiment is to test the proposed ICSLBA algorithm against the existing CSLBA algorithm and to compute value of Average Makespan Time as one of the performance variables. Result obtained are shown in Table 2 below. The algorithm with minimum AMT result under the three-service broker policy is said to be the optimal algorithm.

Table 2. Average Makespan Time (AMT) of CSLBA and ICSLBA for CDCP, ORTP and RDLP policies

Algorithm	CDCP	ORTP	RDLP
CSLBA	32.39	25.52	21.01
ICSLBA	12.01	9.10	9.60

(Source: Proposed result)

Table 2 represent the calculated values of AMT obtained for both algorithms after running the simulation. A corresponding graphical representation shown in Figure 1. depicts the variations of AMT with the changes in the service broker policies. The result obtained after running the simulation indicate that CDCP value of AMT for CSLBA is 32.39 and that of ICSLBA being 12.01 which is reduced by 62.9%. When implemented using ORTP: CSLBA has a value of

Sule Lamido University Journal of Science & Technology Vol. 7 No. 2 [December, 2023], pp. 52-67
https://doi.org/10.56471/slujst.v7i.475
Online ISSN: 2736-0903
Print ISSN: 2736-089X
www.slujst.edu.ng

25.52 of AMT compared to ICSLBA with 9.10 AMT value which shows a drastically decrease of about 64.34%. So also, RDLP has AMT value of 21.01 compare to ICSLBA value 9.60 which shows a reduction of 54.31%.

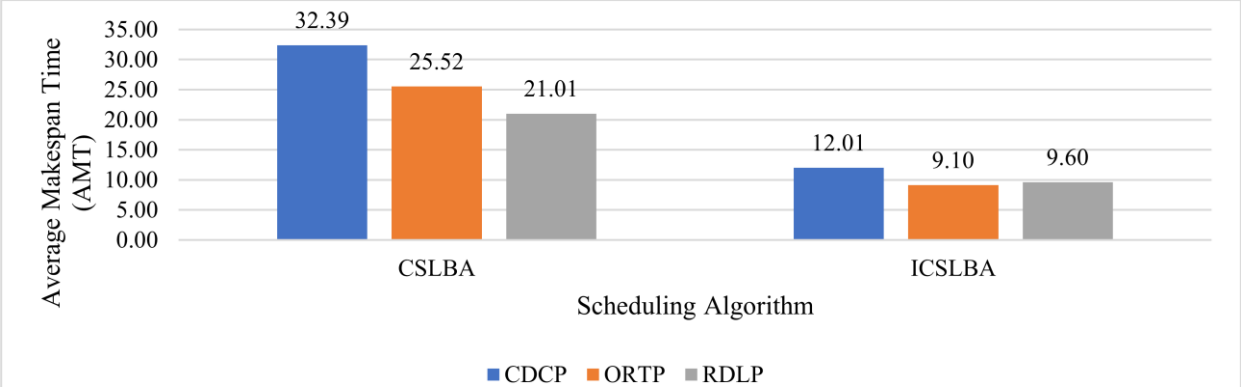

Figure 2. Graphic analysis of Average Makespan Time for CDCP, ORTP and RDLP
(Source: Proposed result)

3.3 Comparative Analysis of Average Waiting Time (AWT) of Data Center

The simulation is to test the proposed ICSLBA against CSLBA algorithm and compute the Average Waiting Time. The result obtained is shown in Table 3. The algorithm with minimum AWT result under the three-service broker policy is said to be the optimal scheduling algorithm.

Table 3 Average Waiting Time (AWT) of CSLBA and ICSLBA for CDCP, ORTP and RDLP policies

	CDCP		**ORTP**		**RDLP**	
	CSLBA	**ICSLBA**	**CSLBA**	**ICSLBA**	**CSLBA**	**ICSLB**
DC1	0.49	0.47	0.49	0.47	0.59	0.54
DC2	0.51	0.48	0.49	0.47	0.63	0.56
DC3	0.50	0.47	0.48	0.47	0.62	0.55
DC4	0.50	0.48	0.49	0.47	0.57	0.54
DC5	0.49	0.48	0.48	0.47	0.58	0.54
DC6	1.71	0.47	0.49	0.47	0.59	0.54

Figure 3. show the variation of AWT on each data center with the changes in the service broker policies is graphically represented below.

Sule Lamido University Journal of Science & Technology Vol. 7 No. 2 [December, 2023], pp. 52-67
https://doi.org/10.56471/slujst.v7i.475
Online ISSN: 2736-0903
Print ISSN: 2736-089X
www.slujst.edu.ng

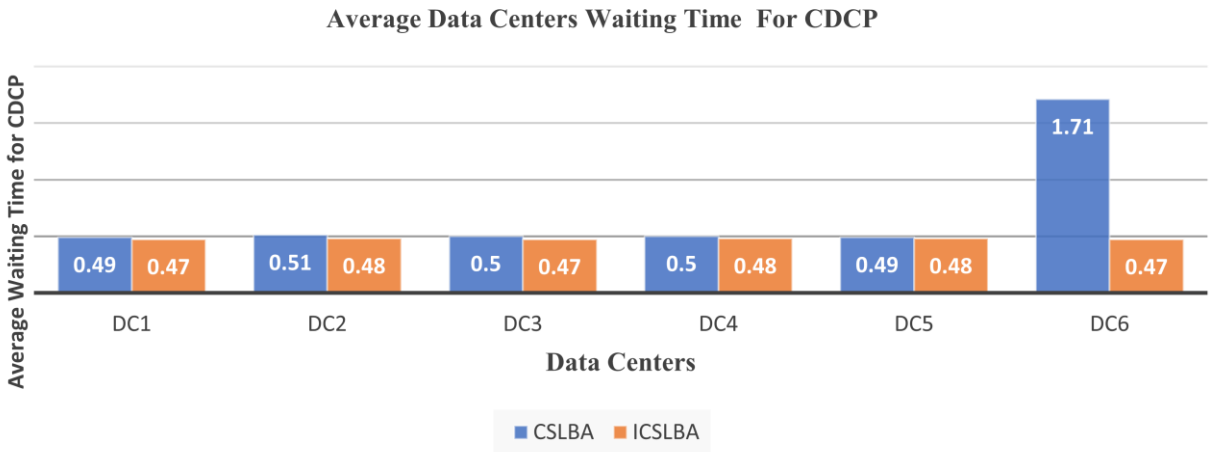

Figure 3. Graphical representation of Average Waiting Time of Data Centre's for CDCP
(Source: Proposed Results)

For values of AWT of each data center for CDCP is graphically represented in Figure 3. For CSLBA when the simulation result was obtained, the AWT values of data center DC1 to DC6 is 0.49, 0.51, 0.50, 0.50, 0.49, and 1.71.

The proposed ICSLBA has a corresponding value of data center DC1 to DC6 is 0.47, 0.49, 0.47, 0.48, 0.48, 0.47 which indicate a reduction of 4%, 3%, 6%, 4%, 2%, and 72.5% respectively.

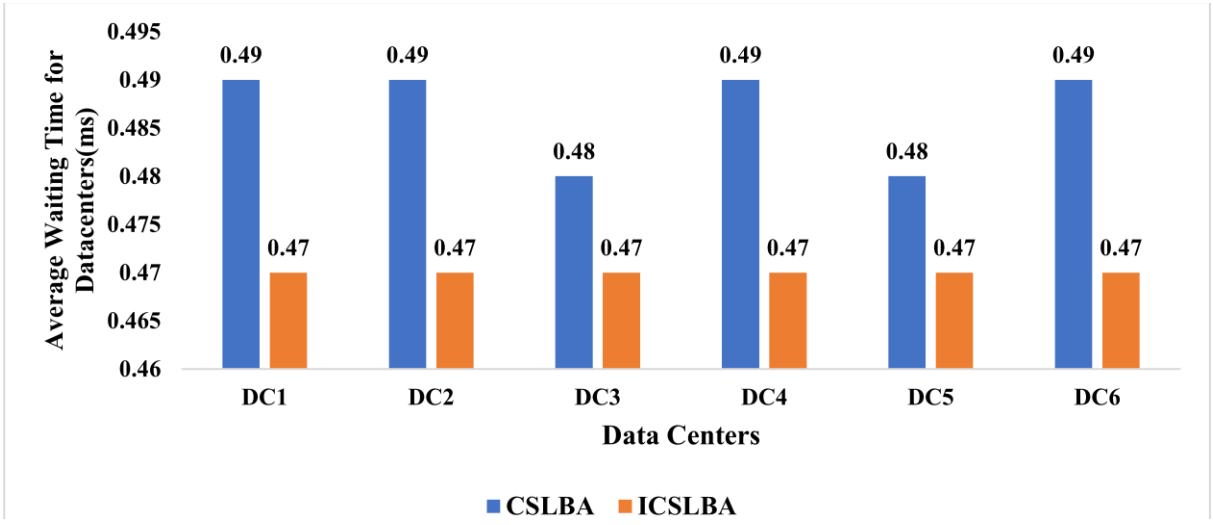

Figure 4. Graphic depict Average Waiting Time of Data Centre's for ORTP
(Source: Proposed Results).

When implemented using ORT, the graphical representation of AWT for data center is shown in Figure 4. CSLBA has AWT value of data center DC1 to DC6 is 0.49, 0.49, 0.48, 0.49, 0.48, 0.49 as compared to 0.47, 0.47, 0.47, 0.47, 0.47 0.47 the proposed ICSLBA shows a decrease of 4%, 4%, 2%, 4%, 2%, and 4% respectively.

Sule Lamido University Journal of Science & Technology Vol. 7 No. 2 [December, 2023], pp. 52-67
https://doi.org/10.56471/slujst.v7i.475
Online ISSN: 2736-0903
Print ISSN: 2736-089X
www.slujst.edu.ng

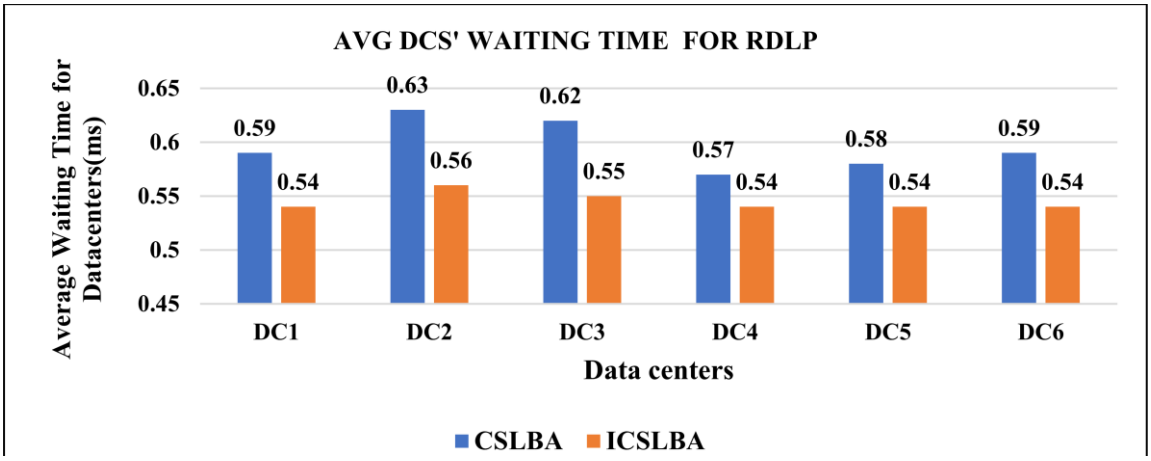

Figure 5. Graphic analysis of Data Centre's Average Waiting Time for RDLP
(Source: Proposed Results).

When implemented using RDLP, the graphical representation of AWT for data center is shown in Figure. 5. CSLBA has AWT value of data center DC1 to DC6 of 0.59, 0.63, 0.62, 0.57, 0.58, 0.59 as compared to 0.54, 0.56, 0.55, 0.54, 0.54, 0.54 the proposed ICSLBA shows a decrease of 8%, 11%, 11%, 7%, 7%, and 8% respectively.

3.4 Comparative Analysis of Average Data Center Processing Time (ADCPT)

The target of this simulation is to test the proposed ICSLBA against CSLBA algorithm and compute the value of Average Data Center Processing Time (ADCPT). The algorithm with minimum AMT result under the three-service broker policy is said to be the optimal algorithm.

Table 4. Average Data Center Processing Time (ADCPT) of CSLBA and ICSLBA for CDCP, ORTP and RDLP policies.

Algorithm	CDCP	ORTP	RDLP
CSLBA	28.79	0.49	0.60
ICSLBA	0.47	0.47	0.54

(Source: Proposed Results)

The Table 4 represents the test calculated values of ADCPT for both algorithms with a corresponding graphical representation shown in Figure 6. This depicts the variations of ADCPT with the changes in the service broker policies. The result obtained after running the simulation indicate that CDCP value of ADCPT for CSLBA is 28.79 and that of ICSLBA which is 0.47 which is drastically reduced by 98.4%. When implemented using ORTP for CSLBA has a value of 0.49 of ADCPT compared to ICSLBA with 0.47 which shows a decrease of about 4%. However, RDLP has ADCPT value of 0.60 compare to ICSLBA value 0.55 which shows a reduction of 8%.

Sule Lamido University Journal of Science & Technology Vol. 7 No. 2 [December, 2023], pp. 52-67
https://doi.org/10.56471/slujst.v7i.475
Online ISSN: 2736-0903
Print ISSN: 2736-089X
www.slujst.edu.ng

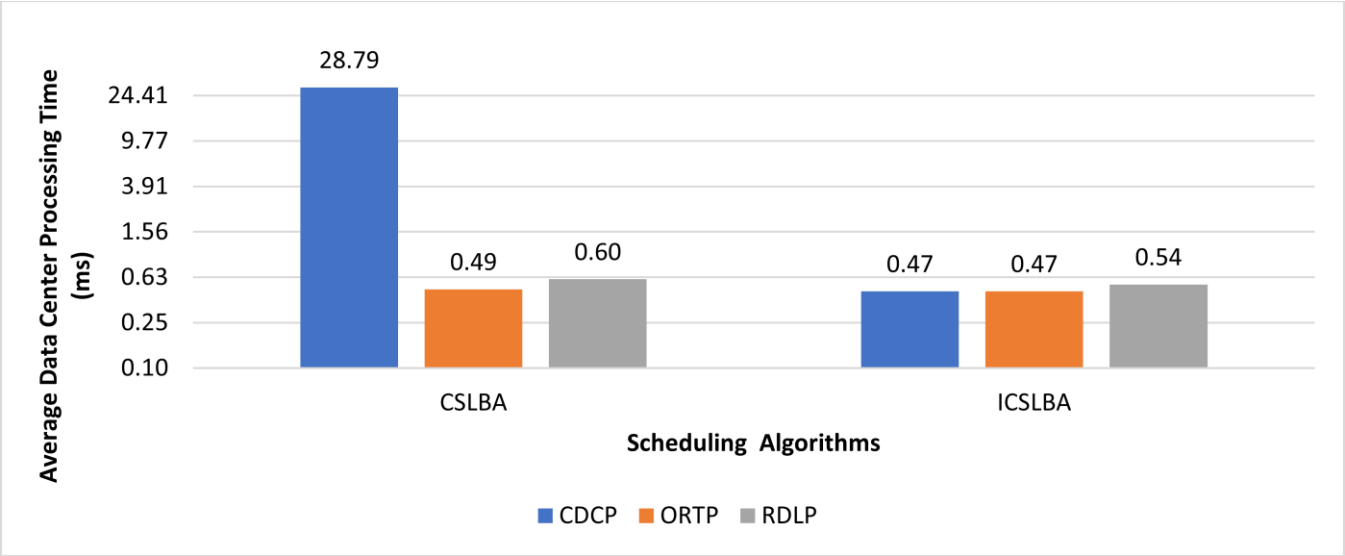

Figure 6. Graphic analysis of Average Data Center Processing Time for CDCP, ORTP and RDLP (Source: Proposed Results).

In a quest for finding an optimal load balancing algorithm between CSLBA, and the proposed ICSLBA, the parameters: ADCPT, AMT and AWT were used as indicators for performance evaluation. AMT under each of the service broker policy should be the minimum to find the optimal algorithm. During experimentation, it has been proved that the proposed ICSLBA has significantly decreased the AMT for each of the service broker policy. Thus, the comparison reveals that proposed ICSLBA is optimum.

The ADCPT in each of the service broker policy should, also, be the minimum to find the optimal algorithm. During experimentation, it has been proved that the proposed ICSLBA has reduced the ADCPT for each of the service broker policy to a greater degree. Thus, ICSLBA results as the optimum comparatively. The AWT for each of the data centers on each of the service broker policies should, as well, be the minimum to find the better algorithm. While experimenting, it has been ascertained that the ICSLBA strategy has reduced the AWT for data center on each of the service broker policy to certain limits. Hence, proved to be the optimum in comparison with the CSLBA. Thus, ICSLBA is optimum algorithm.

4. Conclusion

The study proposed a strategy on the basis of threshold value for load balancing systems and implemented with cloud computing architecture. This research has been issued to gain insight on different distributed load balancing algorithms within the cloud computing environment. The study was structured to provide an evaluation of load balancing algorithms performance. The proposed ICSBLA has been proved to be more optimal strategy than the existing CSBLA, this resulted from the comparative analysis carried out in terms of AMT, AWT and DCPT. The study concluded that, the algorithm called Improved Crow Search Load Balancing Algorithm performed best for all the scenario.

References

Abey, J. & Cyril, R. (2019). Performance Enhancement of Cloud Computing:Methodology & Tool. International Journal of Innovative Technology and Exploring Engineering (IJITEE), Volume-9 Issue-1,. DOI: 10.35940/ijitee.A3959.119119

Abhijit, A.R., & Apte S.S. (2016) A Comparative Performance Analysis of Load Balancing Algorithms in Distributed System using Qualitative Parameters, International Journal of Recent Technology and Engineering (IJRTE), Vol.1(3).

Online ISSN: 2736-0903
Print ISSN: 2736-089X
www.slujst.edu.ng

Alazzam, H., Mardini, W., Alsmady, A., & Enizat, A. (2019). Load Balancing in Cloud Computing Using Water Flow-Like Algorithm. International Conference on Data Science, E-learning and Information Systems. doi.org/10.1145/3368691.3368720

Alireza, S.M. & Navimipour, N.J. (2016). Load balancing mechanisms and techniques in the cloud environments: Systematic literature review and future trends. Journal of Network and Computer Applications 71, 86–98. //dx.doi.org/10.1016/j.jnca.2016.06.003

Aslam, S. & Shah, A.M. (2015). Load Balancing Algorithms in Cloud Computing: A Survey of Modern Techniques. National Software Engineering Conference. 978-1-4673-8163-5/15/$31.00

Atyaf, D. & Khaldun, I.A. (2017). An Efficient Load Balancing Scheme for Cloud Computing**.** Indian Journal of Science and Technology, Vol 10(11).

Ghomi, E.J., Rahmani, A.M., & Qader, N.N. (2017). Load-balancing algorithms in cloud computing: A survey. Journal of Network and Computer Applications 88, 50–71. DOI: https://doi.org/10.1016/j.jnca.2017.04.007.

Hieu, N.L., & Tran, H.C. (2022). ITA: The improved throttled algorithm of load balancing on Cloud Computing. International Journal of Computer Networks & Communications (IJCNC) Vol.14, No.1, DOI: 10.5121/ijcnc.2022.14102

Jena, U.K., Das, P.K., & Kabat, M.R. (2022). Hybridization of meta-heuristic algorithm for load balancing in cloud computing environment. Journal of King Saud University – Computer and Information Sciences. https://doi.org/10.1016/j.jksuci.2

Krishnadoss, P., Natesan, G. & Ali J. (2021). CCSA: Hybrid Cuckoo Crow Search Algorithm for Task Scheduling in Cloud Computing. International Journal of Intelligent Engineering and Systems, Vol.14, No.4, DOI: 10.22266/ijies2021.0831.22

Kumar, D. (2023). A smart, real-time, secure strategy for allocating and scheduling resources based on fog computing and the multi-objective crow search algorithm. United College of Engineering and Research ://doi.org/10.21203/rs.3.rs-2730492/v1.

Kumar, M. & Bhushan, B. (2020). A Methodological Comparison of the most efficient Load Balancing algorithms in Cloud Computing. International Conference on Innovative Computing and Communication. International Conference on Innovative Computing and Communication. DOI: 10.2139/ssrn.3598908

Kumar, P. & Kumar, R. (2019). Issues and Challenges of Load Balancing Techniques in Cloud Computing: A Survey. ACM Computing Surveys. https://doi.org/10.1145/3281010. 35 pages

M. Roshni Thanka, P. Uma Maheswari & E. Bijolin Edwin (2019). A hybrid algorithm for efficient task scheduling in cloud computing environment. International Journal of Reasoning-based Intelligent Systems Vol. 11, No. 2, DOI: 10.1504/IJRIS.2019.10021325

Milani, A.N. (2016) Load balancing mechanisms and techniques in the cloud environments: systematic literature review and future trends. http://dx.doi.org/10.1016/j.jnca.2016.06.003

Mishra, P.J., Panda, R.S. (2020). Load Balancing Using Cloudsim Simulator in Cloud Computing. International journal of Scientific & Technology research volume 9, issue 01. www.ijstr.org/ ISSN 2277-8616

Mishra, S.K., Sahoo, B., & Parida, P.P. (2020). Load balancing in cloud computing: A big picture. Journal of King Saud University – Computer and Information Sciences. https://doi.org/10.1016/j.jksuci.2018.01.003

Parida, S. & Panchal, B. (2018). Review Paper on Throttled Load Balancing Algorithm in cloud computing environment. International Journal of Scientific Research in Science, Engineering and Technology, Volume 4. ISSN: 2394-4099

Patel, J., Thaker C.S. & Chaudhari, H. (2014). Task Execution Efficiency Enrichment in Cloud Based Load Balancing Approaches. dx.doi.org/10.1145/2677855.2677919.

Rizwan, K., Tahir, A., & Imran, R. (2016). Performance Degradation Factors in Cloud. International Journal of Scientific & Engineering Research, Volume 7, Issue 11. http://www.ijser.org/ ISSN 2229-5518

Sambangi, S. & Gondi, L. (2019). Task Scheduling Allocation based on Task Completion Time in Cloud Environment. International Journal of Recent Technology and Engineering (IJRTE). ISSN: 2277-3878, Volume-8 Issue-4,. DOI:10.35940/ijrte.D7409.118419

Saravanan, G.S., Neelakandan, P.E., & Maurya S. (2023) Improved wild horse optimization with levy flight algorithm for effective task scheduling in cloud computing. Journal of Cloud Computing: Advances, Systems and Applications. //doi.org/10.1186/s13677-023-00401-1

Online ISSN: 2736-0903
Print ISSN: 2736-089X
www.slujst.edu.ng

Singh H., Tyagi, S. & Kumar, P. (2019) Crow Search based Scheduling Algorithm for Load Balancing in Cloud Environment, International Journal of innovative Technology and Exploring Engineerimg,.8(9) pp 5. DOI:10.35940/ijitee.I7787.078919

Sova, P. (2016). Cloud Computing in brief. IOSR Journal of Computer Engineering (IOSR-JCE), 101-103.

Swati, I. B., & Ankur, O. B. (2015). Cloud Computing: History, Architecture, Security Issues. International Journal of Advent Research in Computer and Electronics (IJARCE), 102-108.

Thanka, M.R., Maheswari, P.U. & Edwin, E.B. (2019). A hybrid algorithm for efficient task scheduling in cloud computing Environment. Int. J. Reasoning-based Intelligent Systems, Vol. 11, No. 2. pp.134–140. DOI: 10.1504/IJRIS.2019.10021325020.01.012