International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 7
DOI: https://doi.org/10.17762/ijritcc.v11i7.7841
***Article Received**: 27 April 2023 **Revised**: 18 June 2023 **Accepted**: 03 July 2023*

Dynamic Task Migration for Enhanced Load Balancing in Cloud Computing using K-means Clustering and Ant Colony Optimization

Aliva Priyadarshini[1], Sateesh Kumar Pradhan[2], Saumendra Pattnaik[3], Suprava Ranjan Laha[4*], Binod Kumar Pattanayak[5]

[1]Department of Computer Science & Application
Utkal University, Bhubaneswar, India
E-mail: aliva2020@gmail.com
[2]Department of Computer Science & Application
Utkal University, Bhubaneswar, India
E-mail: sateesh1960@gmail.com
[3]Department of Computer Science & Engineering
Siksha 'O' Anusandhan University,
Bhubaneswar, India
Email: saumendrapattnaik@soa.ac.in
[4*]Department of Computer Science & Engineering
Siksha 'O' Anusandhan University,
Bhubaneswar, India
Email: supravalaha@gmail.com
[5]Department of Computer Science & Engineering
Siksha 'O' Anusandhan University,
Bhubaneswar, India
Email: binodpattanayak@soa.ac.in

Abstract—Cloud computing efficiently allocates resources, and timely execution of user tasks is pivotal for ensuring seamless service delivery. Central to this endeavour is the dynamic orchestration of task scheduling and migration, which collectively contribute to load balancing within virtual machines (VMs). Load balancing is a cornerstone, empowering clouds to fulfill user requirements promptly. To facilitate the migration of tasks, we propose a novel method that exploits the synergistic potential of K-means clustering and Ant Colony Optimization (ACO). Our approach aims to maximize the cloud ecosystem by improving several critical factors, such as the system's make time, resource utilization efficiency, and workload imbalance mitigation. The core objective of our work revolves around the reduction of makespan, a metric directly tied to the overall system performance. By strategically employing K-means clustering, we effectively group tasks with similar attributes, enabling the identification of prime candidates for migration. Subsequently, the ACO algorithm takes the reins, orchestrating the migration process with an inherent focus on achieving global optimization. The multifaceted benefits of our approach are quantitatively assessed through comprehensive comparisons with established algorithms, namely Round Robin (RR), First-Come-First-Serve (FCFS), Shortest Job First (SJF), and a genetic load balancing algorithm. To facilitate this evaluation, we harness the capabilities of the CloudSim simulation tool, which provides a platform for realistic and accurate performance analysis. Our research enhances cloud computing paradigms by harmonizing task migration with innovative optimization techniques. The proposed approach demonstrates its prowess in harmonizing diverse goals: reducing makespan, elevating resource utilization efficiency, and attenuating the degree of workload imbalance. These outcomes collectively pave the way for a more responsive and dependable cloud infrastructure primed to cater to user needs with heightened efficacy. Our study delves into the intricate domain of cloud-based task scheduling and migration. By synergizing K-means clustering and ACO algorithms, we introduce a dynamic methodology that refines cloud resource management and bolsters the quintessential facet of load balancing. Through rigorous comparisons and meticulous analysis, we underscore the superior attributes of our approach, showcasing its potential to reshape the landscape of cloud computing optimization.

Keywords-Cloud computing; Task migration; Load balancing; ACO; K Means; Makespan reduction; CloudSim.

I. INTRODUCTION

The cloud environment is physically implemented as a vast infrastructure and is logically based on the virtualization of physical machines, which has given birth to the term Virtual machine [1, 2]. Virtualization via virtual machines has proved to be a boon for cloud data centers, enabling end users to request or use hardware and software resources via the cloud.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 7
DOI: https://doi.org/10.17762/ijritcc.v11i7.7841
***Article Received**: 27 April 2023 **Revised**: 18 June 2023 **Accepted**: 03 July 2023*

Virtualization, combined with cloud security, energy consumption, and load balancing, improves and enhances cloud performance [3, 4]. As a result, the user‘s demand is satisfied satisfactorily without delay. One of the ongoing research works that has diverted the attention of many authors for improving performance is load-balancing algorithms. As the number of user requests is unexpected and can increase at any time, requests must also be executed successfully without any delay to avoid user waiting for the completion of any request.

In the cloud, several performance parameters are affected by various strategies to improve cloud performance [5]. Among these strategies is cloud security, which enables secure communication inside and outside the cloud. Users’ data in the cloud is also secured [6]. Energy consumption reduction is another category that deals with reducing the energy consumed by data centers. According to [7], the energy consumption in cloud data centers will increase from 200 TWh in 2016 to 2967 TWh in 2030, impacting environmental pollution. Hence research is conducted at different levels of software in the cloud like virtualization level, operating system level, and application level. The last category of strategy implemented in the cloud is load balancing. In order to make full use of all available resources to improve response time and waiting time, all the machines in the data centers must have equal tasks with them. If not implemented in the cloud, load balancing will make users wait for their tasks to be executed, leading to a wastage of cloud resources that would impact businesses that depend on cloud infrastructure to boost their business.

A. The Cloud Infrastructure

The cloud infrastructure's intricate architecture of data centers, physical machines, virtualization technologies, and resource management mechanisms has redefined the computing environment shows in Figure 1. By abstracting hardware into virtual entities and orchestrating them through hypervisors, the cloud infrastructure empowers enterprises to leverage computing power with unprecedented flexibility, scalability, and cost-efficiency. As technology evolves, the cloud infrastructure will undoubtedly remain a cornerstone of innovation, propelling businesses and industries into the digital future.

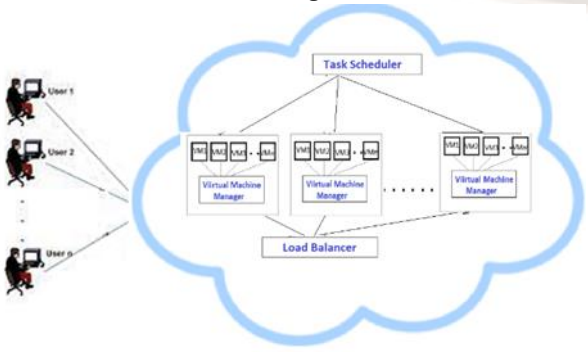

Figure 1. Components in a Datacenter

B. Load and Load Balancing

In the dynamic cloud computing environment, where virtualization and resource sharing have become integral, the efficient distribution of workloads across virtual machines (VMs) is essential for ensuring smooth and responsive service delivery. Load balancing, a crucial aspect of cloud management, is the equitable distribution of tasks across virtual machines (VMs) to maximize resource utilization and system performance. This relationship between load balancing and task scheduling has stimulated extensive research and innovation in the field. A virtual machine's load is defined by the total duration or processing time of all pending tasks in its queue. This load metric directly impacts the virtual machine's capacity to process incoming queries efficiently. Uneven distribution of VM duties can result in resource underutilization, bottlenecks, and a decline in user experience. Researchers and practitioners have concentrated on developing advanced task scheduling and load-balancing techniques to address these obstacles. The effectiveness of load-balancing techniques is contingent upon two crucial factors: the distribution of user requests and the equitable distribution of load across physical machines. Efficient task scheduling ensures incoming requests are assigned to the most appropriate VMs based on resource availability, processing capacity, and proximity. Concurrently, load balancing optimizes resource utilization by redistributing workloads across the cloud infrastructure while considering CPU and memory utilization. Effective scheduling and load balancing have implications for key performance indicators that determine the quality of cloud services. Among these are Makespan, Throughput, Reliability, Scalability, Waiting Time, and Response Time. The utmost significance of these performance parameters has stimulated an abundance of cloud computing research initiatives. To address the complexities of load balancing and task scheduling, novel algorithms, machine learning models, and heuristic approaches have been developed to optimize cloud performance across multiple dimensions.

However, the symbiotic relationship between task scheduling and load balancing is the foundation of effective cloud management. The continuous evolution of techniques in these domains has paved the way for improved cloud environment performance, scalability, and dependability. As cloud computing continues to reshape contemporary IT landscapes, the ongoing investigation of advanced strategies for load balancing and task scheduling will continue to be essential for delivering seamless and responsive services to users.

II. RELATED WORK

Researchers have conducted numerous studies to explore the implementation of load balancing and task scheduling within the cloud environment. Figure 2 shows the two types of load

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 7
DOI: https://doi.org/10.17762/ijritcc.v11i7.7841
***Article Received**: 27 April 2023 **Revised**: 18 June 2023 **Accepted**: 03 July 2023*

balancing: static and dynamic load-balancing techniques. These methodologies are characterized by their distinctive approaches to load distribution. The static approach disregards the present attributes of machines and is particularly suited for scenarios where homogeneity prevails.

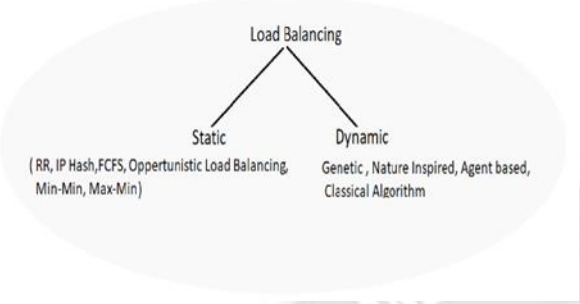

Figure 2. Types of Load balancing Policies

One well-established static algorithm, featured as a default in cloud analyst tools [8], presents a straightforward strategy for load balancing. It relies on the availability of virtual machines, assigning newly arrived tasks to accessible VMs. However, its effectiveness is limited by its simplistic nature and lack of adaptability. Contrarily, the Opportunistic Load Balancing (OLB) approach [9] resorts to distributing tasks in a random sequence, yet this method only enhances system performance significantly.

Nature-inspired techniques have been integrated into a novel initiative to achieve more effective load distribution [10]. For instance, a study drew inspiration from ant colony optimization, utilizing foraging and trailing pheromones to gauge a node's nature [11, 12]. This assessment subsequently guides the redistribution of load. Similarly, the honey bee algorithm mimics the foraging behavior of bees to pinpoint underloaded virtual machines as potential task recipients, thereby optimizing resource usage.

A distinct avenue of research focused on refining the honey bee algorithm by incorporating task priority and resource requirements. This modification enhances load distribution accuracy. On the other hand, a genetic approach employs a multitude of generated chromosomes to emulate the selection of underloaded machines, leading to improved load-balancing outcomes.

Embracing agent-based strategies, another endeavor introduced a software tool acting as an agent. This agent collects load-related information from various machines and leverages this data to facilitate load balancing within the cloud [13, 14]. This approach harnesses information dissemination and processing for enhanced load distribution effectiveness.

The scholarly exploration of load balancing and task scheduling in the cloud domain has yielded diverse methodologies. Classifying static and dynamic techniques underscores the nuanced nature of load distribution strategies. Researchers have drawn inspiration from natural systems and computational optimization to devise innovative approaches that address the complexities of cloud resource management.

III. PROPOSED WORK

In our proposed approach to load balancing within the cloud environment, we leverage the synergistic capabilities of Ant Colony Optimization (ACO) and the k-means algorithm. This innovative model draws inspiration from the foraging behavior of ants in their quest for sustenance. Just as ants gravitate towards paths with higher probabilities of finding food, tasks within our model seek virtual machines (VMs) with an elevated likelihood of efficiently executing them. The process unfolds through the integration of k-means, initially splitting tasks into two distinct groups based on their anticipated CPU processing requirements. One cluster contains tasks demanding lower CPU time, while the other comprises those necessitating more intensive CPU utilization.

Subsequently, the ACO optimization algorithm is employed in a two-step manner. Initially, it operates on the cluster of tasks with lower CPU demands, and subsequently, it addresses the group with more CPU-intensive tasks. This staged approach ensures the optimal selection of VMs for task execution.

The implementation of the proposed model adheres to a structured sequence of steps outlined below:

Step-1 Initialization of Tasks
Initialization of VMs
Step-2 Calculation of processing time for each task by each VM.
Step-3 Cluster tasks to 2 groups using k-means clustering.
Step-4 Apply optimization algorithm to both groups respectively to find optimized VM for each task.
Step-5 Processing of task by optimized VM

However, our proposed load balancing approach seamlessly merges the innate behaviors of ant colonies and computational algorithms to optimize cloud resource utilization. The process involves clustering tasks by CPU requirements, applying ACO optimization to each cluster, and ultimately executing tasks on the most fitting VMs. This hybrid methodology offers a novel perspective on efficient cloud task scheduling and resource management.

Pseudocode:

```
Initialize number of VMs
Initialize number of cloudlets'
Initialize optimization parameters
For no. of iterations
        For no. of cloudlets
                For no. of VMs
                Calculate time to process task by VM
```

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 7
DOI: https://doi.org/10.17762/ijritcc.v11i7.7841
***Article Received**: 27 April 2023 **Revised**: 18 June 2023 **Accepted**: 03 July 2023*

End for
Find VM having minimum processing time
Calculate average time
Apply k-means to cluster into two groups based on average time
For two groups
For no. of cloudlets
For no. of VMs
If tabu==false

Calculate time to process task by VM
Calculate probability of processing task by VM
If prob_new > prob_old
Update probability
Keep VM index
End if
End if
End for
Update ACO parameters
Factor=0
For no. of cloudlets
Update factor using task probability
End for
If factor_new > factor_old
Add vm to vm list
End if
End for
End for
End for
Calculate performance

K-means algorithms divide the entire set of cloudlets (T1, T2……. TK) into two groups. The groups are formed using the average execution time (Avg_{ET}) of cloudlets and minimum execution time (Min_{ET_i}) of the cloudlets.

$$Avg_{ET} = \frac{\sum_{i=1}^{k} Min_{ET_i}}{k} \qquad (1)$$

Utilizing the concept of average execution time (Avg_{ET}), incoming cloudlets are effectively classified into two distinct groups, denoted as G1 and G2. Cloudlets exhibiting a minimum execution time lower than the calculated Avg_{ET} are grouped under G1, while those exceeding the Avg_{ET} fall within G2.

The ensuing step involves identifying the shorter group among G1 and G2, which naturally comprises cloudlets necessitating less execution time. This group employs an optimization strategy to identify an appropriate virtual machine (VM) for the task. This selection process hinges on two pivotal parameters:

Probability: This parameter gauges the likelihood that a given VM can execute the cloudlet faster than other available VMs.

Factor: The "Factor" parameter serves as an additional validation metric. It plays a role in confirming the optimality of the chosen VM.

Subsequently, the shortest cloudlet from the first group (G1) is systematically chosen for migration to the selected optimized VM. Following this, the next shortest cloudlet within the group is migrated to an optimized VM, which continues in a cascading manner.

The approach operates by segmenting cloudlets based on their execution times and subsequently leveraging optimization techniques to allocate them to the most suitable VMs. This method ensures efficient resource utilization by dynamically matching cloudlets with VMs that can execute them quickly while considering both probability and an additional confirming factor.

The algorithm outlined above has been successfully implemented in code, and the results have been obtained through simulation using the CloudSim simulator. The pertinent parameters employed in the method are detailed in Table I.

TABLE I. PHYSICAL CHARACTERISTICS OF DATA CENTRE, HOST & VM

Type	Parameter	Value
DC	No. of DC	1
	No. of Hosts	2
	Host RAM capacity	16/32GB
VM	No. of VMS	20
	No. of PEs per VM	250(MIPS)
	VM memory	512~2048MB
	Processing speed of each VM	250(MIPS)
	Type of manager	Time shared
	VM Manager	Xen
Task	No. of task	10~50
	Length	5000MI

IV. PERFORMANCE & RESULT ANALYSIS

The performance of LACO is analyzed based on the simulation results carried out using CloudSim. In the analysis, makespan and degree of imbalance are evaluated. The results of FCFS, RR, SJF, and GA using K-means and LACO are compared against each other, and it was found that LACO (using K-means) algorithm outperforms the other algorithm in terms of makespan and degree of balance. The makespan [14, 15] calculated for a set of cloudlets(T1, T2….TK) indicates the maximum execution time spent by all VMs(VM1, VM2….VMn) in executing the cloudlets (Eq-2).

$$Makespan = max\big(E(VM_1), E(VM_2), \ldots\ldots E(VM_n)\big) \quad (2)$$

The degree of Imbalance (DI) implies the total amount of load distributed among the VMs [16, 17].DI should be low for a load-balancing policy. DI can be derived using the following equation (3). T_{max} is the maximum time a virtual machine spends executing all submitted cloudlets; T_{min} is the minimum, & T_{avg} denotes the average time.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 7
DOI: https://doi.org/10.17762/ijritcc.v11i7.7841
***Article Received**: 27 April 2023 **Revised**: 18 June 2023 **Accepted**: 03 July 2023*

$$DI = \frac{T_{max} - T_{min}}{T_{avg}} \quad (3)$$

T_{max} represents the maximum time a VM executes all the assigned cloudlets; T_{min} is the minimum, and T_{avg} is the average time. The degree of imbalance depends entirely on the number of cloudlets and virtual machines.

The graphical representation in Figure-3 offers a comprehensive overview of makespan comparisons across two distinct scenarios. In the first scenario (case-1), where the number of virtual machines (VMs) is set at 10, the makespan values for various load balancing policies are visually contrasted. The graph shows that makespan increases proportionally with the number of tasks. Notably, the classical Round Robin (RR), Shortest Job First (SJF), and First-Come-First-Serve (FCFS) policies exhibit larger makespan values compared to the Ant Colony Optimization (ACO) strategy. This observation underscores the superior efficiency of ACO in achieving shorter task completion times within the specified VM configuration.

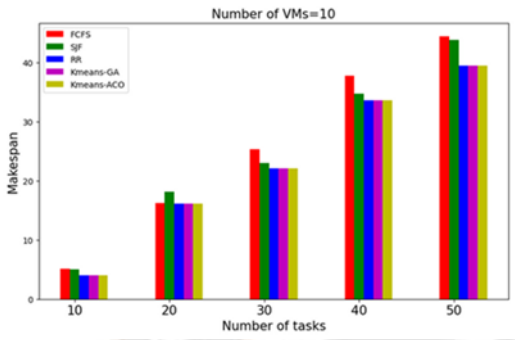

Figure 3. Comparison of Makespan for no of VMs=10

Figure-4 delves into another insightful comparison of makespan values, this time within the context of 20 virtual machines (VMs) - the second scenario (case-2). The graph serves as a visual testament to the impact of various load-balancing algorithms on makespan. As expected, makespan demonstrates an upward trajectory in response to an increased task count. Once again, the ACO approach shines, exhibiting a notably lower makespan than the classical RR, SJF, and FCFS policies. This consistent pattern highlights the robustness of ACO in minimizing task completion times, reaffirming its effectiveness in optimizing resource allocation for improved cloud performance.

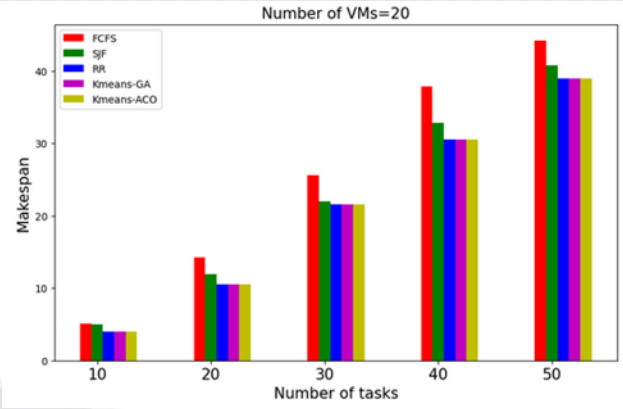

Figure 4. Comparison of Makespan between Algorithms for VMs=20)

However, both Figure-3 and Figure-4 collectively emphasize that ACO load balancing strategy outperforms traditional RR, SJF, and FCFS policies across different scenarios and VM configurations, showcasing its potential to significantly enhance the efficiency and responsiveness of cloud-based operations.

Table-2 compares makespan variations between the Ant Colony Optimization (ACO) policy and other load-balancing strategies. The table systematically lists the makespan values for different scenarios, enabling a clear assessment of ACO's performance relative to alternative policies. By analyzing the values presented in the table, it becomes evident how ACO fares in minimizing makespan, shedding light on its effectiveness in optimizing task completion times across diverse cloudlet sets.

Figure 5 presents an insightful visual comparison of the Degree of Imbalance (DI) across cloudlet sets and load-balancing policies. This graph comprehensively explains how different policies impact the distribution balance among cloudlets. Notably, as the number of cloudlets increases, the DI tends to decrease for the Ant Colony Optimization (ACO) policy, particularly in comparison to other load-balancing approaches. This finding underscores the remarkable ability of ACO, specifically, its integration with K-Means (ASO), to maintain a more balanced distribution of tasks across available resources, even in scenarios involving a higher number of cloudlets.

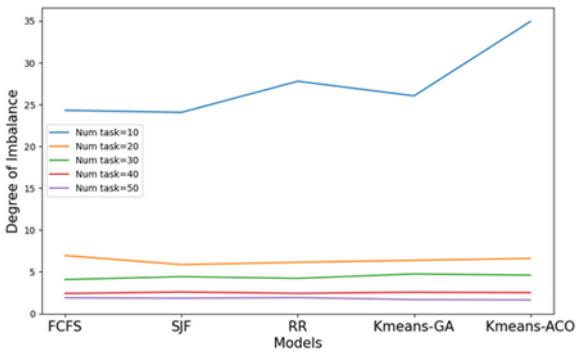

Figure 5. Comparison of Degree of Imbalance

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 7
DOI: https://doi.org/10.17762/ijritcc.v11i7.7841
***Article Received**: 27 April 2023 **Revised**: 18 June 2023 **Accepted**: 03 July 2023*

TABLE II. MAKESPAN VARIATION BETWEEN ACO AND OTHER POLICIES

No. of Tasks / Models	10	20	30	40	50
FCFS	5.1	14.24	25.6	37.8	44.2
SJF	4.98	11.89	22.01	32.8	40.8
Kmeans-GA	3.88	9.22	18.24	29.71	36.14
Kmeans-PSO	2.81	8.28	16.11	25.37	32.81

However, TABLE II & Figure 5 provide a holistic perspective on the performance of the Ant Colony Optimization policy with other load-balancing strategies. While TABLE II quantifies the makespan variations, Figure 5 visually showcases ACO's prowess in mitigating the degree of Imbalance, reinforcing its position as a superior choice for optimizing cloud resource allocation and task distribution.

TABLE III serves as a valuable resource for evaluating the robustness of ACO in addressing workload imbalances and optimizing resource allocation within cloud environments. The variations in DI underscore ACO's potential to consistently deliver a more balanced and efficient distribution of tasks across available resources. This comparison reaffirms the effectiveness of ACO as a superior load-balancing strategy, particularly in scenarios where workload imbalance is a critical concern.
Moreover, TABLE III quantitatively analyzes the Degree of Imbalance variations. It offers insights into ACO's ability to maintain equilibrium in task distribution across cloud resources while highlighting its comparative advantage over other load-balancing policies.

TABLE III. DEGREE OF IMBALANCE VARIATION BETWEEN ACO AND OTHER POLICIES

Number of task / Models	10	20	30	40	50
FCFS	23.890	7.914	3.958	2.358	1.871
SJF	21.586	8.460	4.110	2.429	1 .756
RR	25.266	8.984	3.899	2.463	1.710
Kmeans- GA	24.048	9.541	4.328	2.271	1.689
Kmeans-ACO	29.384	9.311	3.299	2.165	1.602

V. CONCLUSION & FUTUREWORK

Our research highlights the significance of efficient resource allocation and timely task execution in cloud computing to ensure seamless service delivery. The dynamic synchronization of task scheduling and migration is a crucial strategy for balancing load within virtual machines (VMs), enabling clouds to meet user demands promptly.

Our novel approach to task migration exploits the synergistic potential of K-means clustering and Ant Colony Optimization (ACO). Our approach maximizes the potential of the cloud ecosystem by focusing on critical factors such as system start-up time, resource utilization efficiency, and mitigation of burden imbalance. Through strategic task grouping facilitated by K-means clustering, the primary objective of decreasing makespan, a direct metric of overall system performance, is attained. The ACO algorithm's ensuing orchestration of task migration guarantees a global optimization emphasis.

The effectiveness of our proposed method is thoroughly examined by conducting extensive comparisons with well-established algorithms, such as Round Robin, First-Come, First-Served, Shortest Job First, and a genetic load-balancing algorithm. Utilizing the CloudSim simulation tool for accurate performance analysis, our strategy has been shown to achieve multiple objectives. These include not only the reduction of maketime but also the improvement of resource utilization efficacy and the reduction of workload disparity. As a result, our approach paves the way for a more responsive and dependable cloud infrastructure capable of meeting users' requirements more effectively.

In the future, there are opportunities for improvement. Our emphasis on Ant Colony Optimization affords future opportunities to investigate task dependency, priority, and deadline considerations. Incorporating these factors can result in even greater system performance enhancements. Our study's innovative methodology, which integrates K-means clustering and ACO, provides a foundation for improving cloud resource management and load balancing as the cloud computing landscape evolves. Through meticulous analysis and comparative evaluation, we have demonstrated the extraordinary characteristics of our approach, highlighting its potential to reshape and advance cloud computing optimization.

REFERENCES

[1] Laha, S. R., Parhi, M., Pattnaik, S., Pattanayak, B. K., & Patnaik, S. (2020). Issues, Challenges and Techniques for Resource Provisioning in Computing Environment. In 2020 2nd International Conference on Applied Machine Learning (ICAML) (pp. 157-161). IEEE.

[2] Pattanaik, B. C., Sahoo, B. K., Pati, B., & Laha, S. R. (2023). Dynamic Fault Tolerance Management Algorithm for VM Migration in Cloud Data Centers. International Journal of Intelligent Systems and Applications in Engineering, 11(3), 85-96.

[3] Parhi, M., Pattanayak, B. K., & Patra, M. R. (2018). A multi-agent-based framework for cloud service discovery and selection

using ontology. Service Oriented Computing and Applications, 12, 137-154.

[4] Pati, A., Parhi, M., Alnabhan, M., Pattanayak, B. K., Habboush, A. K., & Al Nawayseh, M. K. (2023). An IoT-Fog-Cloud Integrated Framework for Real-Time Remote Cardiovascular Disease Diagnosis. In Informatics (Vol. 10, No. 1, p. 21). MDPI.

[5] Rao, M. N. . (2023). A Comparative Analysis of Deep Learning Frameworks and Libraries. International Journal of Intelligent Systems and Applications in Engineering, 11(2s), 337–342. Retrieved from https://ijisae.org/index.php/IJISAE/article/view/2707

[6] Pati, A., Parhi, M., & Pattanayak, B. K. (2022). Heartfog: Fog computing enabled ensemble deep learning framework for automatic heart disease diagnosis. In Intelligent and Cloud Computing: Proceedings of ICICC 2021 (pp. 39-53). Singapore: Springer Nature Singapore.

[7] Dr. S. Praveen Chakkravarthy. (2020). Smart Monitoring of the Status of Driver Using the Dashboard Vehicle Camera. International Journal of New Practices in Management and Engineering, 9(01), 01 - 07. https://doi.org/10.17762/ijnpme.v9i01.81

[8] Pattanayak, B. K., Pattnaik, O., & Pani, S. (2021). Dealing with Sybil attack in VANET. In Intelligent and Cloud Computing: Proceedings of ICICC 2019, Volume 1 (pp. 471-480). Springer Singapore.

[9] Katal, A., Dahiya, S., & Choudhury, T. (2023). Energy efficiency in cloud computing data centers: a survey on software technologies. Cluster Computing, 26(3), 1845-1875.

[10] Al Nuaimi, K., Mohamed, N., Al Nuaimi, M., & Al-Jaroodi, J. (2012). A survey of load balancing in cloud computing: Challenges and algorithms. In 2012 second symposium on network cloud computing and applications (pp. 137-142). IEEE.

[11] Hussein, W., Peng, T., & Wang, G. (2015). A weighted throttled load balancing approach for virtual machines in cloud environment. International Journal of Computational Science and Engineering, 11(4), 402-408.

[12] Paul Garcia, Ian Martin, Laura López, Sigurðsson Ólafur, Matti Virtanen. Enhancing Student Engagement through Machine Learning: A Review. Kuwait Journal of Machine Learning, 2(1). Retrieved from http://kuwaitjournals.com/index.php/kjml/article/view/163

[13] Deepa, T., & Cheelu, D. (2017). A comparative study of static and dynamic load balancing algorithms in cloud computing. In 2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS) (pp. 3375-3378). IEEE.

[14] 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, 32(2), 149-158.

[15] Chraibi, A., Ben Alla, S., & Ezzati, A. (2021). Makespan optimisation in cloudlet scheduling with improved DQN algorithm in cloud computing. Scientific Programming, 2021, 1-11.

[16] LD, D. B., & Krishna, P. V. (2013). Honey bee behavior inspired load balancing of tasks in cloud computing environments. Applied soft computing, 13(5), 2292-2303.

[17] Sahoo, D. K. . (2022). A Novel Method to Improve the Detection of Glaucoma Disease Using Machine Learning. Research Journal of Computer Systems and Engineering, 3(1), 67–72. Retrieved from https://technicaljournals.org/RJCSE/index.php/journal/article/view/44

[18] Remesh Babu, K. R., & Samuel, P. (2016). Enhanced bee colony algorithm for efficient load balancing and scheduling in cloud. In Innovations in Bio-Inspired Computing and Applications: Proceedings of the 6th International Conference on Innovations in Bio-Inspired Computing and Applications (IBICA 2015) held in Kochi, India during December 16-18, 2015 (pp. 67-78). Springer International Publishing.

[19] Singh, A., Juneja, D., & Malhotra, M. (2015). Autonomous agent based load balancing algorithm in cloud computing. Procedia Computer Science, 45, 832-841.

[20] Madni, S. H. H., Latiff, M. S. A., Coulibaly, Y., & Abdulhamid, S. I. M. (2017). Recent advancements in resource allocation techniques for cloud computing environment: a systematic review. cluster computing, 20, 2489-2533.

[21] Mapetu, J. P. B., Chen, Z., & Kong, L. (2019). Low-time complexity and low-cost binary particle swarm optimization algorithm for task scheduling and load balancing in cloud computing. Applied Intelligence, 49, 3308-3330.q