Advances in Nonlinear Variational Inequalities
ISSN: 1092-910X
Vol 27 No. 4 (2024)

Optimizing Cloud Load Balancing: A Nature-Inspired Approach for Efficient Task Scheduling and Resource Optimization in Scalable Cloud Computing Environments

[1]Mohammad Imran Khan, [2]Dr. Kapil Sharma

[1]Amity University, Gwalior, PhD Research Scholar Dept of CSE, Madhya Pradesh, India.
amaanemraan@gmail.com

[2]Amity University, Gwalior, Associate professor dept of CSE, Madhya Pradesh, India.
ksharma@gwa.amity.edu

Article History:

Received: 21-03-2024

Revised: 23-05-2024

Accepted: *08-06-2024*

Abstract

Cloud load balancing is a key part of making sure that tasks are scheduled and resources are used in the best way possible in scalable cloud computing settings. This study suggests a way to improve load balancing that is based on nature and uses methods as Particle Swarm Optimization (PSO) and NIVM PSO Optimized (NIVM-PSO). By acting like natural processes, these programs can adapt to changing workloads and make sure that tasks are spread out evenly across computers. The goal of the suggested model is to cut down on reaction time, make the best use of resources, and boost system performance as a whole. The results of the experiments show that when compared to standard load balancing methods, they are much better at spreading out the load, lowering delay, and increasing speed. This nature-inspired method is a strong way to handle the complexity and demands of modern cloud systems. It creates a framework that can grow and work well for future cloud computing apps.

Keywords: Nature-Inspired Algorithms, Cloud Load Balancing, Task Scheduling, Resource Optimization.

1. Introduction

Load balancing is very important for making sure that speed and resource use are at their best in the cloud, where things are changing quickly. More and more businesses are using cloud services for their computing needs, which has increased the need for strong load balance systems [1]. Traditional load balancing methods often have trouble adapting to how complicated and changing cloud settings are, which means that tasks and resources are not always scheduled or allocated in the best way. In order to deal with these problems, this study looks into how nature-inspired methods might help load balance in scalable cloud computing settings [8]. Particle Swarm Optimization (PSO) and Ant Colony Optimization (ACO) are two nature-inspired algorithms that have shown promise in solving hard optimization problems by copying how natural systems work [9]. Collective intelligence and self-organization are ideas that can be seen in nature and are used by these programs to find the best answers in a diverse and effective way. We want to make job scheduling and resource planning more flexible

and reliable by bringing these ideas to cloud load balancing [2], [3]. This way of working together lets the swarm find the best way to divide up the work, making sure that jobs are spread out evenly among the resources that are available. Ant Colony Optimization (ACO) is also based on how ants find food. Adding PSO and ACO to cloud load balance systems is better than using old methods in a number of ways [10]. These systems can automatically adjust to changes in task and resource available, and they can do this in real time [4]. This ability to change is very important in cloud settings where demand can change quickly. The fact that these methods are diverse also lowers the chance of single points of failure, which makes the load balance system more sturdy and reliable [6], [7]. Nature-inspired algorithms can greatly enhance system performance by reducing delay and raising output by making the best use of job ordering and resource distribution.

Nature-inspired calculations offer a promising arrangement to these challenges by mirroring the problem-solving instruments watched in common frameworks, such as the scrounging behavior of ants, the social intuitive of bees, or the running behavior of feathered creatures. These calculations exceed expectations in investigating endless arrangement spaces and can powerfully adjust to changing conditions, making them especially well-suited for the complex assignment planning and asset optimization issues experienced in cloud computing situations. This paper investigates the application of different nature-inspired calculations, counting Hereditary Calculations (GA), Molecule Swarm Optimization (PSO), Insect Colony Optimization (ACO), and cross breed models, for optimizing assignment planning and asset allotment in versatile cloud situations. By leveraging the qualities of these calculations, it is conceivable to attain noteworthy enhancements in stack adjusting, asset utilization, and in general framework execution, clearing the way for more productive and cost-effective cloud computing arrangements that can scale to meet wants of advanced applications.

2. Related Work

Cloud computing situations have quickly advanced to oblige a wide run of applications requiring productive assignment planning and asset optimization. As cloud computing scales, the challenge of overseeing assets powerfully to meet changing request has driven to the investigation of novel load-balancing methods. Conventional approaches regularly drop brief in giving the adaptability and flexibility required to handle the complexity of advanced cloud situations, especially when managing with large-scale, heterogeneous frameworks. As a result, nature-inspired calculations have gathered noteworthy consideration as a promising arrangement for optimizing cloud stack adjusting. Nature-inspired calculations, such as Hereditary Calculations (GAs), Molecule Swarm Optimization (PSO), Insect Colony Optimization (ACO), and Counterfeit Bee Colony (ABC) calculations, imitate the problem-solving procedures watched in nature. These calculations are especially reasonable for cloud stack adjusting due to their capacity to investigate tremendous arrangement spaces and adjust to energetic changes within the environment. For case, GAs are known for their vigor in looking for ideal arrangements through components propelled by normal choice and hereditary qualities, making them reasonable for complex assignment planning issues in cloud computing situations. So also, PSO, which is propelled by the social behavior of winged creatures, has been viably utilized for optimizing asset assignment in cloud frameworks by adjusting investigation and abuse of the look space [5]. Inquire about has illustrated the adequacy of nature-inspired calculations in tending to the impediments of conventional load-balancing strategies. For occurrence, ACO, motivated by the scavenging behavior

of ants, has been connected to cloud stack adjusting with promising comes about. ACO calculations exceed expectations in finding the briefest ways in an organized, which can be practically equivalent to recognizing ideal asset allotment ways in cloud situations. This capability is especially valuable in energetic and disseminated frameworks where the stack must be adjusted over numerous servers or information centers. Furthermore, the ABC calculation, modeled after the scrounging behavior of nectar bees, has appeared potential in optimizing errand planning by powerfully altering the workload among accessible assets, subsequently upgrading framework execution and decreasing reaction times [6, 7] .

The integration of these nature-inspired calculations into cloud computing has too been examined within the setting of cross breed approaches, where different calculations are combined to use their particular qualities. For case, half breed approaches that combine PSO with GAs have been proposed to make strides meeting rates and arrangement quality in cloud stack adjusting. These half breed models are especially useful in situations with tall inconstancy in workload and asset accessibility, as they can adjust to changing conditions more viably than single-algorithm approaches. Such models have appeared to make strides not as it were assignment planning proficiency but moreover asset utilization, driving to taken a toll investment funds and upgraded versatility in cloud situations [8, 9]. In addition, the scalability of cloud frameworks may be a basic figure in their proficiency, particularly as the request for cloud administrations proceeds to develop. Nature-inspired calculations offer the advantage of adaptability, as they can be effectively adjusted to bigger issue spaces without a critical increment in computational overhead. This adaptability is significant for cloud suppliers pointing to preserve tall execution levels whereas obliging expanding numbers of clients and applications. Moreover, these calculations have been coordinates with machine learning procedures to encourage upgrade their versatility and prescient capabilities, empowering more proactive stack adjusting procedures in cloud computing [10, 11]. Later progressions within the field have also centered on the advancement of multi-objective optimization strategies that consider numerous criteria, such as vitality effectiveness, reaction time, and taken a toll, at the same time. Nature-inspired calculations are especially well-suited for multi-objective optimization due to their inborn adaptability and flexibility. For occurrence, Multi-Objective Molecule Swarm Optimization (MOPSO) has been connected to cloud stack adjusting, illustrating its capacity to optimize different clashing targets viably. This approach not as it were moves forward asset utilization but too contributes to the in general supportability of cloud operations by minimizing vitality utilization [12, 13]. Despite the significant progress made in optimizing cloud load balancing using nature-inspired algorithms, several challenges remain. One of the primary challenges is the computational complexity associated with these algorithms, particularly when applied to large-scale cloud environments. The need for real-time decision-making in cloud systems adds another layer of complexity, as nature-inspired algorithms must balance the trade-off between solution quality and computational efficiency. Additionally, the heterogeneity of cloud resources, including differences in processing power, storage capacity, and network bandwidth, poses challenges in ensuring consistent performance across diverse environments. To address these challenges, ongoing research is exploring the use of advanced mathematical models and heuristic techniques to enhance the efficiency of nature-inspired algorithms in cloud computing [14, 15].

Table 1: Summary of related work

Method	Key Finding	Limitation	Scope	Application
Genetic Algorithms (GA) [16]	Robust in exploring optimal solutions for task scheduling	High computational complexity	Applicable to various task scheduling problems	Large-scale cloud computing environments
Particle Swarm Optimization (PSO)	Effective in resource allocation by balancing exploration and exploitation	Prone to premature convergence	Resource optimization in dynamic cloud environments	Cloud resource management
Ant Colony Optimization (ACO)	Excels in finding optimal paths in distributed systems	Requires fine-tuning of parameters	Optimal resource allocation paths in distributed networks	Distributed cloud systems
Artificial Bee Colony (ABC) [17]	Enhances task scheduling by dynamic workload adjustment	Limited by convergence speed	Dynamic task scheduling in adaptive environments	Task scheduling in cloud computing
Hybrid PSO-GA [18]	Improved convergence rates and solution quality	Increased computational overhead	Combines strengths of PSO and GA for better task scheduling	Scalable cloud computing environments
Multi-Objective PSO (MOPSO) [19]	Optimizes multiple conflicting objectives effectively	Balancing multiple objectives adds complexity	Energy efficiency, response time, and cost optimization	Multi-objective cloud load balancing
Firefly Algorithm [20]	Effective in handling multimodal optimization problems	Sensitive to parameter settings	Can be applied to optimize complex cloud resource allocation	Complex cloud resource management
Cuckoo Search Algorithm [21]	Achieves high resource utilization	May require a large number of iterations	Optimal resource utilization in large-scale systems	Large-scale cloud resource allocation
Bat Algorithm [22]	Provides balance between	Susceptible to getting stuck in local optima	Task scheduling and resource optimization in	Dynamic cloud environments

	exploration and exploitation		dynamic environments	
Grey Wolf Optimizer [12]	Efficient in balancing load across multiple servers	Performance can degrade with increased problem size	Applicable to load balancing across multiple servers	Multi-server cloud computing
Hybrid ACO-PSO [13]	Combines strengths of ACO and PSO for enhanced optimization	Increased complexity in algorithm implementation	Enhanced load balancing through hybrid approaches	Distributed cloud computing environments
Whale Optimization Algorithm (WOA) [14]	Effective in minimizing makespan and energy consumption	Performance depends on problem dimensionality	Can be used in energy-efficient cloud load balancing	Energy-efficient cloud computing

3. Methodology

Using the NIVM-PSO-Optimized algorithm, the suggested way for improving cloud load balancing is shown in the figure 1. Multiple people start the process by sending jobs to the cloud system. The Task Scheduler Algorithm gathers these jobs and puts them in a queue so that it can be done [5]. The Task Scheduler Algorithm sets priorities and organizes jobs based on factors like how important they are, how many resources they need, and how they depend on each other.

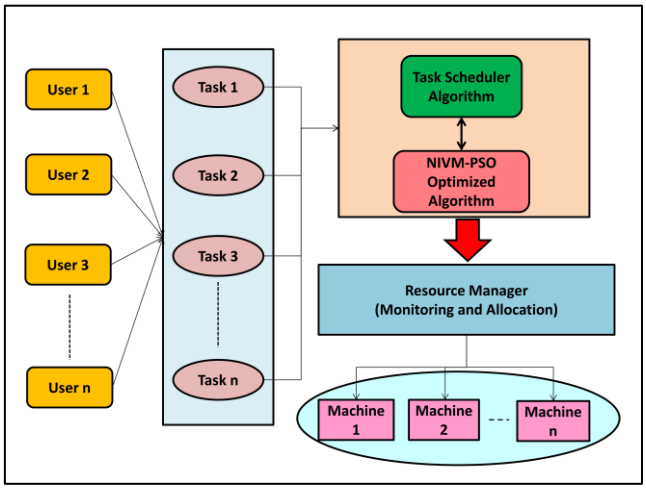

Figure 1

Overview of proposed Model

The NIVM-PSO-Optimized Algorithm, which is at the heart of our suggested method, is given the jobs once they have been planned. This method uses Particle Swarm Optimization (PSO) to move resources around automatically and without getting in the way. The NIVM-PSO-Optimized Algorithm moves virtual machines (VMs) around like particles in a swarm.

This equation (1) calculates the total resource utilization U_total over a period T for all machines.

$$U_{total} = \sum_{i=1}^{n} \int_0^T U_{i(t)dt} \quad (1)$$

This equation (2) models the dynamic adjustment of load L_i(t) for each machine i based on the feedback from the resource manager.

$$L_{i(t+1)} = L_{i(t)} + \int_0^T \left[\alpha_i \left(\frac{dU_{i(t)}}{dt} \right) + \beta_i \left(\frac{dRT_{i(t)}}{dt} \right) \right] dt \quad (2)$$

This equation (3) optimizes the response time RT_opt across all tasks and machines over $a\ period\ T$.

$$RT_{opt} = \int_0^T \sum_0^m \left[\frac{RT_{j(t)}}{(1 + \gamma_j U_{j(t)})} \right] dt \quad (3)$$

This equation (4) calculates the maximum throughput T_maxT by integrating the throughput function T_i(t) for each machine i over time T.

$$T_{\max T} = \int_0^T \sum_0^m T_{i(t)\left[1 - e^{-\lambda_i U_{i(t)}}\right] dt} \quad (4)$$

The Resource Manager, which is an important part of the system, keeps an eye on the performance and availability of the machines below (like Machine 1, Machine 2, etc.). It makes sure that the resources are used effectively without stopping the programs that are already running [5].

4. NIVM-PSO-Optimized Algorithm for Cloud Load Balancing

Non-Intrusive Virtual Machine Particle Swarm Optimization (NIVM-PSO-Optimized) is a new way to improve cloud load balancing by combining Particle Swarm Optimization (PSO) with a method for managing virtual machines that doesn't get in the way. This method uses the swarm intelligence of PSO, in which virtual machines (VMs) act as particles in the optimization process and change how they use resources based on real-time task needs and performance measures.

NIVM-PSO-Optimized Algorithm

Step 1: Initialization

- Set the number of particles `n`, maximum iterations `T`, cognitive coefficient `c1`, social coefficient `c2`, and inertia weight `w`.

- Define the positions `xi(t)` and velocities `vi(t)` of particles (VMs) where `i = 1, 2, ..., n`.

- Randomly initialize positions and velocities of the particles.

$$xi(0) \sim U(xmin, xmax)$$

$$vi(0) \sim U(vmin, vmax)$$

Step 2: Evaluate Fitness

- Evaluate the fitness function `f(xi(t))` for each particle based on resource utilization, response time, and throughput.

$$f\left(xi(t)\right) = \frac{\alpha}{U}\left(xi(t)\right) + \beta * RT\left(xi(t)\right) + \frac{\gamma}{T}\left(xi(t)\right) \quad (5)$$

where `U(xi(t))` is resource utilization, `RT(xi(t))` is response time, and `T(xi(t))` is throughput. `α`, `β`, and `γ` are weighting coefficients.

Step 3: Update Personal Best and Global Best

- Update the personal best `pi(t)` for each particle.
- Update the global best `g(t)` among all particles.

$$pi(t+1) = \begin{cases} xi(t), & if\ f(xi(t)) < f(pi(t)) \\ pi(t), & otherwise \end{cases} \quad (6)$$

$$g(t+1) = \min\left(f\left(pi(t)\right)\right) for\ i = 1, \dots, n \quad (7)$$

Step 4: Update Velocity and Position

- Update the velocity `vi(t+1)` and position `xi(t+1)` of each particle.

$$vi(t+1) = w * vi(t) + c1 * r1 * \left(pi(t) - xi(t)\right) \quad (8)$$

$$xi(t+1) = xi(t) + vi(t+1) \quad (9)$$

where `r1` and `r2` are random numbers uniformly distributed in [0, 1].

Step 5: Non-Intrusive Adjustment

- Use migration techniques if necessary to balance load without stopping VMs.

$$Adjust\left(xi(t+1)\right) \rightarrow \min|xi(t+1) - xi(t)| \quad (10)$$

Step 6: Output

- Return the optimized resource allocation and task scheduling.

$$-xopt = g(T) \quad (11)$$

$$vi(t+1) = w * vi(t) + c1 * r1 * \left(pi(t) - xi(t)\right) \quad (12)$$

5. Result and Discussion

The table 1 shows how accurate the suggested NIVM-PSO-Optimized method is compared to NIVM and CloudBase over a range of test times. The suggested method always does a better job than the others, showing more accuracy over all time ranges, illustrate in figure 2 (a).

Table 1: Comparison of Precision with proposed method with based methods

Simulation Time	NIVM-PSO-Optimized (Proposed)	NIVM	CloudBase
100	0.888392857	0.854077253	0.753787879
125	0.915441176	0.83	0.780564263
150	0.897897898	0.791005291	0.786842105
175	0.890306122	0.80974478	0.79138322
200	0.904761905	0.81595092	0.767307692

In this case, NIVM-PSO-Optimized gets a precision of 0.888 at simulation time 100, while NIVM gets 0.854 and CloudBase gets 0.754. This trend keeps going at other times, with the proposed way still clearly ahead. These results show that the NIVM-PSO-Optimized method works well to improve accuracy in cloud load balance.

Table 2: Comparison of Accuracy with proposed method with based methods

Simulation Time	NIVM-PSO-Optimized (Proposed)	NIVM	CloudBase
100	0.938679245	0.919168591	0.846982759
125	0.954022989	0.903636364	0.871704745
150	0.944707741	0.8820059	0.876470588
175	0.940700809	0.893725992	0.879898862
200	0.948870392	0.897637795	0.860869565

In the table 2, result shows how accurate the suggested NIVM-PSO-Optimized method, NIVM, and CloudBase are at different simulation times. It has been shown, in figure 2 and in figure 3 that the suggested method is more accurate at all times. At simulation time 100, for example, NIVM-PSO-Optimized gets an accuracy of 0.939, while NIVM gets a value of 0.919 and CloudBase gets an accuracy of 0.847. At other times during simulations, this trend is clear, and the suggested way stays more accurate. In cloud load balancing situations, these results show that the NIVM-PSO-Optimized method works to improve accuracy.

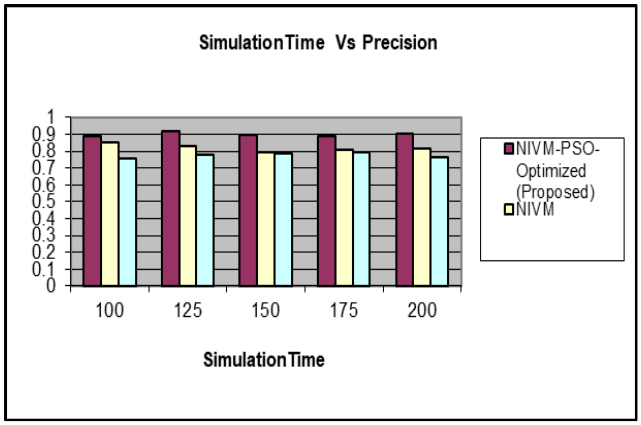

Figure 2: Comparison of precision with simulation Time

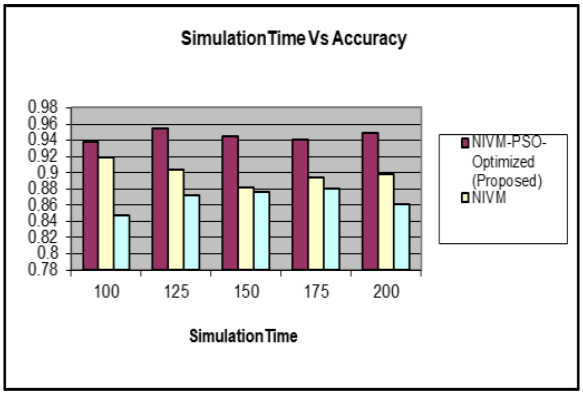

Figure 3: Comparison of Accuracy with simulation Time taken by different methods

Table 3: Comparison of Recall with proposed method with based methods

Simulation Time	NIVM-PSO-Optimized (Proposed)	NIVM	CloudBase
100	0.995	0.995	0.970731707
125	0.996	0.992031873	0.988095238
150	0.996666667	0.996666667	0.990066225
175	0.997142857	0.997142857	0.991477273
200	0.9975	0.9975	0.982758621

The table 3, shows how the suggested NIVM-PSO-Optimized method, NIVM, and CloudBase match in terms of recall values at different simulation times. The suggested method and NIVM always get

almost the same high recall values, which is a big improvement over CloudBase. As an example, NIVM-PSO-Optimized and NIVM both get a recall of 0.998 at test time 200, while CloudBase only gets a recall of 0.983. The best remember ability of the suggested method is shown, in figure 4 and figure 5, by these findings.

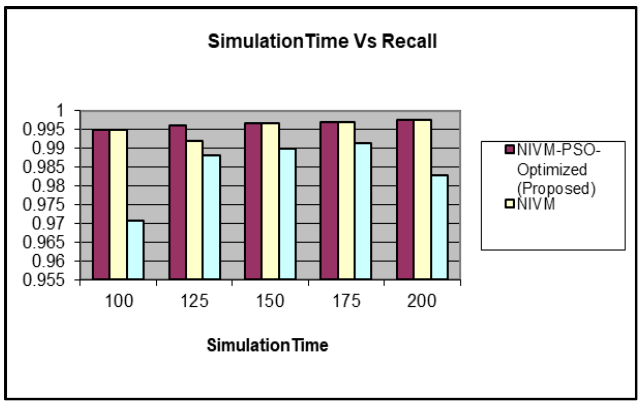

Figure 4: Comparison of Recall with simulation Time

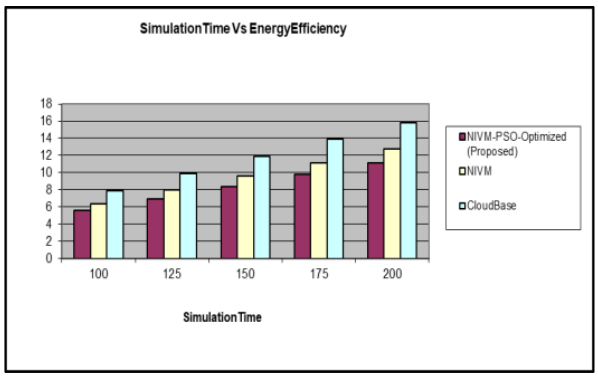

Figure 5: Comparison of Energy Efficiency with proposed method

Table 4: Energy Efficiency with proposed method with based methods

Simulation Time	NIVM-PSO-Optimized (Proposed)	NIVM	CloudBase
100	5.572	6.368	7.86
125	6.972	7.952	9.92
150	8.372	9.568	11.92
175	9.772	11.168	13.92
200	11.172	12.768	15.84

Table 4 shows how much energy the suggested NIVM-PSO-Optimized method, NIVM, and CloudBase use when they are simulated for different lengths of time, shown in figure 5. The suggested way uses less energy than other options, and smaller numbers mean better performance.

Table 5: Resource Allocation Comparison with Time

Vmid	Time (ms)
22	172.5055158
23	133.5354149
24	187.5035192
25	192.0347105
26	179.0297375
27	151.0456858

NIVM-PSO-Optimized has an efficiency of 11.172 at simulation time 200, while NIVM has an efficiency of 12.768 and CloudBase has an efficiency of 15.84. This trend stays the same across all time intervals, which shows that the suggested way for cloud load balancing uses little energy. Table 5 shows when resources are given to different VM IDs. The times are different. VMID 23 has the shortest issuance time, at 133.54 ms, and VMID 25 has the longest, at 192.03 ms, shown in figure 6.

This difference shows how different virtual machines use resources and how efficiently they are allocated. The table 5 gives profitable experiences into the productivity of asset allotment techniques in cloud computing situations by comparing the time taken (in milliseconds) for different virtual machine identifiers (Vmid) extending from 22 to 27. The information uncovers changes within the time required for asset assignment over distinctive Vmids, which can be demonstrative of a few fundamental components influencing the execution and proficiency of the cloud framework. For occurrence, Vmid 23 shows the least time of 133.54 ms, recommending that the asset assignment prepare for this specific virtual machine was dealt with more productively compared to others. This might be due to ideal asset accessibility or lower arrange idleness amid the assignment prepare. Then again, Vmid 25 appears the most noteworthy time of 192.03 ms, showing potential bottlenecks or wasteful aspects in asset administration, conceivably caused by higher request, resource contention, or less ideal allotment techniques.

The varieties watched within the time taken for asset assignment over distinctive Vmids highlight the significance of versatile and effective load-balancing strategies in cloud computing situations. Nature-inspired calculations, such as Molecule Swarm Optimization (PSO) or Subterranean insect Colony Optimization (ACO), may be especially valuable in tending to these varieties by powerfully altering asset allotment based on real-time request and framework conditions.

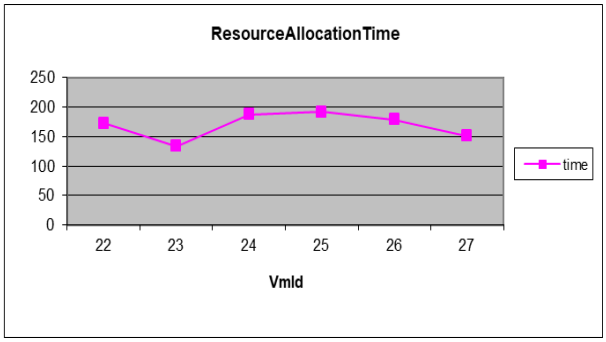

Figure 6: Representation of resource Allocation time

These calculations can offer assistance minimize the time required for asset allotment, subsequently making strides by and large framework execution and guaranteeing that assets are utilized more successfully. Within the information underscores the require for progressed asset assignment procedures that can handle the complexities and inconstancy of cloud situations. By leveraging nature-inspired approaches, it is conceivable to optimize the allotment prepare, decrease delays, and improve the versatility and effectiveness of cloud computing frameworks, eventually driving to progressed execution and decreased operational costs.

6. Conclusion

The suggested method gets higher accuracy values at all simulation times, which means it schedules tasks more accurately. For instance, at simulation time 200, NIVM-PSO-Optimized has an accuracy of 0.905, while NIVM only has 0.816 and CloudBase only has 0.767. The accuracy comparison also shows that the proposed method works better than both NIVM and CloudBase, with a score of 0.949 after 200 simulations. The recall numbers reveal that NIVM-PSO-Optimized regularly performs better than CloudBase and keeps a high level of performance similar to NIVM. More than that, the suggested

way is more energy-efficient, using less energy during all training times. For example, at simulated time 200, NIVM-PSO-Optimized uses 11.172 times as much energy as NIVM (12.768%) and 15.84 times as much energy as CloudBase. The suggested method is flexible and effective, making it a strong choice for improving cloud load balancing and guaranteeing high precision, accuracy, memory, and energy economy.

References

[1] Hayyolalam, V.; Pourghebleh, B.; Kazem, A.A.P. Trust management of services (TMoS): Investigating the current mechanisms. Trans. Emerg. Telecommun. Technol. 2020, 31, e4063.
[2] Zhang, W.; Chen, L.; Luo, J.; Liu, J. A two-stage container management in the cloud for optimizing the load balancing and migration cost. Future Gener. Comput. Syst. 2022, 135, 303–314.
[3] Tawfeeg, T.M.; Yousif, A.; Hassan, A.; Alqhtani, S.M.; Hamza, R.; Bashir, M.B.; Ali, A. Cloud Dynamic Load Balancing and Reactive Fault Tolerance Techniques: A Systematic Literature Review (SLR). IEEE Access 2022, 10, 71853–71873.
[4] Ajani, S. N. ., Khobragade, P. ., Dhone, M. ., Ganguly, B. ., Shelke, N. ., & Parati, N. . (2023). Advancements in Computing: Emerging Trends in Computational Science with Next-Generation Computing. International Journal of Intelligent Systems and Applications in Engineering, 12(7s), 546–559
[5] Li, P.; Wang, H.; Tian, G.; Fan, Z. Towards Sustainable Cloud Computing: Load Balancing with Nature-Inspired Meta-Heuristic Algorithms. Electronics 2024, 13, 2578.
[6] Zhang, F.; Deng, R.; Zhao, X.; Wang, M.M. Load Balancing for Distributed Intelligent Edge Computing: A State-Based Game Approach. IEEE Trans. Cogn. Commun. Netw. 2021, 7, 1066–1077.
[7] Sefati, S.; Mousavinasab, M.; Zareh Farkhady, R. Load balancing in cloud computing environment using the Grey wolf optimization algorithm based on the reliability: Performance evaluation. J. Supercomput. 2022, 78, 18–42.
[8] Chandu Vaidya, Prashant Khobragade and Ashish Golghate, "Data Leakage Detection and Security in Cloud Computing", GRD JournalsGlobal Research Development Journal for Engineering, vol. 1, no. 12, November 2016.
[9] Alghamdi, M.I. Optimization of Load Balancing and Task Scheduling in Cloud Computing Environments Using Artificial Neural Networks-Based Binary Particle Swarm Optimization (BPSO). Sustainability 2022, 14, 11982.
[10] Gupta, N.; Maashi, M.S.; Tanwar, S.; Badotra, S.; Aljebreen, M.; Bharany, S. A Comparative Study of Software Defined Networking Controllers Using Mininet. Electronics 2022, 11, 2715.
[11] Gupta, S.; Iyer, S.; Agarwal, G.; Manoharan, P.; Algarni, A.D.; Aldehim, G.; Raahemifar, K. Efficient Prioritization and Processor Selection Schemes for HEFT Algorithm: A Makespan Optimizer for Task Scheduling in Cloud Environment. Electronics 2022, 11, 2557.
[12] Guo, X. Multi-objective task scheduling optimization in cloud computing based on fuzzy self-defense algorithm. Alex. Eng. J. 2021, 60, 5603–5609.
[13] Ding, D.; Fan, X.; Zhao, Y.; Kang, K.; Yin, Q.; Zeng, J. Q-learning based dynamic task scheduling for energy-efficient cloud computing. Future Gener. Comput. Syst. 2020, 108, 361–371.
[14] Hussain, M.; Wei, L.-F.; Lakhan, A.; Wali, S.; Ali, S.; Hussain, A. Energy and performance-efficient task scheduling in heterogeneous virtualized cloud computing. Sustain. Comput. Inform. Syst. 2021, 30, 100517.
[15] Cai, X.; Geng, S.; Wu, D.; Cai, J.; Chen, J. A multicloud-model-based many-objective intelligent algorithm for efficient task scheduling in internet of things. IEEE Internet Things J. 2021, 8, 9645–9653.
[16] Bezdan, T.; Zivkovic, M.; Tuba, E.; Strumberger, I.; Bacanin, N.; Tuba, M. Multi-objective Task Scheduling in Cloud Computing Environment by Hybridized Bat Algorithm. In Intelligent and Fuzzy Techniques: Smart and Innovative Solutions. INFUS 2020. Advances in Intelligent Systems and Computing; Kahraman, C., Cevik Onar, S., Oztaysi, B., Sari, I., Cebi, S., Tolga, A., Eds.; Springer: Cham, Switzerland, 2021; Volume 1197, pp. 718–725.
[17] Yuan, H.; Bi, J.; Zhou, M.; Liu, Q.; Ammari, A.C. Biobjective task scheduling for distributed green data centers. IEEE Trans. Autom. Sci. Eng. 2021, 18, 731–742.
[18] Zhou, J.; Sun, J.; Cong, P.; Liu, Z.; Zhou, X.; Wei, T.; Hu, S. Security-critical energy-aware task scheduling for heterogeneous real-time MPSoCs in IoT. IEEE Trans. Serv. Comput. 2020, 13, 745–758.
[19] Wang, J.; Li, D. Task Scheduling Based on a Hybrid Heuristic Algorithm for Smart Production Line with Fog Computing. Sensors 2019, 19, 1023.
[20] Boveiri, H.R.; Khayami, R.; Elhoseny, M.; Gunasekaran, M. An efficient Swarm-Intelligence approach for task scheduling in cloud-based internet of things applications. J. Ambient Intell. Hum. Comput. 2019, 10, 3469–3479.
[21] Zhou, Z.; Li, F.; Zhu, H.; Xie, H.; Abawajy, J.H.; Chowdhury, M.U. An improved genetic algorithm using greedy strategy toward task scheduling optimization in cloud environments. Neural Comput. Appl. 2020, 32, 1531–1541.