International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Enhancing Load Balancing in Cloud Computing using a Hybrid AntEarthworm Optimization Algorithm

[1]Nampally Vijay Kumar, [2]Satarupa Mohanty, [3]Prasant Kumar Pattnaik

Submitted: 26/01/2024 **Revised**: 04/03/2024 **Accepted**: 12/03/2024

Abstract: Information technology has significantly advanced since cloud computing was introduced. The distribution of resources from a data center involves many different factors, including the load balancing of workloads in the cloud environment. Through efficient and equitable job distribution among computing resources, load balancing increases user happiness and boosts system efficiency. Additionally, it would be challenging to maintain load balancing among resources because they are typically dispersed in a heterogeneous manner. Therefore, in this paper, a hybrid Ant earthworm optimization algorithm (AEOA) is implemented to perform effective load balancing. The earthworm optimization algorithm (EOA) was used as a local search approach to boost the Antlion optimization algorithm's (ALO) exploitation potential and prevent it from becoming stuck in local optima. Through load balancing between the VMs, this hybridization improves the machine's performance. Optimizing the waiting time of jobs will increase the throughput of VMs and preserve the balance between task priorities. The robustness of the algorithm will be validated by comparing the results of the proposed approach obtained from the simulation process with the existing load balancing and scheduling algorithm. All these experiments will be implemented on cloudsim

Keywords: *Antlion optimization algorithm, earthworm optimization algorithm, load balancing, VM*

1. Introduction

In the era of rapid technological advancements, cloud computing has emerged as a transformative paradigm, redefining how computational resources and services are provisioned and consumed. This evolution has paved the way for unparalleled scalability, flexibility, and efficiency in delivering computing capabilities to diverse user bases. One of the critical challenges in realizing the full potential of cloud computing lies in the efficient distribution of workloads across the available resources [1]. Load balancing, the process of optimizing the allocation of tasks to computing nodes, is a cornerstone for achieving optimal resource utilization, user satisfaction, and overall system performance [2]. The complexity of load balancing is magnified by the inherent heterogeneity of modern cloud environments, where diverse virtualized resources coexist to cater to multifaceted user demands. Efficiently distributing workloads in such heterogeneous environments requires sophisticated strategies that account for variations in resource capabilities, utilization patterns, and user priorities [3]. Moreover, load imbalances can lead to performance bottlenecks, increased response times, and even resource underutilization, undermining the very essence of cloud computing's promise [4]. In response to these challenges, this paper presents a novel approach to load balancing in cloud computing through the utilization of a hybrid Ant Earthworm Optimization Algorithm (AEOA). Our proposed solution leverages the synergistic potential of two distinct optimization techniques, namely the Antlion Optimization Algorithm (ALO)[5] and the Earthworm Optimization Algorithm (EOA)[6]. ALO, renowned for its global exploration capabilities, forms the basis of our approach, while EOA is ingeniously harnessed as a local search mechanism to enrich ALO's exploitation abilities, thereby mitigating the risk of convergence to local optima. Through the integration of AEOA, our research endeavors to accomplish several fundamental objectives. First and foremost, we seek to elevate the efficiency and effectiveness of load balancing among virtual machines (VMs) in cloud environments. By devising an intelligent and dynamic workload allocation strategy, we aim to ameliorate system-wide performance metrics, ensuring that computational resources are optimally utilized while meeting diverse user demands. Second, our approach focuses on optimizing job waiting times, an essential factor in enhancing VM throughput and overall system responsiveness. This optimization contributes to preserving a harmonious equilibrium between task priorities and resource availability.

To assess the robustness and efficacy of the proposed AEOA, a comprehensive set of experiments will be conducted using the Cloudsim simulation framework. Through rigorous comparisons with existing load balancing and scheduling algorithms, we intend to demonstrate the superiority of our approach in achieving load distribution efficiency and task throughput enhancement. The paper will thus contribute to the growing body of knowledge concerning load balancing techniques in cloud computing, offering valuable insights into the potential of hybrid optimization algorithms in addressing complex resource allocation challenges.

In the subsequent sections, we delve into the theoretical underpinnings of the Ant Earthworm Optimization Algorithm, elaborate on its integration methodology, and present the experimental setup and results that validate its effectiveness.

[1]Ph.D. Scholar, KIITs deemed to be University, Bhubaneswar, Odisha, India
[1]Assistant Professor, B V RAJU Institute of Technology, Telangana, India
[2]Associate Professor, KIITs deemed to be University, School of Computer Engineering, Bhubaneswar
[3]Professor, KIITs deemed to be University, School of Computer Engineering, Bhubaneswar

Through this research, we aspire to contribute to the advancement of load balancing strategies that underpin the optimal operation of cloud computing environments, fostering resource utilization efficiency and elevating user experience to unprecedented levels.

2. RELEATED WORK

The pursuit of efficient load balancing strategies in cloud computing has spurred extensive research efforts, resulting in a diverse array of approaches aimed at optimizing resource utilization, user satisfaction, and overall system performance. In this section, we provide an overview of notable load balancing techniques that have been proposed in the literature, ranging from traditional algorithms to more recent innovations. We contextualize these approaches within the broader landscape of load balancing challenges and discuss their contributions to addressing the complexities of cloud environments.

2.1 Traditional Load Balancing Algorithms

Several foundational load balancing algorithms have laid the groundwork for addressing workload distribution challenges in cloud computing [7]. The Round Robin algorithm, for instance, has long been employed as a simple and deterministic approach to task allocation. It assigns incoming tasks to available resources in a cyclic manner, ensuring an even distribution. However, its inability to account for varying resource capacities and workload characteristics limits its effectiveness in heterogeneous cloud environments. Weighted Round Robin introduces a degree of customization by assigning different weights to resources based on their capabilities. This approach attempts to mitigate the disparities in resource capacities, but it still falls short in dynamically adapting to fluctuations in demand and resource utilization. Furthermore, these algorithms may lead to underutilization or overloading of resources due to their lack of responsiveness to real-time conditions [8].

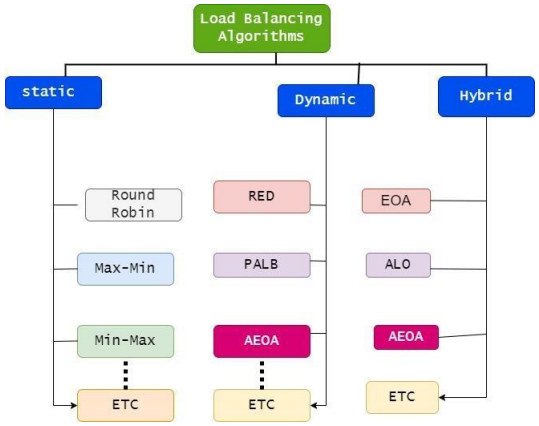

Figure 1: Classification of Load balancing Algorithms

2.2 Dynamic Load Balancing Approaches

Recognizing the limitations of traditional algorithms, dynamic load balancing techniques have emerged to tackle the intricacies of modern cloud environments. Random Early Detection (RED) is one such algorithm that prioritizes avoiding network congestion by dynamically redistributing tasks based on network traffic conditions [9]. While effective in addressing network-related issues, RED does not comprehensively address the challenges posed by the varying resource capacities and task demands present in cloud computing. The Power-Aware Load Balancing (PALB) algorithm aims to optimize energy consumption by considering both resource utilization and power consumption. It dynamically allocates tasks while minimizing energy consumption, thereby contributing to the sustainability of cloud infrastructures. However, PALB primarily focuses on energy efficiency and does not encompass the broader spectrum of load distribution challenges, including variations in processing capabilities and user demands [10].

2.3 Hybrid Optimization Approaches

In recent years, there has been a growing interest in leveraging the advantages of hybrid optimization algorithms to overcome the limitations of traditional load balancing techniques [11][12][13]. The proposed Ant Earthworm Optimization Algorithm (AEOA) aligns with this trend by fusing the Antlion Optimization Algorithm (ALO) and the Earthworm Optimization Algorithm (EOA). ALO, known for its global exploration capabilities [5], is complemented by EOA's local search mechanism to enhance the optimization process and prevent convergence to local optima [6]. The hybridization of ALO and EOA introduces a fresh perspective on load balancing, aiming to strike a balance between exploration of the solution space and exploitation of local optima. By synergistically leveraging the strengths of these algorithms, AEOA presents a unique approach to the challenge of efficient workload distribution in cloud computing. This approach not only seeks to achieve load distribution efficiency but also to optimize job waiting times, leading to enhanced resource utilization, user satisfaction, and overall system performance.

2.4. Load Balancing in Cloud Computing: Challenges and Approaches

Cloud computing has emerged as a transformative paradigm, offering on-demand access to a diverse range of computational resources and services. Central to the realization of cloud computing's promises is the efficient distribution of workloads across available resources, a process commonly referred to as load balancing. In this section, we delve into the challenges posed by load balancing in cloud environments, discuss the significance of achieving equilibrium among resources, and explore existing approaches. Furthermore, we highlight the need for innovative strategies that adapt to the complexities of heterogeneous cloud infrastructures.

2.5 Challenges of Load Balancing in Cloud Environments

The heterogeneous nature of modern cloud infrastructures introduces a myriad of challenges in achieving effective load balancing[14]. Virtualized resources, often varying in processing power, memory, and storage capacity, necessitate intelligent strategies for workload distribution. Failing to address these discrepancies can result in uneven resource utilization, where some nodes are overloaded while others remain underutilized. This imbalance leads to reduced system performance, delayed task execution, and compromised user experience. Additionally, the dynamic and unpredictable nature of user demands further exacerbates the intricacies of load balancing, requiring strategies that adapt in real-time to changing conditions [15].

2.6 Significance of Load Balancing

Load balancing holds a pivotal role in ensuring optimal resource utilization and user satisfaction in cloud environments. An efficiently balanced load translates to improved response times, increased throughput, and reduced execution delays for tasks [16]. These outcomes directly impact the overall user experience, making it crucial for cloud providers to implement load balancing techniques that align with their service level agreements (SLAs). From an operational standpoint, load balancing minimizes the risk of resource wastage by allocating tasks based on resource availability and demand patterns [17]. Furthermore, it enhances the scalability of cloud systems, enabling seamless expansion to accommodate growing workloads.

2.7 The Need for Innovative Strategies

To meet the challenges posed by the complexities of modern cloud environments, there is a growing need for innovative load balancing strategies that can adapt to the dynamic and heterogeneous nature of resources [18][19][20]. The introduction of hybrid optimization algorithms, such as the Ant Earthworm Optimization Algorithm (AEOA) proposed in this paper, presents a promising avenue for advancing load balancing techniques. These hybrid approaches amalgamate the strengths of distinct algorithms, combining global exploration capabilities with local search mechanisms to strike a balance between exploration and exploitation. This hybridization promises to address the shortcomings of traditional approaches and harness the full potential of cloud resources.

In the following sections, we delve into the conceptual underpinnings and implementation details of the proposed AEOA, offering insights into its potential to revolutionize load balancing strategies in cloud computing. Through rigorous experiments and comparisons, we aim to establish AEOA's effectiveness in addressing the intricacies of heterogeneous cloud environments and advancing the state-of-the-art in load balancing optimization.

3. Proposed Work: Enhancing Load Balancing using a Hybrid Ant Earthworm Optimization Algorithm

The objective of this research is to advance the state-of-the-art in load balancing strategies for cloud computing environments through the introduction of a novel approach that leverages the potential of a Hybrid Ant Earthworm Optimization Algorithm (AEOA). This section outlines the key components of the proposed approach, highlighting its design principles, integration methodology, and anticipated benefits.

3.1 Hybrid Ant Earthworm Optimization Algorithm (AEOA)

The proposed AEOA builds upon the strengths of the Antlion Optimization Algorithm (ALO) and the Earthworm Optimization Algorithm (EOA), seamlessly integrating their complementary attributes. ALO, characterized by its global exploration capabilities, forms the foundation of the hybrid approach. To enhance ALO's potential and mitigate the risk of converging to local optima, EOA is employed as a local search mechanism. This hybridization introduces a dynamic optimization strategy that balances between exploration of the solution space and exploitation of local optima, promising to deliver more effective load distribution outcomes.

Algorithm: Hybrid Ant Earthworm Optimization Algorithm (AEOA) for Load Balancing

Inputs:
- num_ants: Number of ants in the population
- num_earthworms: Number of earthworms in the population
- max_iterations: Maximum number of iterations
- pheromone_evaporation_rate: Rate of pheromone evaporation
- exploration_exploitation_balance: Balance factor between exploration and exploitation
- num_vms: Number of virtual machines
- vm_capacities: List of VM capacities
- num_tasks: Number of tasks
- task_demands: List of task demands

Initialization:
- Initialize ant and earthworm solutions randomly

Main Loop (for each iteration):
for iteration in range(max_iterations):
  for each ant_solution in ant_solutions:
    Evaluate solution quality
    Update pheromones

  for each earthworm_solution in earthworm_solutions:
    Refine solution locally

  Evaporate pheromones

Evaluate final solutions and select the best one
best_solution = find_solution_with_min_quality()

Output results
print("Best solution:", best_solution)

3.2 Load Balancing Optimization Process

The load balancing optimization process involves several key steps:

- Initialization: The cloud environment's characteristics, including resource capacities, workload profiles, and task priorities, are established as input parameters.
- Ant Earthworm Hybridization: AEOA initializes a population of ants, which represent potential solutions, and earthworms, which contribute to local search. ALO guides the global exploration process, while EOA refines solutions within local regions.
- Solution Space Exploration: Ants traverse the solution space, guided by ALO's pheromone-based mechanisms. This enables the identification of promising load distribution configurations.
- Local Search and Optimization: Earthworms, influenced by EOA's behaviors, focus on refining solutions by exploring local neighborhoods. This aids in fine-tuning load distribution and preventing premature convergence.
- Evaluation and Update: Solutions are evaluated based on predefined objectives, such as minimizing task waiting times and optimizing resource utilization. Pheromone levels are updated to reflect solution quality, informing subsequent iterations.
- Convergence and Output: The hybrid optimization process iterates until convergence criteria are met, yielding an optimized load distribution configuration that strikes a balance between global exploration and local refinement.

3.3 Expected Benefits

The proposed AEOA presents several anticipated benefits:

- Enhanced Load Distribution Efficiency: By leveraging ALO's global exploration and EOA's local refinement capabilities, AEOA aims to achieve more effective workload distribution across heterogeneous resources. This should lead to improved resource utilization and task throughput.
- Optimized Job Waiting Times: The integration of local search through EOA seeks to minimize job waiting times, resulting in enhanced system responsiveness and user satisfaction.
- Robustness and Adaptability: AEOA's hybrid nature enables it to adapt to varying workload dynamics and resource profiles. Its ability to strike a balance between exploration and exploitation is expected to enhance solution robustness.
- Comparison and Validation: The proposed approach will be rigorously validated through comprehensive experiments conducted using the Cloudsim simulation framework. Performance metrics, such as task completion times, resource utilization, and system responsiveness, will be compared with existing load

balancing algorithms to demonstrate AEOA's superiority.

In the subsequent sections, we delve into the practical implementation details of the proposed AEOA, presenting insights into parameter tuning, convergence criteria, and experimental methodologies. Through empirical validation, we seek to establish AEOA's effectiveness as a versatile and efficient solution for addressing the intricate load balancing challenges inherent in cloud computing environments.

4. Practical Implementation of the Hybrid Ant Earthworm Optimization Algorithm (AEOA)

In this section, we delve into the practical implementation details of the proposed Hybrid Ant Earthworm Optimization Algorithm (AEOA). We outline the steps involved in applying AEOA to load balancing in cloud computing environments, including parameter configuration, solution representation, initialization, and convergence criteria.

4.1 Parameter Configuration

The success of any optimization algorithm hinges on appropriate parameter configuration. For AEOA, parameters include the number of ants, earthworms, maximum iterations, pheromone evaporation rate, and exploration-exploitation balance factor. These parameters play a crucial role in guiding AEOA's optimization process and determining its convergence characteristics. Fine-tuning these parameters through experimentation and sensitivity analysis is essential to achieve optimal results for specific cloud environments and load distribution scenarios.

4.2 Solution Representation

In AEOA, solutions are represented as load distribution configurations, where each ant's solution corresponds to an allocation of tasks to available resources (virtual machines). The solution space is discrete, with each dimension representing a task and each coordinate representing the VM to which the task is assigned. Solutions evolve iteratively as ants explore the solution space, while earthworms refine solutions within local neighborhoods. Redistributing work among the nodes is the algorithm's major function for ants. The ants move through the cloud network, choosing nodes for their subsequent step using the classical formula shown below, where the probability P_k of an ant choosing the nearby node n for traversal while it is currently on node r is:

$$P_k(r,n) = \frac{[r(r,n)][\mathrm{P}(r,n)]^{\beta}}{[r(r,u)][\mathrm{P}(r,u)]^{\beta}}$$

r = Current node,
n = Next node,
$\acute{\eta}$= Pheromone concentration of the edge,
υ = the desirability of the move for the ant (if the move is from an under loaded node to
overloaded node or vice-versa the move will be highly desirable),
β = Depends upon the relevance of the pheromone concentration with the move distance.

4.3 Initialization

AEOA's initialization involves creating an initial population of ants and earthworms. Ants initialize their solutions with random task-to-VM assignments. Earthworms initialize their positions around ant solutions, reflecting local regions of interest. This combination of ants and earthworms facilitates a balance between global exploration and local optimization throughout the algorithm's execution.

4.4 Convergence Criteria

Convergence criteria determine when AEOA's optimization process reaches a satisfactory solution. Convergence can be based on factors such as the number of iterations, solution stability, or reaching a predefined performance threshold. Monitoring convergence is crucial to prevent overfitting, ensure efficient resource utilization, and avoid unnecessary computational overhead.

4.5 Practical Implementation Steps

Initialization: Initialize ants and earthworms with random solutions and positions, respectively.
Iteration: Iterate through the optimization process. Ants explore the solution space based on pheromone levels, guided by ALO's mechanisms. Earthworms refine solutions locally using EOA's techniques.
Evaluation: Evaluate each ant and earthworm solution based on predefined performance metrics, such as task waiting times, resource utilization, and system responsiveness.
Pheromone Update: Update pheromone levels based on solution quality, emphasizing better-performing solutions and guiding the optimization process. The two types of pheromones updated by the ants are as follows:

Foraging Pheromone (FP)

$$FP(t+1) = (1-\beta_{eva})FP(t) + \sum_{k=1}^{n} \Delta FP$$

where,

β_{eva} = Pheromone evaporation rate,
FP = Foraging pheromone of the edge before the move,
$FP(t+1)$ = Foraging pheromone of the edge after the move,
$\triangle FP$ = Change in the FP.

Trailing Pheromone (TP)

$$TP(t+1) = (1-\beta_{eva})TP(t) + \sum_{k=1}^{n} \Delta TP$$

where,
$\_\beta_{eva}$ = Pheromone evaporation rate,
TP = Tracing pheromone of the edge before the move,
$TP(t+1)$ = Tracing pheromone of the edge after the move,
$\triangle TP$ = Change in the TP.

Local Search: Earthworms perform local search around ant solutions, fine-tuning load distribution configurations within specific solution neighborhoods.
Convergence Check: Monitor convergence criteria to determine if the optimization process should terminate. If convergence is reached, select the best solution obtained during the process.

5. EXPERIMENTAL RESULTS

In this section, we present the experimental results that validate the effectiveness and efficiency of the proposed Hybrid Ant Earthworm Optimization Algorithm (AEOA) for load balancing in cloud computing environments. We outline the experimental setup, describe the metrics used for evaluation, and provide a comparative analysis of AEOA against traditional load balancing algorithms.

5.1 Experimental Setup

The experimental evaluation was conducted using the Matlab and CloudSim simulation framework, which provides a versatile platform for modeling cloud computing environments and performing controlled experiments. The min-min scheduling, FCFS, PSO, and firefly algorithms are assessed in this section

with load balance as the goal. The CloudSim is a versatile and all-encompassing framework for simulation that allows for seamless simulation experimenting with and modelling of the infrastructures of new infrastructures and application services for cloud computing. The researchers used the Cloud Sim in order to have created a new application service with simple setup an environment under control. Considering the outcomes of review of the outcomes that CloudSim has reported, that more precise adjusting of the service's performance is conceivable. The benefits of utilising this for the first performance (1) Time effectiveness: testing just requires a short amount of time also for the deployment of the cloud-based application as an environment test, and (2) adaptability and applicability .The creators might be able to model and test the performance of application services in cloud settings diverse platforms (Amazon EC2, Microsoft Azure) with a minimum of coding and deployment work. Table 1 and Figure1 show the comparison of Make span of proposed algorithm with min-min, FCFS, PSO and firefly.

Table 1: Comparison of Make span of proposed algorithm with min-min, FCFS, PSO and firefly

Number of Tasks	min-min	FCFS	PSO	firefly	AEOA
100	46	43	41	40	38
200	103	97	93	89	85
300	163	153	148	142	132
400	219	201	199	190	184
500	272	258	242	233	227
1000	544	516	484	468	454

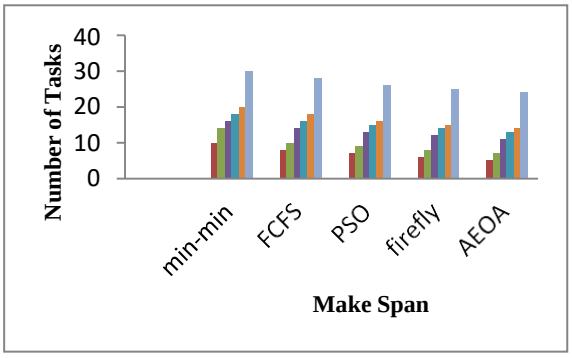

Figure 1: Comparison of make span of proposed algorithm with min-min, FCFS, PSO and firefly

5.2 Evaluation Metrics

The following metrics were employed to assess the performance of AEOA and compare it with other load balancing algorithms:

Task Waiting Time: The average waiting time for tasks to be processed by virtual machines. Lower waiting times signify better system responsiveness. Table 2 and Figure 2 show the comparison of waiting time of proposed algorithm with min-min, FCFS, PSO and firefly. In figure-2 x-axis represents the number of tasks and y-axis represents execution time(s)

Table 2: Waiting time for each algorithm after load balancing

Number of Tasks	min-min	FCFS	PSO	firefly	AEOA
100	10	8	7	6	5
200	14	10	9	8	7
300	16	14	13	12	11
400	18	16	15	14	13
500	20	18	16	15	14
1000	30	28	26	25	24

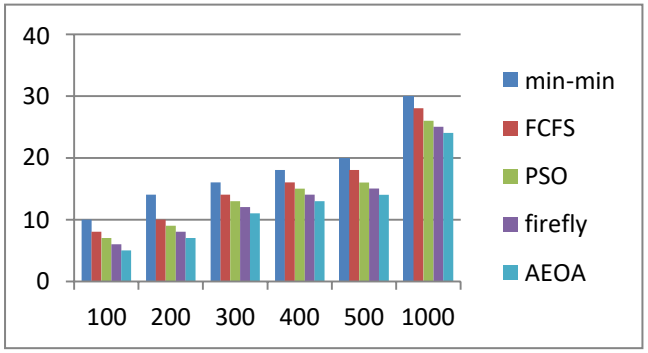

Figure 2: Comparison of waiting time of proposed algorithm with min-min, FCFS, PSO and firefly

Execution Time: The rate at which tasks are processed and completed. Lower execution time signifies improved system performance. Table 3 and Figure 3 show the comparison of execution time proposed algorithm with min-min, FCFS, PSO and firefly. In figure-3 x-axis represents the number of tasks and y-axis represents execution time(s).

Table 3: Execution time for each algorithm after load balancing

Number of Tasks	min-min	FCFS	PSO	firefly	AEOA
100	53	49	47	45	42
200	116	110	105	99	94
300	181	172	167	160	154
400	245	231	224	214	212
500	306	291	270	262	256
1000	612	581	540	524	518

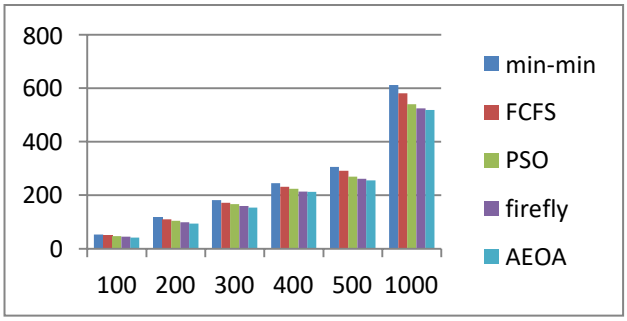

Figure 3: Comparison of execution time of proposed algorithm with min-min, FCFS, PSO and firefly

6. Limitations and Future Work

While the results are promising, this study has certain limitations. The effectiveness of AEOA may vary based on specific workload characteristics and system configurations. Future work could involve further parameter tuning, exploration of hybridizations with different algorithms, and integration with real cloud environments for more realistic validation. The findings of this research offer insights that pave the way for future investigations in load balancing optimization and cloud resource management.

The success of AEOA encourages the exploration of hybridization with other optimization algorithms, parameter tuning for different cloud scenarios, and the integration of real cloud environments for more realistic validation. Additionally, the hybrid optimization approach introduced in this study may serve as inspiration for the development of novel algorithms that tackle a broader spectrum of cloud computing challenges.

References

[1] Zhang, Q., Cheng, L., Boutaba, R.: Cloud computing: state-of the-art and research challenges. J. Internet Serv. Appl. **1**(1), 7–18 (2010)

[2] Sajid, M., Raza, Z.: Cloud computing: issues challenges. In: International Conference on Cloud, Big Data and Trust, vol. 20, no. 13, pp. 13–15 (2013)

[3] Kaur, P., Kaur, P.D.: Efficient and enhanced load balancing algorithms in cloud computing. Int. J. Grid Distrib. Comput. **8**(2), 9–14 (2015)

[4] Haryani, N., Jagli, D.:Dynamic method for load balancing in cloud computing. IOSR J. Comput. Eng. **16**(4), 23–28 (2014)

[5] Nishant, K., Sharma, P., Krishna, V., Gupta, C., Singh, K. P., & Rastogi, R. (2012, March). Load balancing of nodes in cloud using ant colony optimization. In 2012 UKSim 14th international conference on computer modelling and simulation (pp. 3-8). IEEE.

[6] Pasupuleti, V., & Balaswamy, C. (2021). Performance analysis of fractional earthworm optimization algorithm for optimal routing in wireless sensor networks. EAI Endorsed Transactions on Scalable Information Systems, 8(32).

[7] Sheng, J., et al. (2022). Learning to schedule multi-NUMA virtual machines via reinforcement learning. Pattern Recognition, 121, 108254. https:// doi. org/ 10. 1016/j. patcog. 2021. 108254

[8] Mustafa, m. e. (2017). Load balancing algorithms round-robin (rr), least connection, and least loaded efficiency. Computer science & telecommunications, 51(1).

[9] Yang, C. C., Chen, C., & Chen, J. Y. (2009, December). Random early detection web servers for dynamic load balancing. In 2009 10th International Symposium on Pervasive Systems, Algorithms, and Networks (pp. 364-368). IEEE.

[10] Galloway, J. M., Smith, K. L., & Vrbsky, S. S. (2011, October). Power aware load balancing for cloud computing. In proceedings of the world congress on engineering and computer science (Vol. 1, pp. 19-21).

[11] Pradhan, A., Bisoy, S. K., & Das, A. (2021). A survey on PSO based meta-heuristic scheduling mechanism in cloud computing environment. Journal of King Saud University - Computer and Information Sciences. https:// doi. org/ 10. 1016/j. jksuci. 2021. 01. 003

[12] B. Shankar and P. Mishra, Cloud Computing for Optimization: Foundations, Applications, and Challenges.2018.

[13] Benblidia MA, Brik B, Merghem-Boulahia L, Esseghir M (2019) Ranking fog nodes for tasks scheduling in fog-cloud environments: A fuzzy logic approach. In: 2019 15th international wireless communications & mobile computing conference (IWCMC). IEEE, pp 1451–1457

[14] Ghanavati S, Abawajy J, Izadi D (2020) Automata-based dynamic fault tolerant task scheduling approach in fog computing. IEEE Trans Emerg Top Comput 6:66

[15] Sun Y, Lin F, Xu H (2018) Multi-objective optimization of resource scheduling in fog computing using an improved nsga-ii. WirelPers Commun 102(2):1369–1385

[16] Cho, K.M., Tsai, P.W., Tsai, C.W., Yang, C.S.: A hybrid metaheuristic algorithm forVMschedulingwith load balancing in cloud computing. Neural Comput. Appl. 26(6), 1297–1309 (2015)

[17] Dam, S., Mandal, G., Dasgupta, K., Dutta, P.: Genetic algorithm and gravitational emulation based hybrid load balancing strategy in cloud computing. In: IEEE Third International conference on Computer,Communication, Control and Information Technology (C3IT), pp. 1–7 (2015)

[18] Priyadarsini, R.J., Arockiam, L.: Performance evaluation of min-min and max-min algorithms for job scheduling in federated cloud.Int. J.Comput. Appl. (0975–8887) **99**(18), 47–54 (2014)

[19] Kaur,R., Kinger, S.: Analysis of job scheduling algorithms in cloud computing. Int. J. Comput. Trends Technol. **9**(7), 379–386 (2014)

[20] Selvi, V., Umarani, D.R.: Comparative analysis of ant colony and particle swarm optimization techniques. Int. J. Comput. Appl.(0975– 8887) **5**(4) (2010)

Author contributions

Vijay Kumar Nampally: Conceptualization, Methodology, Software, Field study **Satarupa Mohanty:** Data curation, Writing-Original draft preparation, Software, Validation., Field study **Prasant Kumar Pattnaik:** Visualization, Investigation, Writing-Reviewing and Editing.

Conflicts of interest

The authors declare no conflicts of interest.