Optimized resource allocation in cloud computing for enhanced performance with modified particle swarm optimization

Sreenivasulu Gogula[1*], P. *Sridhar*[2], S. *Arvind*[3], *Abhisek* Sethy[4], S.D. *Prabu* Ragavendiran[5], *Pradeep* Balasubramani[6], and *Koppuravuri* Gurnadha Gupta[7]

[1]Department of Computer Science and Engineering (Data Science), Vardhaman College of Engineering, Shamshabad, Hyderabad, India, 501218
[2]Department of EEE, Institute of Aeronautical Engineering, Hyderabad, India
[3]Department of Computer Science Engineering, Hyderabad Institute of Technology and Management, Telangana, India
[4]Department of Computer Science & Engineering, Silicon Institute of Technology, Bhubaneswar, Odisha, India, 751024
[5]Department of Computer Science and Engineering, Builders Engineering College, Kangeyam Tirupur, Tamil Nadu, India, 638108
[6]PS Consulting and Solutions
[7]Department of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation, Vaddeswaram, Guntur District, Andhra Pradesh, India, 522302

Abstract. Cloud Computing (CC) offers abundant resources and diverse services for running a wide range of consumer applications, although it faces specific issues that need attention. Cloud customers aim to choose the most suitable resource that fulfills the requirements of consumers at a fair cost and within an acceptable timeframe; however, at times, they wind up paying more for a shorter duration. Many advanced algorithms focus on optimizing a single variable individually. Hence, an Optimized Resource Allocation in Cloud Computing (ORA-CC) Model is required to achieve equilibrium between opposing aims in Cloud Computing. The ORA-CC study aims to create a task processing structure with the decision-making ability to choose the best resource in real-time for handling diverse and complicated uses on Virtual Computers (VC). It will utilize a Modified Particle Swarm Optimization (MoPSO) method to meet a deadline set by the user. The fitness value is calculated by combining a base value with the enhanced estimation of resources based on the ORA-CC algorithm to create a robust arrangement. The ORA-CC technique's effectiveness is evaluated by contrasting it with a few current multi-objective restrictions applied to machine scheduling strategies utilizing the Cloudsim simulation. The comparison demonstrates that the suggested ORA-CC strategy offers more efficient resource allocation than other techniques.

1 Introduction

* Corresponding author: gsrinivasulu1678@vardhaman.org

© The Authors, published by EDP Sciences. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (https://creativecommons.org/licenses/by/4.0/).

Advancements in computer networks, devices, and Internet speeds have prompted numerous companies and users to utilize cloud services. Data centers have become more efficient due to sharing of resources, application virtualization, and advanced scheduling techniques. This has prompted users to utilize cloud services to fulfill their requirements. Users require powerful processors and high-speed networks to establish CC centers across various global locations. An essential issue in this field is energy [1, 2].

In the past decade, a focus has been on improving methods for regulating power in cloud servers. Numerous investigators have developed various algorithms to lower expenditures by decreasing energy utilization and minimizing greenhouse gas release. Testing has been conducted to enhance the performance of multiple algorithms by influencing either the hardware or the software. Virtualization techniques allow the creation of multiple virtual machines on a single physical machine, reducing the requirement for additional hardware and enhancing efficiency [3]. CC utilizes virtualization technology tailored to user requirements to supply resources, decreasing energy usage. Utilizing metaheuristic techniques is a popular method for decreasing energy use. This method may draw attention because of its straightforward implementation.

Cloud service suppliers are inclined towards CC due to the advantages they gain from serving clients and customers, like entities and financial institutions, by decreasing or removing the infrastructure operating expenses. Consumers require assurance from service providers regarding the security of highly confidential enterprise applications operating in the cloud. A warranty is typically established between the client and the supplier via a service-level contract. Virtualization has recently become popular for enhancing the efficiency of cloud networks [4].

VC can be migrated between hosts to improve the efficiency of applications utilized by users without causing service disruptions. The quantity of assets assigned to a client can change periodically [5, 6]. CC users are charged based on their usage and services received. Their data is transferred to the cloud via the Internet, allowing them to access it globally anytime. Users do not require high-capacity hardware components for handling and storing data, as sophisticated servers in the cloud service provider's infrastructure manage all CC functions and caching. CC solutions are available in three forms: SaaS (Software as a Service), PaaS (Platform as a Service), and IaaS (Infrastructure as a Service).

Utilizing virtualization-based networks and saving information in the cloud can help distribute network load. However, it may disrupt the equilibrium, similar to big data's phase mapping and radius. Utilizing virtualization and load-balancing methods can be highly beneficial. This paper introduces the MoPSO algorithm focusing on scheduling and resource allocation for ORA-CC.

2 Related works

Selvapandian and Santosh (2022) created a hybrid optimized model for allocating resources in multi-cloud environments by combining BAT and PSO algorithms. Their model demonstrated a remarkable average resource utilization of 87% on various cloud platforms [7].

Ramasamy and Thalavai Pillai (2020) introduced an efficient HPSO-MGA optimization algorithm for adapting resource allocation in cloud environments. Their method led to a notable 15% decrease in resource wastage in dynamic cloud environments [8]. Shao, Fu, and Wang (2023) introduced a novel approach that combines genetic algorithm and PSO to schedule data-intensive tasks in diverse cloud computing environments. Their approach resulted in a 20% reduction in task completion time and a 25% enhancement in overall system throughput [9].

Hafsi, Gharsellaoui, and Bouamama (2022) proposed a genetically modified multi-objective particle swarm optimization method for scheduling high-performance computing workflows. Their method showed a significant 30% decrease in the time it takes to complete tasks and a 10% improvement in the use of resources [10].

Alfakih, Hassan, and Al-Razgan (2021) studied multi-objective accelerated PSO combined with dynamic programming for allocating resources in mobile edge computing. A significant 25% enhancement in energy efficiency and a 15% boost in overall system performance were documented [11]. Mirmohseni, Javadpour, and Tang (2021) introduced LBPSGORA, a method combining particle swarm optimization and genetic algorithms to enhance resource distribution and reduce energy usage in cloud networks. Their approach led to a notable 20% enhancement in load distribution and a 15% decrease in energy usage [12].

Pirozmand et al. (2023) introduced an enhanced PSO for task scheduling in cloud computing. Their method resulted in a significant 30% reduction in task completion time and a 20% improvement in resource utilization, demonstrating its effectiveness in enhancing cloud resource management [13]. Thus, an ORA-CC model is required to achieve equilibrium between opposing aims in CC. The ORA-CC study aims to create a task processing structure with a decision-making ability to choose the best resource in real-time for handling diverse and complicated uses of VC.

3 Proposed ORA-CC model

The use of Modified Particle Swarm Optimization (MoPSO) to enhance resource allocation in CC environments is examined in this paper. Cloud service providers can maximize resource utilization and performance through dynamic resource allocation, including processing power and storage. The study examines how adding features or constraints specific to CC environments to MoPSO improves standard PSO algorithms. The research aims to demonstrate the effectiveness of MoPSO in achieving better resource allocation strategies that lead to enhanced overall performance, scalability, and cost-effectiveness in CC infrastructures through extensive experimentation and performance evaluations.

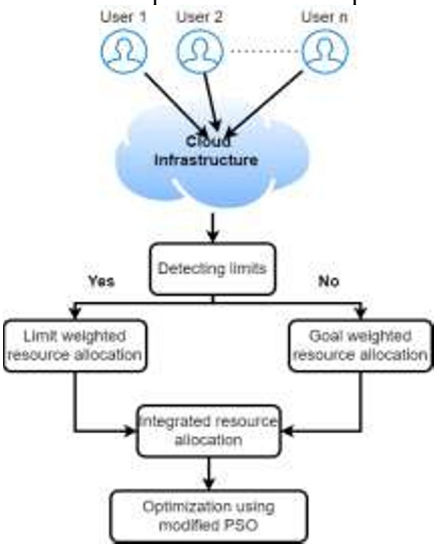

Fig. 1. Architecture of the proposed ORA-CC model

A cloud server datacentre is a large collection of servers connected to the Internet. The job scheduler must coordinate job completions on the cloud. An efficient scheduler minimizes utilizing resources such as energy, bandwidth, memory, and time to complete user tasks. Most services the requester utilizes within the industrial IT setting are limited to IaaS. The quantity of job submissions to the IaaS is experiencing a significant increase. The clients' request involves utilizing virtualization technology to operate in the cloud. The ORA-CC method is developed to select the most efficient VC for managing client projects within various limits. Load balancing and resource allocation are closely interconnected in the cloud environment. The ORA-CC procedure is designed to address the challenge of managing multiple cloud resources with specific requirements and allocation. The ORA-CC Technique efficiently schedules client solicitation tasks across VC by adjusting the client solicitation level using two load balancing algorithms. The ORA-CC technique enhances resource allocation planning productivity by efficiently distributing tasks among VC in the cloud with increased effectiveness and minimal time.

Fig. 1 illustrates the architecture of the ORA-CC strategy. It illustrates that numerous consumer inquiries are transmitted to the cloud server. The client acquisition rates received by the cloud server are inconsistent. Occasionally, many requests are received quickly, leading to a bottleneck and affecting the load-balancing performance. The ORA-CC technique is suggested for efficiently organizing client assignments in critical and regular workloads on the cloud server. When the cloud server receives multiple client requests, the ORA-CC uses a limit detector to determine if the workload state is normal or if a limit is applied. Based on the limit detector result, the ORA-CC selects the appropriate limit adjustment solution for a given state. The scheduler will use the goal-based weighted resource allocation strategy if the query is not based on necessities. Subsequently, the selected load balancing algorithm assigns the task to an optimal VC based on resource availability using fitness value. The ORA-CC enhances scheduling efficiency by reducing cloud processing time compared to previous methods.

3.1 Modified Particle Swarm Optimization (MoPSO)

The MoPSO method is an enhanced version of the classic PSO approach designed to increase its efficiency in addressing optimization issues. This investigation examines the fundamental concepts of MoPSO, explaining its position update equation and velocity update equation. It explains how characteristics like inertia weight, acceleration coefficients, and random values influence the search process. It also explores possible changes and improvements that might be added to MoPSO to boost its performance even further. Utilizing MoPSO in cloud computing shows great promise for enhancing resource allocation, job scheduling, and energy management. Cloud providers may use MoPSO to assign resources dynamically, optimize job scheduling, and monitor energy usage to improve efficiency and cost-effectiveness. MoPSO is capable of adjusting to changing and uncertain cloud environments, guaranteeing strong and expandable resource allocation plans.

The MoPSO with hybrid GA-PSO manages hyperparameter tuning to avoid the time-consuming process of manual trial-and-error tuning. Hybrid GA-PSO integrates the concepts of Genetic Algorithm (GA) and PSO to use their complimentary capabilities in exploring and exploiting solutions. The hybrid technique involves running GA and PSO algorithms simultaneously or in sequence, with GA focusing on global exploration and PSO on local exploitation.

In each iteration, the particle uses its current individual best (P_best) and current global best (G_best) values to determine its velocity for the next instant, after which it shifts its location. Assuming there are j particles in the vicinity, the $(M+1)$ movement vector apprising formula and $(M+1)$ position vector apprising formula for the i^{th} particle is as follows:

$$vel_i^{M+1} = W * vel_i^M + MC_1A_1(P_{-besti}^M - P_i^M) + MC_2A_2(P_{-besti}^M - P_i^M) \quad (1)$$

$$P_i^{M+1} = P_i^M + vel_i^{M+1} \quad (2)$$

where A_1 and A_2 are arbitrary integers between 0 and 1, and MC_1 and MC_2 are the individual learning and population training constraints, which, respectively, show regional and global optimization capabilities. The weight (W) represents the impact of the previous iteration's velocity on the current iteration's velocity. A higher weight indicates a higher possibility for global optimization, whereas a lower weight indicates the opposite.

Hyperparameter Tuning Process in MoPSO

Step 1: Begin by creating a population of particles that each represent distinct hyperparameter setups.

Step 2: Assess the effectiveness of each particle by analyzing its performance on a validation set using a predetermined measure like as accuracy or loss.

Step 3: Utilize PSO to adjust the velocity and location of particles in order to navigate the hyperparameter space.

Step 4: Utilize genetic algorithm operators (selection, crossover, mutation) to develop the population and preserve variety.

Step 5: Continue the procedure for many generations or until the convergence conditions are satisfied.

Step 6: Choose the hyperparameter setup that demonstrates the highest performance on the validation set as the ultimate optimized configuration.

Generally, GA is in charge of preserving variety and navigating the search space via selection, crossover, and mutation processes. PSO is used to enhance solutions identified by GA by the exploration of the local search area around potential solutions.

4 Analysis and discussion of results

The efficacy of the suggested resource allocation framework is assessed across a range of workloads, revealing that the modified PSO algorithm implemented in the controller node automatically assigns and releases resources in response to user demand, with the assistance of other components. Experiments were carried out on various artificial data sets utilizing the Cloudsim tool. A comparison was made between the efficiency of the proposed algorithm and that of other cutting-edge scheduling algorithms across several critical parameters, including implementation cost, job acceptance ratio, computing time, and throughput.

This section provides an analysis and comparison of the simulation outcomes obtained from the developed MoPSO algorithm concerning benchmark resource allocation algorithms and other standards, including the Enhanced Max-Min algorithm for Load Balancing (EMMLB) [14], Artificial Bee Colony (ABC) [15], BAT algorithm [16], and PSO [17]. ABC, BAT, and PSO are algorithms that rely on meta-heuristic techniques, while EMMLB utilizes a heuristic model.

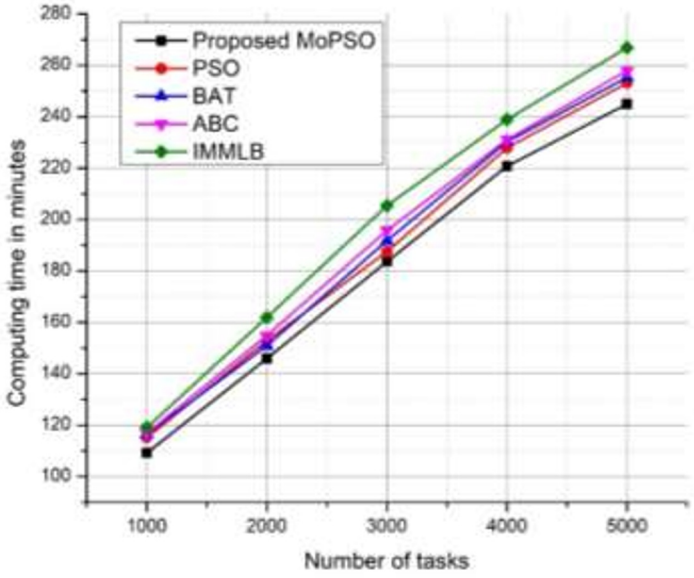

Fig. 2. Computing time (in minutes) for varying tasks with different optimization algorithms in the ORA-CC model

The computation time (measured in minutes) for various tasks executed using distinct optimization algorithms within the ORA-CC model is detailed in Fig. 2. The number of VCs is kept at 500. Generally, the computation duration for each algorithm increases as the number of tasks increases. Across all task counts, the proposed MoPSO algorithm consistently outperforms alternative algorithms, exhibiting the shortest computation times. As an illustration, the proposed MoPSO algorithm completes one thousand tasks in 109.21 minutes, whereas the respective execution times for PSO, BAT, ABC, and IMMLB are 115.14, 116.47, 116.74, and 119.08 minutes. The sustained pattern of this trend with the escalation of task count proves that the proposed MoPSO algorithm outperforms conventional optimization techniques within the ORA-CC model.

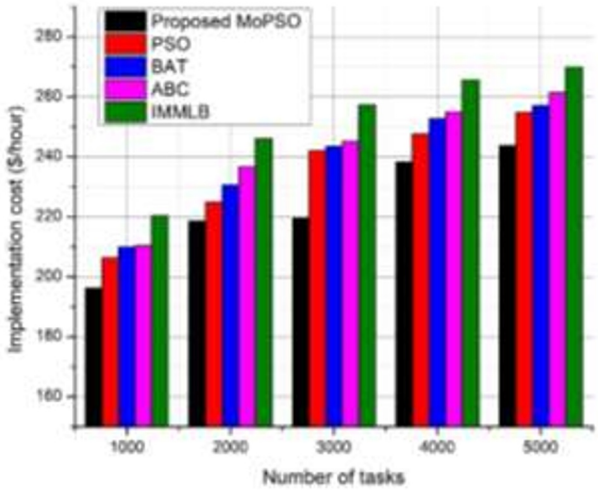

Fig. 3. Implementation cost ($/hour) for the varying tasks with different optimization algorithms in the ORA-CC model

The implementation cost per hour for an assortment of tasks utilizing distinct optimization algorithms in the ORA-CC model is detailed in Fig. 3. In comparison to alternative algorithms; it is indisputable that the proposed MoPSO algorithm consistently provides the most cost-effective implementations across all task quantities. The proposed MoPSO algorithm entails an implementation cost of $196.2 per hour for 1000 tasks. In contrast, the costs incurred by PSO, BAT, ABC, and IMMLB are $206.4 per hour for PSO, $209.9 per hour for ABC, and $210.3 per hour for IMMLB and PSO. In a similar vein, the cost-effectiveness of the proposed MoPSO algorithm in optimizing resource allocation in the ORA-CC model is underscored by the fact that it retains its cost advantage over other algorithms even as the number of tasks increases.

5 Conclusion

This paper presented an Optimized Resource Allocation in Cloud Computing (ORA-CC) Model for CC to attain equilibrium between competing objectives. The ORA-CC research initiative aims to develop a task-processing framework to identify the most optimal resource in real time for managing a wide range of complex applications on VC. A MoPSO technique will be implemented to satisfy a user-specified deadline. The fitness value is computed to generate a robust arrangement by combining a base value with the improved estimation of resources generated by the ORA-CC algorithm. The effectiveness of the ORA-CC method is assessed through a comparison with several contemporary multi-objective constraints implemented in machine scheduling strategies via the Cloudsim simulation. The proposed MoPSO algorithm completes one thousand tasks in 109.21 minutes, whereas the respective execution times for PSO, BAT, ABC, and IMMLB are 115.14, 116.47, 116.74, and 119.08 minutes. The proposed MoPSO algorithm entails an implementation cost of $196.2 per hour for 1000 tasks. In contrast, the costs incurred by PSO, BAT, ABC, and IMMLB are $206.4 per hour for PSO, $209.9 per hour for ABC, and $210.3 per hour for IMMLB and PSO. The comparison illustrates that the ORA-CC strategy with MoPSO, as proposed, provides a more effective allocation of resources in comparison to alternative techniques.

References

1. S.M. Mirmohseni, C. Tang, A. Javadpour. *Using Markov learning utilization model for resource allocation in cloud of thing network*. Wirel. Pers. Commun., **115**, 653-677, (2020)
2. A. Katal, S. Dahiya, T. Choudhury. *Energy efficiency in cloud computing data centers: a survey on software technologies*. Cluster Comput., **26**, 3, 1845-1875, (2023)
3. V.K. Sharma, A. Singh, K.R. Jaya, A.K. Bairwa, D.K. Srivastava. *Introduction to virtualization in cloud computing*. In Machine Learning and Optimization Models for Optimization in Cloud, Chapman and Hall/CRC, 1-14, (2022)
4. H. Shukur, S. Zeebaree, R. Zebari, D. Zeebaree, O. Ahmed, A. Salih. *Cloud computing virtualization of resources allocation for distributed systems*. J. Appl. Res. Technol. Tren., **1**, 3, 98-105, (2020)
5. R. Zolfaghari, A. Sahafi, A.M. Rahmani, R. Rezaei. *Application of virtual machine consolidation in cloud computing systems*. Sustain. Comput.: Inform. Syst., **30**, (2021)
6. R. Zolfaghari, A.M. Rahmani. *Virtual machine consolidation in cloud computing systems: Challenges and future trends*. Wirel. Pers. Commun., **115**, 3, 2289-2326, 2020.

7. D. Selvapandian, R. Santosh. *A Hybrid Optimized Resource Allocation Model for Multi-Cloud Environment Using Bat and Particle Swarm Optimization Algorithms*. Comput. Assist. Methods Eng. Sci., **29**, 1–2, 87-103, (2022)
8. V. Ramasamy, S. Thalavai Pillai. *An effective HPSO-MGA optimization algorithm for dynamic resource allocation in cloud environment*. Cluster Comput., **23**, 1711-1724, (2020)
9. K. Shao, H. Fu, B. Wang. *An efficient combination of genetic algorithm and particle swarm optimization for scheduling data-intensive tasks in heterogeneous cloud computing*. Electronics, **12**, 16, (2023)
10. H. Hafsi, H. Gharsellaoui, S. Bouamama. *Genetically-modified Multi-objective Particle Swarm Optimization approach for high-performance computing workflow scheduling*. Appl. Soft Comput., **122**, (2022)
11. T. Alfakih, M.M. Hassan, M. Al-Razgan. *Multi-objective accelerated particle swarm optimization with dynamic programing technique for resource allocation in mobile edge computing*. IEEE Access, **9**, 167503-167520, (2021)
12. S.M. Mirmohseni, A. Javadpour, C. Tang. *LBPSGORA: create load balancing with particle swarm genetic optimization algorithm to improve resource allocation and energy consumption in clouds networks*. Math. Probl. Eng., 1-15, (2021)
13. P. Pirozmand, H. Jalalinejad, A.A.R. Hosseinabadi, S. Mirkamali, Y. Li. *An improved particle swarm optimization algorithm for task scheduling in cloud computing*. J. Ambient Intell. Humaniz. Comput., **14**, 4, 4313-4327, (2023)
14. T.C. Hung, L.N. Hieu, P.T. Hy, N.X. Phi. *MMSIA: improved max-min scheduling algorithm for load balancing on cloud computing*. In Proceedings of the 3rd International Conference on Machine Learning and Soft Computing, 60-64, (2019)
15. D.I.J. Jacob, D.P.E. Darney. *Artificial bee colony optimization algorithm for enhancing routing in wireless networks*. J. Artif. Intell. Cap Netw., **3**, 1, 62-71, (2021)
16. S.U. Umar, T.A. Rashid. *Critical analysis: bat algorithm-based investigation and application on several domains*. World J. Eng., **18**, 4, 606-620, (2021)
17. M.I. Alghamdi. *Optimization of Load Balancing and Task Scheduling in Cloud Computing Environments Using Artificial Neural Networks-Based Binary Particle Swarm Optimization (BPSO)*. Sustainability, **14**, 19, 11982, (2022)