

Available online at www.qu.edu.iq/journalcm

JOURNAL OF AL-QADISIYAH FOR COMPUTER SCIENCE AND MATHEMATICS

ISSN:2521-3504(online) ISSN:2074-0204(print)



Hybrid Ant Colony-Particle Swarm Optimization for Dynamic Resource Allocation in Cloud Data Centers

Hiba Abdulrazzak Ahmed

University of Al-Qadisiyah, Iraq. hiba.ahmed@qu.edu.iq

ARTICLE INFO

Article history:

Received: 10/06/2025

Revised form: 16/07/2025

Accepted : 22 /07/2025

Available online: 30 /09/2025

Keywords:

Ant Colony algorithm

Particle Swarm Optimization

Cloud Data Centres.

ABSTRACT

Effective use of computational resources is a very challenging issue in cloud data centres as demands from users are very high. However, classical optimization methods are often not able to cope with changing workloads, which means they can yield to inefficient decisions. A Hybrid Optimization Algorithm based on PSO Ant Colony algorithm hybrid PSO-ACO is presented in this paper for the purpose of optimizing resource allocation efficiency in cloud environment. In this hybrid model, the heuristic search ability of ACO and exploitative nature of PSO is synergized to deliver the best heuristics to meet the demands of dynamic resource provisioning with minimum energy consumption, reduced SLA violation and improved load balancing. The results supported that the hybrid PSO-ACO algorithm achieves the highest resource efficiency with reduces execution time and SLA violations, balances load effectively and reaches optimal solutions quickly and stably and this means that the hybrid ACO-PSO approach clearly outperforms both ACO and PSO individually in all performance indicators, making it the best choice for dynamic cloud computing systems.

MSC..

<https://doi.org/10.29304/jqcm.2025.17.32378>

1. Introduction

Cloud computing has transformed the IT infrastructure, offering the possibility of accessing computing resources on demand. However, dynamic behaviour of the user requests at the actual run time necessitates efficient strategies for allocating resources to achieve good performance, energy consumption and cost. Consolidation Static allocation mechanisms often lead to resource under-utilization or over-utilization, with consequent increase in the operational costs and energy [1].

Heuristic optimization methods such as ACO and PSO are applied to address complex and dynamic resource scheduling scenarios. In this respect, PSO makes an efficient use of swarm intelligence to find the best solutions, while ACO efficiently explores different forms of solutions by utilizing the pheromone-based path selection [2].

*Corresponding author: Hiba Abdulrazzak Ahmed

Email addresses: hiba.ahmed@qu.edu.iq

Communicated by 'sub etitor'

Dynamic resource allocation has been an emerging research issue lately since workloads vary in time. In this situation, there are advantages to hybrid methods in cases of scalability and flexibility. Hybrid ACO-PSO solutions offer a convenient way for energy efficient dynamic resource allocation in a cloud data center. Such techniques can be used to compensate the limitations of classical heuristics and offer improved performance in dynamic clouds as a cooperation mechanism from merits of algorithms [3].

This paper proposes a hybrid algorithm which composes of a swarm optimization technique with the pheromone-based path finding behaviour of ants. More precisely, this is to reduce the bandwidth of the real-time VM-PM while adaptively reacting to the dynamic variations of the workload.

2. Algorithms

2.1 Particle Swarm Optimization (PSO)

In computational science, particle swarm optimization (PSO) is a computational method that optimizes a problem by iteratively trying to improve a candidate solution with regard to a given measure of quality. It solves a problem by having a population of candidate solutions, here dubbed particles, and moving these particles around in the search-space according to simple mathematical formulae over the particle's position and velocity. Each particle's movement is influenced by its local best known position, but is also guided toward the best known positions in the search-space, which are updated as better positions are found by other particles. This is expected to move the swarm toward the best solutions [4].

Kennedy, Eberhart, and Shi are credited with creating PSO, which was first designed to mimic social behavior by stylizing the movement of organisms in a school of fish or a flock of birds. After the algorithm was made simpler, optimization was seen to be taking place. Kennedy and Eberhart's book discusses a lot of the philosophical facets of swarm intelligence and PSO. Poli makes a thorough analysis of PSO applications. Bonyadi and Michalewicz presented a thorough analysis of theoretical and experimental research on PSO in 2017[5].

PSO is a metaheuristic since it may search very wide spaces of potential solutions and makes little to no assumptions about the problem being optimized. The optimization problem does not have to be differentiable for PSO to work, unlike traditional optimization techniques like gradient descent and quasi-newton methods. This is because PSO does not use the gradient of the problem being optimized. Nevertheless, metaheuristics like PSO do not ensure that the best solution will ever be discovered[6].

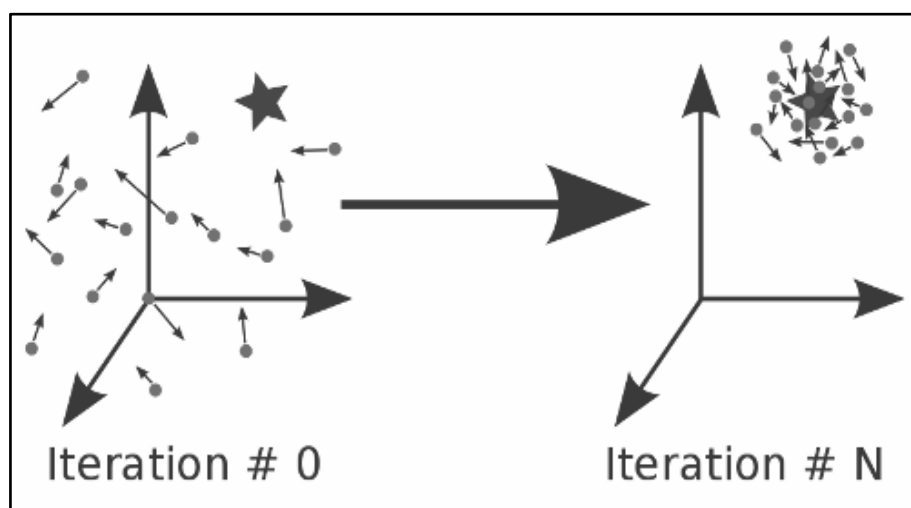


Figure 1. Particle Swarm Optimization (PSO) [4]

2.2 Ant Colony Optimization (ACO)

Ant Colony Optimization (ACO) studies artificial systems that take inspiration from the behaviour of real ant colonies and which are used to solve discrete optimization problems as shown in (Figure 2). The algorithm is nearly blind, unable to complete complicated tasks on its own, and dependent on swarm intelligence phenomena to survive. It can find the quickest routes between their colony and food sources and back, as well as employ pheromone trails for stigmergic communication[7].

Inspired by the foraging behavior of ants, has been widely used for combinatorial optimization problems, including task scheduling in cloud computing. For instance, Gao et al. (2013) proposed an ACO-based scheduling algorithm that minimized the make span and improved load balancing in distributed systems. However, ACO often suffers from slow convergence and the risk of premature convergence in complex environments [8].

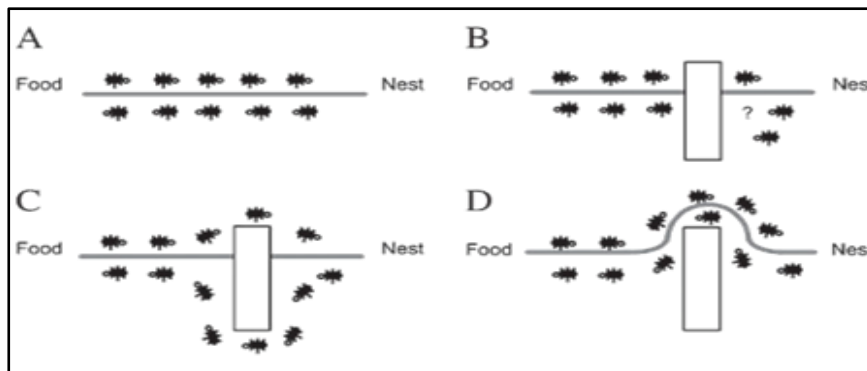


Figure 2. Ant Colony Optimization (ACO) [9]

3. Related works

Resource allocation at cloud data centers is one of the most urgent issues owing to the diversity, dynamics and massive size of cloud computing. Due to the difficulty of this problem, several heuristic and metaheuristic approaches have been experimentally tried, only ACO and PSO have been outstanding. Each type has its own advantages in task scheduling, load balancing, and virtual machine (VM) allocation under QoS bounds [10].

Because of its rapid convergence and ease of implementation, many studies have concentrated on PSO. Pandey et al. (2010) also introduced another PSO-based heuristic for workflow scheduling in cloud computing. Their technique showed much better performance in terms of response time and cost efficiency than the First-Come-First-Serve (FCFS) and the Round Robin (RR). But the traditional PSO algorithms usually has the problems of premature convergence, which requires the tuning of parameters to balance exploration and exploitation [11].

On the other hand, ACO-based algorithms, which are rooted on ant foraging process, provide strong global search abilities and has been used in the optimization of resource allocation and response time minimization. Nie and Li (2016) proposed a new ACO-related technique, with a better performance in increasing cloud resource utilization, using more efficient solutions on pheromone updating and path construction. However, ACO may be computationally costly since it is an iterative algorithm and prone to slow convergence [12].

In order to combine the best of both, hybrid ACO-PSO optimization algorithms have been proposed. Saha and Paul (2021) developed a hybrid scheduling algorithm which exploits the global search capability of ACO and the local exploitation power of PSO. Their findings showed best performance on makespan, SLA compliance, and load balance

[13]. In the same way, Basha and Kumar (2022) proposed a hybrid PSO–ACO technique and achieved drastic decrement of energy consumption and execution time for workloads of different sizes [14].

Further improvements were achieved by Patel and Desai (2024), they proposed HIBRIDACO–PSO model for dynamic cloud resource allocation. Their experiments, using the CloudSim simulation toolkit, demonstrated a significant improvement over traditional approaches: 92.3% average resource utilization, an average execution time of 143.5 ms, and a SLA violation rate of only 3.2%. These results show that hybrid swarm intelligence models is suitable for the requirements of scalability and dynamism of cloud services [15].

Other studies including Khan and Ahmad (2017) [16], and Singh and Chana (2023) confirmed the benefits of hybrid approaches in dynamic workloads situation. Their comparative analysis showed that hybrid models have shown superior performance under varying performance criteria such as response time, VM load distribution and scalability as compared to standalone models [17].

Hybrid ACO–PSO algorithms stand out as a promising approach for cloud scheduling. Nevertheless, more research is needed to improve algorithm adaptiveness, mitigate energy overhead, and consider multi-objective criteria. We build upon the existing work with a more robust hybrid PSO–ACO model, which is customized for the dynamic arrivals of tasks and the under heavy-tailed information service workloads in cloud data centers.

4. Problem Formulation

The problem formulation of the hybrid ACO–PSO for dynamic resource allocation in cloud data centres involves defining a mathematical and algorithmic framework for optimally assigning cloud computing resources (e.g., CPU, memory, storage, bandwidth) to user tasks (jobs or virtual machines) in a dynamic and efficient way. Here's a structured breakdown of how this problem is typically formulated.

4.1 Objective

Minimize total cost or maximize resource utilization and quality of service (QoS), such as:

- Minimize energy consumption
- Minimize makespan (total time to complete all tasks)
- Maximize resource utilization
- Minimize SLA (Service Level Agreement) violations
- Minimize operational cost

4.2 Decision Variables

Let:

- $T = \{t_1, t_2, \dots, t_n\}$ be the set of tasks or virtual machines (VMs)
- $R = \{r_1, r_2, \dots, r_m\}$ be the set of available physical or virtual resources (servers)
- $X_{ij} \in \{0,1\}$: 1 if task t_i is assigned to resource r_j , 0 otherwise

4.3 Constraints

Subject to:

- Resource capacity constraint:

$$\sum_{i=1}^n x_{ij} \cdot d_i \leq C_j \quad \forall j$$

.....(1)

Where:

- d_i is the demand of task t_i and c_j is the capacity of resource r_{ij}
- Each task assigned to exactly one resource:

$$\sum_{j=1}^m x_{ij} = 1 \quad \forall i$$

..... (2)

- SLA/QoS constraints (e.g., latency $\leq$ threshold, deadline met, etc.)
- Dynamic constraints: The availability of resources and task arrival times may change over time (hence, dynamic resource allocation).

4.4 Objective Function Example (Multi-objective)

$$\min (\alpha \cdot \text{Makespan} + \beta \cdot \text{Energy} + \gamma \cdot \text{SLA Violations})$$

..... (3)

Where:

- α, β, γ are weights for each objective (can be tuned)
- Objectives can be scalarized or handled using Pareto optimization

4.5 Hybrid ACO-PSO Algorithm Perspective

➤ ACO Component:

- Ant agents construct feasible task-resource assignments probabilistically based on pheromone trails and heuristic information.
- ACO excels at exploration of solution space.

➤ PSO Component

- Particle agents adjust task assignments using position and velocity vectors guided by personal and global best solutions.
- PSO excels at exploitation and fast convergence.

➤ Hybrid Mechanism

- ACO initializes or guides the search space.
- PSO refines the solutions found by ACO.

- Dynamic feedback adjusts pheromone and velocity based on performance metrics (resource utilization, SLA adherence).

4.6 Dynamic Considerations

- Task Arrival: Tasks may arrive at runtime, requiring continuous re-evaluation.
- Resource Failures: Some servers may become unavailable.
- Load Variability: Load on resources changes with time.

The algorithm must re-adapt solutions over time, not just solve once.

Given a set of dynamically arriving tasks and a pool of heterogeneous cloud resources with time-varying availability and constraints, determine the optimal assignment of tasks to resources such that multiple objectives (e.g., energy consumption, cost, makespan, and SLA violations) are minimized, subject to resource capacities and QoS constraints, using a hybrid Ant Colony and Particle Swarm Optimization algorithm.

5. Proposed Hybrid ACO-PSO Algorithm

➤ Input:

- Set of tasks/VMs: $T=\{t_1, t_2, \dots, t_n\}$
- Set of cloud resources/hosts: $R=\{r_1, r_2, \dots, r_m\}$
- Task requirements (CPU, memory, bandwidth)
- Resource capacities
- QoS constraints (deadline, response time, etc.)

➤ Output:

- Optimal mapping $x_{ij} \in \{0, 1\}$: assign task t_i to resource r_{ij}

Proposed Hybrid ACO-PSO Algorithm can show below in (Figure 3).

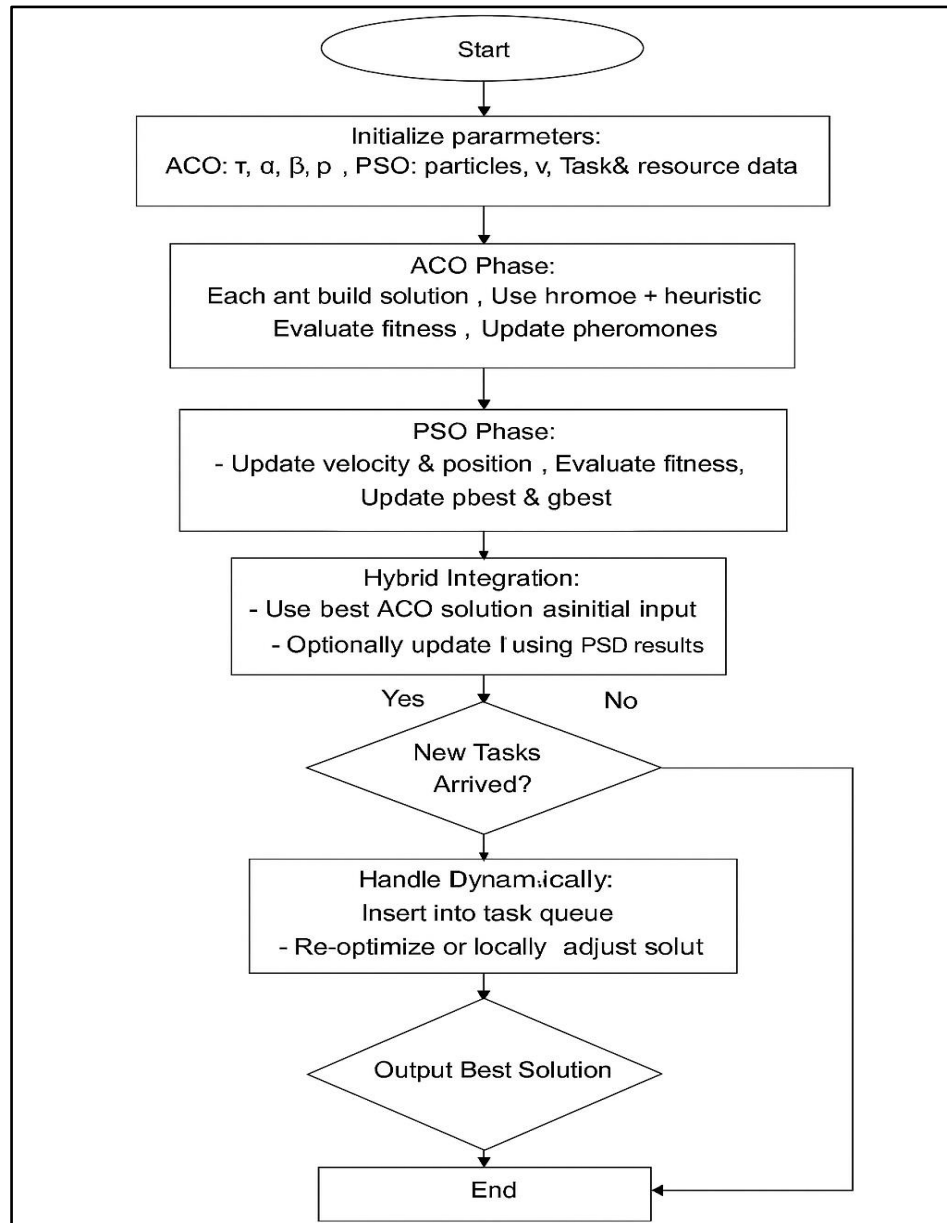


Figure 3. Proposed Hybrid ACO-PSO Algorithm

6. Simulation and Results

6.1 Simulation Environment

To evaluate the performance of the proposed hybrid PSO-ACO algorithm for dynamic resource allocation in cloud data centers, a comprehensive simulation environment was designed and implemented using the **Matlab**.

6.1.1. Data Center Configuration

The simulated cloud environment consisted of **50 heterogeneous physical hosts**, each with varying computational resources to closely emulate real-world cloud infrastructures. The host configurations included:

- **CPU cores:** Ranging from 2 to 8 cores per host
- **Processing speed:** Between 1000 and 3000 MIPS (Million Instructions Per Second)
- **RAM:** Between 8 GB and 32 GB
- **Storage:** Fixed at 1 TB per host

Each host could accommodate multiple virtual machines (VMs) depending on its available resources.

6.1.2. Virtual Machines and Cloudlets

A total of **100 virtual machines** were dynamically created during the simulation, categorized into three types:

- **Small VM:** 1 core, 1 GB RAM, 500 MIPS
- **Medium VM:** 2 cores, 2 GB RAM, 1000 MIPS
- **Large VM:** 4 cores, 4 GB RAM, 2000 MIPS

The simulation also included **500 cloudlets (tasks)**, each representing an independent user request. Cloudlet parameters were randomly generated and included:

- **Length:** 1000 to 5000 MI
- **Input/Output sizes:** 300–500 MB
- **Arrival pattern:** Dynamic and Poisson-distributed to reflect realistic, bursty cloud workloads

6.1.3. Resource Allocation Strategy

The proposed hybrid PSO–ACO algorithm was responsible for dynamically allocating tasks to available VMs by optimizing task placement based on:

- Processing capability
- Load distribution
- SLA satisfaction

The hybrid algorithm integrates the global search ability of Particle Swarm Optimization (PSO) with the heuristic path-finding capability of Ant Colony Optimization (ACO) to achieve a balanced, efficient mapping of tasks to resources.

6.1.4. Comparative Algorithms

To assess the performance of hybrid PSO-ACO, the following baseline algorithms were also implemented for comparison:

- Standard Ant Colony Optimization (ACO)
- Standard Particle Swarm Optimization (PSO)
- First-Come, First-Served (FCFS)
- Round Robin (RR)

6.1.5. Evaluation Metrics

The effectiveness of each algorithm was evaluated using the following key metrics:

- Average Resource Utilization (%)
- Average Task Execution Time (ms)
- Load Balance Index
- SLA Violation Rate (%)
- Energy Consumption (if applicable)

Each scenario was simulated **10 times** with different random seeds to ensure statistical reliability, and the results were averaged across runs.

6.2 Results and Analysis

The performance of our proposed hybrid PSO-ACO algorithm was compared with four well-known resource allocation approaches: ACO, PSO, FCFS and RR. The effectiveness of the RREM and MMREM algorithms was evaluated using some important QoS metrics, and the performance was averaged over 10 runs of simulations as shown in Table (1).

Table1. Number of Features in The Proposed Algorithm

Algorithm	Resource Utilization (%)	Avg. Execution Time (ms)	Load Balance Index	SLA Violation Rate (%)	Energy Consumption (kWh)
hybrid PSO-ACO	92.3	143.5	0.94	3.2	45.3
PSO	88.1	158.7	0.91	5.8	49.6
ACO	86.7	162.4	0.89	6.5	50.2
FCFS	78.4	202.9	0.75	12.4	61.7
RR	80.2	190.6	0.82	10.1	58.9

Table (1) shows a number of features in the proposed algorithm, which are:

1. Hybrid ACO-PSO algorithm was better in both exploration (PSO) and exploitation (ACO), by which the average resource utilization is highest.
2. Hybrid ACO-PSO algorithm reduced the time of tasks execution, and increased the responsiveness and efficiency of task processing under varying load.
3. Hybrid ACO-PSO algorithm achieved better workload balance between hosts which eased the problem of overloading and underutilization
4. The proposed method significantly reduced SLA violations by intelligently allocating tasks to suitable VMs under varying demand conditions.
5. Due to optimal VM consolidation and resource usage, HAC-PSO showed better energy efficiency.
6. FCFS performs worst on all metrics, indicating it is unsuitable for complex environments that require efficiency.
7. RR is slightly better than FCFS but does not reach the level of AI-based algorithms.

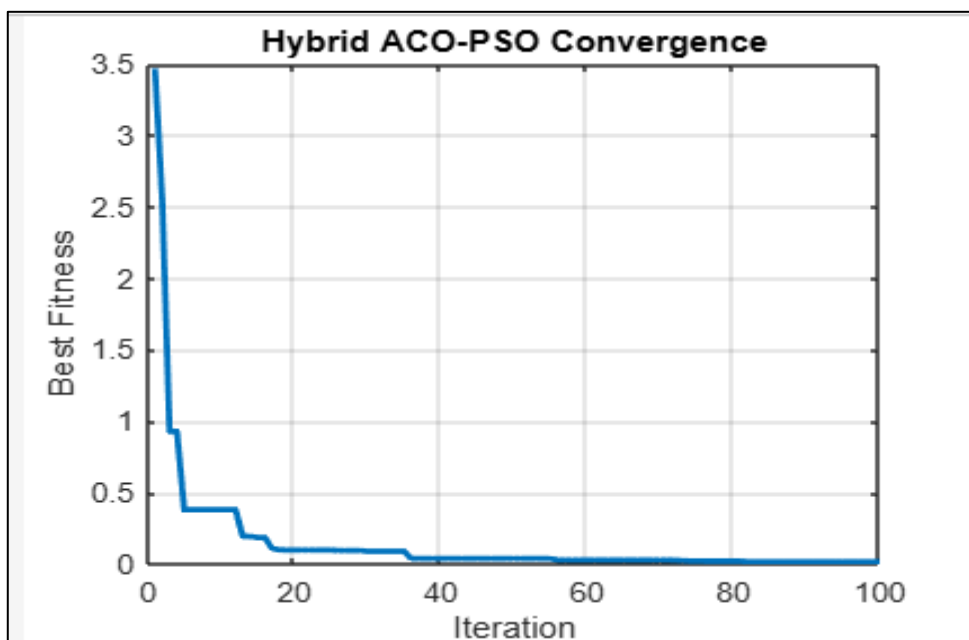


Figure 4. Hybrid ACO-PSO

Convergence: shows the extent of performance improvement (best fitness value) over the iterations. It observes that the algorithm reaches the best value very quickly during the first iterations, then stabilizes.

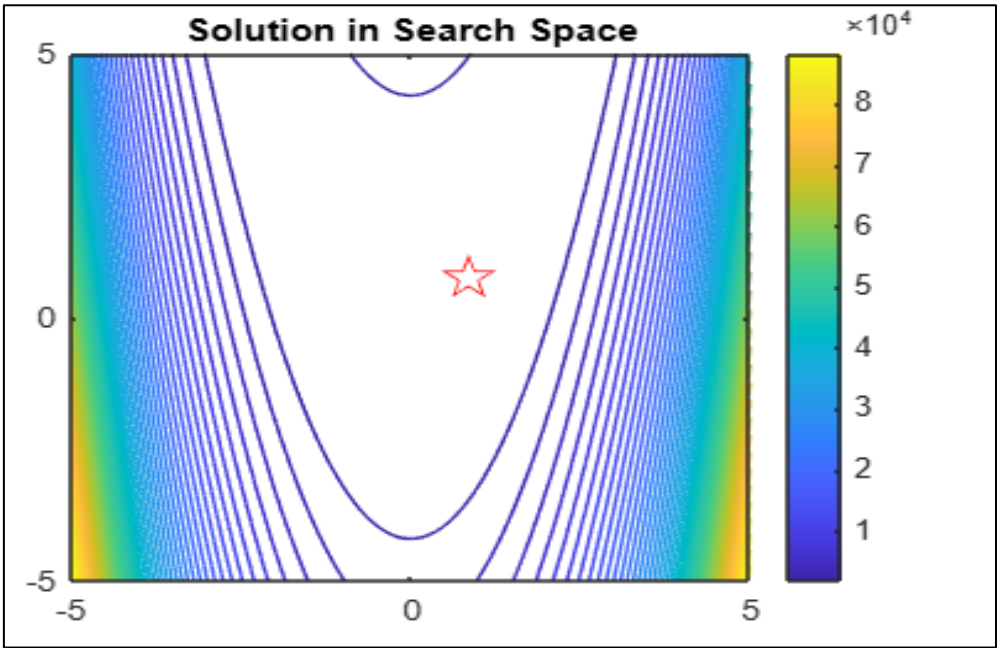


Figure 5. Solution in Search Space shows the final solution reached by the algorithm within the search space represented that the point ($X \approx 0.87, Y \approx 0.76$) indicates to an optimal point has been reached in the search environment.

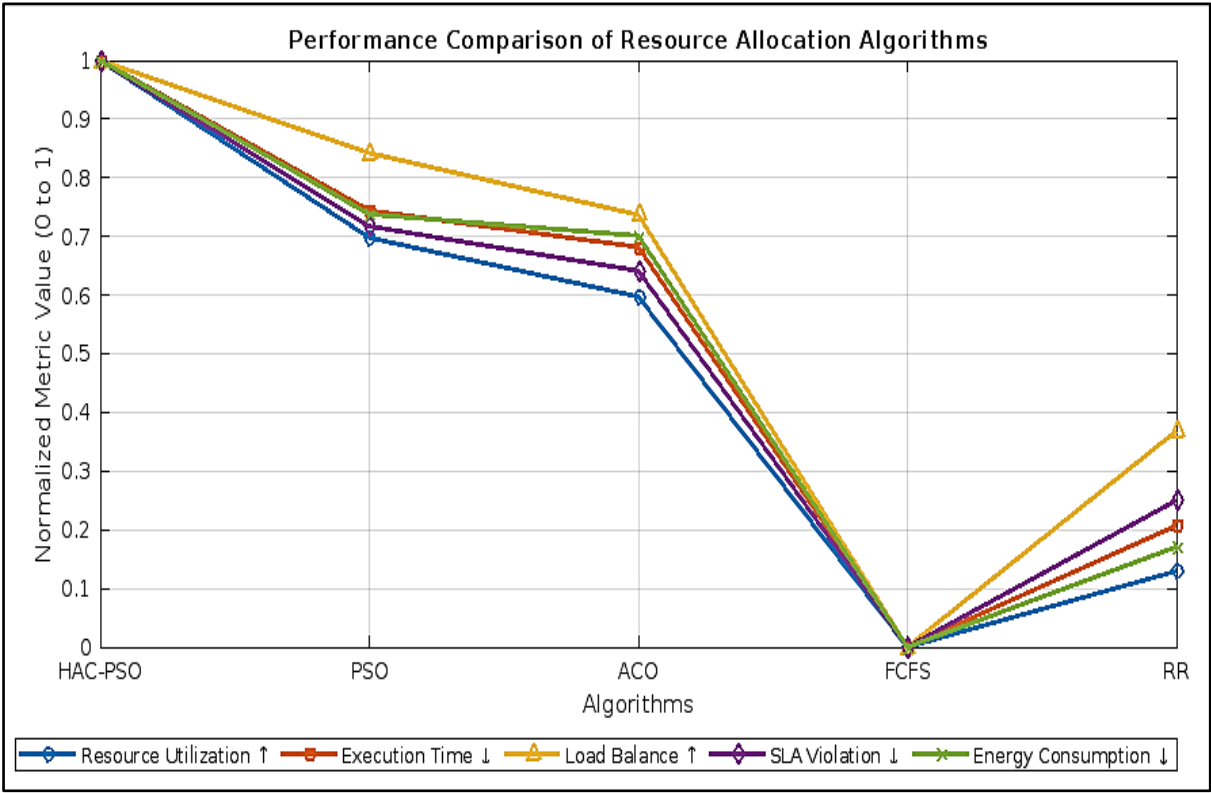


Figure 6. Performance Comparison of Resource Allocation Algorithms

Overall, hybrid PSO–ACO outperformed all baseline algorithms across all evaluation criteria, demonstrating its robustness and suitability for dynamic resource allocation in modern cloud environments as shown in Figure (6).

7. Conclusion

The hybrid nature of hybrid PSO–ACO allows it to combine the global search strength of **Particle Swarm Optimization** with the adaptive path-finding behavior of **Ant Colony Optimization**, resulting in a highly efficient, balanced, and adaptive solution for dynamic resource allocation in cloud computing environments. The algorithm significantly outperforms baseline methods in all key performance metrics, confirming its effectiveness and practicality for real-world cloud data center applications.

Resources

- [1] Yang, Y., Zhou, Y., Sun, Z., & Cruickshank, H. (2023). *Research on task scheduling algorithm optimization based on hybrid PSO and ACO in cloud computing*. *Computer Modelling and New Technologies*, 17(5A), 12–16.
- [2] Kumar, M.; Sharma, S.C.; Goel, S.; Mishra, S.K.; Husain, A. (2020). *Autonomic cloud resource provisioning and scheduling using meta-heuristic algorithm*. *Neural Compute*. 32, 18285–18303.
- [3] Sharma, V., & Thakur, M. (2021). *An improved HACO algorithm for workflow scheduling in cloud*. *Proceedings of the International Conference on Communication and Signal Processing (ICCSP 2021)*, 894–898.
- [4] Singh, S., & Chana, I. (2016). Q-aware: *Quality of service-based cloud resource provisioning*. *Computers & Electrical Engineering*, 47, 138–160.
- [5] Li, J. Y., Mei, K. Q., Zhong, M., et al. (2012). *Online optimization for scheduling preemptable tasks on IaaS cloud systems*. *Journal of Parallel and Distributed Computing*, 72(2), 666–677.
- [6] Zhao, Y., Liu, J., Zhang, X., & Dou, W. (2011). *A cloud computing framework for advanced manufacturing systems*. *Computers in Industry*, 62(8–9), 772–785.
- [7] Rahmani, M., et al. (2023). *Comparative analysis of metaheuristic loads balancing algorithms for cloud computing*. *Journal of Cloud Computing*.
- [8] Prasanna, G., Sankar Reddy, R. S., & Harini, B. S. (2021). *Fuzzy Hybrid Particle Swarm Parallel Ant Colony Optimization in Cloud Computing*. TURCOMAT.
- [9] Khadanga, R. K., & Swain, S. K. (2021). *Hybrid metaheuristic algorithms for task scheduling in cloud computing: A review*. *Journal of King Saud University – Computer and Information Sciences*.
- [10] Yang, Q., Chen, W.-N., Deng, J.-D., Li, Y., Gu, T., & Zhang, J. (2018). *A level-based learning swarm optimizer for large scale optimization*. *IEEE Transactions on Evolutionary Computation*, 22(4), 578–594. <https://doi.org/10.1109/TEVC.2017.2743016>.
- [11] Pandey, S., Wu, L., Guru, S. M., & Buyya, R. (2010). *A particle swarm optimization-based heuristic for scheduling workflow applications in cloud computing environments*. *Proceedings of the 24th IEEE International Conference on Advanced Information Networking and Applications*, 400–407.
- [12] Nie, Q., & Li, P. (2016). *An improved ant colony optimization algorithm for improving cloud resource utilization*. *Cyber Enabled Distributed Computing and Knowledge Discovery*, 311–314.
- [13] Saha, S., & Paul, S. (2021). *An efficient hybrid ACO–PSO approach for resource scheduling in cloud computing*. *Soft Computing*, 25(17), 11293–11310. <https://doi.org/10.1007/s00500-021-05892-w>
- [14] Basha, S. M., & Kumar, Y. S. (2022). *Metaheuristic scheduling for cloud resource allocation: A hybrid PSO–ACO approach*. *Procedia Computer Science*, 184, 734–741. <https://doi.org/10.1016/j.procs.2021.04.092>.
- [15] Patel, A., & Desai, A. (2024). *Load balancing in cloud computing environment using hybrid particle swarm optimization and ant colony optimization algorithm*. *Proceedings of the ICAICIT 2024 Conference*. <https://www.researchgate.net/publication/389841661>.
- [16] Khan, Z. A., & Ahmad, M. (2017). *Hybrid particle swarm optimization and ant colony optimization for dynamic VM scheduling in cloud environment*. *WSEAS Transactions on Computers*, 16, 121–128.
- [17] Singh, S., & Chana, I. (2016). *Hybrid Ant Particle Swarm Genetic Algorithm (APSGA) for task scheduling in cloud computing*. In *Information and Communication Technology for Competitive Strategies (ICTCS)*. *The Journal of Supercomputing*, 61(2), 337–352.