IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Optimized Workflow Scheduling In Cloud Computing Using Collaborative Heuristic Model

Vinay Kumar Sriperambuduri[1], Neha Reddy Thummala[2], Katta Sruthi Reddy[3]

[1,2,3]Vasavi College of Engineering, Hyderabad, India

Abstract

The era of relying solely on local hard drives for data storage and retrieval has become a thing of the past. Cloud computing has transformed how we store and access data, offering flexible and cost-effective solutions. However, one major challenge in cloud environments, especially in Infrastructure-as-a-Service (IaaS) clouds, is the efficient scheduling of workflows—particularly when it comes to complex scientific tasks. Efficient scheduling is key to ensuring that resources are used optimally, workflows run faster, and costs are minimized. While a lot of research has been done on this topic, many approaches overlook cloud-specific features like elasticity and the diversity of available resources, which are crucial for achieving the best performance. In this study, we present a hybrid algorithm that combines Ant Colony Optimization (ACO) and Particle Swarm Optimization (PSO) to tackle these issues. Our approach aims to meet both local and global goals by taking advantage of the strengths of both algorithms. The result is a more efficient use of cloud resources and faster execution of workflows. Through our experiments, we show that this hybrid algorithm can significantly improve scheduling efficiency, offering a practical solution for managing scientific workflows in the cloud.

1.Introduction

Cloud computing has revolutionized data processing, enabling on-demand access to scalable and virtualized computing resources. As organizations increasingly migrate complex applications to the cloud, scientific workflows-representing a series of computational tasks with interdependencies-are gaining traction for domains such as genomics, astronomy, and climate modeling. These workflows consist of task nodes with dependencies and require optimized scheduling to improve efficiency and reduce costs.

However, efficient scheduling of these workflows in cloud environments presents an NP-hard optimization problem due to resource heterogeneity, cost-performance trade- offs, fluctuating workloads, and QoS constraints. Traditional methods often fall short when it comes adapting to dynamic workload conditions and heterogeneous resource availability in real time.

Workflow scheduling determines the optimal mapping of tasks to cloud resources to meet quality of service (QoS) requirements such as minimizing execution time (makespan) and reducing cost. Traditional heuristic methods, including Min-Min and HEFT, are fast but often settle for suboptimal solutions. Metaheuristic approaches, such as Genetic Algorithms, Particle Swarm Optimization (PSO), and Ant Colony Optimization (ACO), offer better global search capabilities but may struggle with local refinement or suffer from premature convergence.

To address these challenges, we propose a hybrid model that collaboratively integrates PSO and ACO. The rationale behind this combination lies strength in global exploration and ACO's effectiveness in refining solutions through PSO's probabilistic learning. By allowing these algorithms to exchange information periodically, we aim to achieve better convergence towards optimal task scheduling. The model is validated using WorkflowSim and benchmark scientific workflows, and results indicate considerable improvements in both makespan and cost compared to standalone techniques. The model is envisioned to support applications with high computational demand and strict deadline constraints, opening new opportunities in edge-cloud coordination, real-time analytics, and service-oriented architectures.

2. Literature Review

Workflow scheduling has been extensively studied with numerous heuristic and metaheuristic approaches. Heuristics like Min-Min, Max-Min, and HEFT [1] provide fast scheduling but are prone to local optima. Metaheuristics like GA [2], PSO [3], ACO [4], and DE [5] offer better global exploration but often require fine-tuning. Hybrid approaches [6][7] have demonstrated better performance by combining strengths of individual techniques. Recent research has employed reinforcement learning [8], fuzzy logic [9], and cloud-specific optimizations [10] to enhance decision-making under uncertainty. Energy-aware and deadline-constrained scheduling strategies [11][14][15] are gaining traction for sustainable cloud computing. Ontology based discovery and intelligent provisioning mechanisms [12] have also contributed to more adaptive workflow management. Fair resource allocation and SLA-driven policies [13] are essential in multi-tenant cloud platforms.

Simulation tools such as CloudSim and WorkflowSim [13] are widely used for evaluating such algorithms under reproducible scenarios.Additional contributions involve hybrid GA-ACO frameworks with adaptive mutation techniques for faster convergence.Some efforts incorporate predictive analytics using neural networks for task duration estimation.Load balancing through decentralized agents and auction-based scheduling methods have also shown promising results.Intelligent fault tolerance and checkpointbased recovery strategies are gaining attention in fault-prone environments.

Our proposed model aligns with this trend by fusing PSO’s global convergence and ACO’s local adaptation into a unified collaborative heuristic framework, advancing the state-of-the-art in multi-criteria optimization for workflow scheduling.

3.Proposed Methodology

i. Problem Definition

The task scheduling problem aims to find an optimal solution where tasks need to be assigned to processors in a way that minimizes two primary objectives:

- **Makespan:** The time required to complete all tasks.
- **Cost:** The computational cost or resource usage involved in completing the tasks.

ii. Hybridization Strategy

The hybridization strategy involves two main components:

- **ACO** will first explore the search space and construct initial feasible schedules.
- **PSO** will then refine these solutions by fine-tuning the schedules to further reduce makespan and cost.

iii. ACO for Initial Schedule Construction

The first step in the hybrid algorithm is to use **Ant Colony Optimization** (ACO) for constructing initial schedules:

- **Pheromone Initialization:** Initialize pheromone values for each task and processor pair.
- **Task Assignment:** Each ant represents a solution, and each ant explores the schedule space by assigning tasks to processors.
- **Fitness Evaluation:** The fitness function evaluates each schedule based on **makespan** and **cost**. The fitness function is given by:

  $\text{Fitness}=\alpha\times\text{Makespan}+\beta\times\text{CostFitness}$

  where **α** and **β** are weighting factors that control the importance of each objective.

- **Pheromone Adjustment:** After solutions have been constructed, the pheromones are altered. The pheromone value for each task-processor pair is updated based on the fitness of the solutions.
- **Exploration and Exploitation:** ACO explores new solutions while exploiting known good solutions via pheromone trails.

iv. PSO for Refining Solutions

Once ACO has generated a set of candidate schedules, **Particle Swarm Optimization(PSO)** is used to refine these solutions:

- **Initialization of Particles:** The best schedules from ACO are used to initialize the PSO particles.
- **Particle Velocity and Position Update**.
- **Fitness Evaluation:** Each particle’s fitness is evaluated using the same fitness function:

  $\text{Fitness}=\alpha\times \text{Makespan}+\beta\times \text{CostFitness}$

- **Particle Update:** Each particle's velocity is updated.
- **Personal and Global Best Update:** Each particle's personal best and the global best solution are updated based on the fitness values

v. Iterative Refinement and Convergence

The algorithm continues iterating between **ACO** and **PSO** until a stopping criterion is met. This could be a set number of iterations or a threshold value for **makespan** and **cost**. In each iteration:

- **ACO** explores the solution space and generates diverse solutions.
- **PSO** refines the solutions found by ACO to minimize the makespan and cost further.
- The best solution found during the iterations is returned as the final result.

vi. Final Solution

Once the stopping criteria are met, the algorithm outputs the **optimal workflow schedule** that minimizes both **makespan** and **cost**. This solution represents the best balance between exploration (ACO) and refinement (PSO)

Algorithms used:

Algorithm 1 Ant Colony Optimization for Workflow Scheduling

```
 1: Initialize pheromone trails $\tau_{ij}$ and heuristic information $\eta_{ij}$
 2: Set parameters: $\alpha$, $\beta$, $\rho$, $Q$, number of ants $m$, maximum iterations
    $MaxIterations$
 3: Initialize best schedule $S_{best}$ and best makespan $C_{best} \leftarrow \infty$
 4: for $iteration = 1$ to $MaxIterations$ do
 5:     for each ant $k = 1$ to $m$ do
 6:         Place ant at a randomly selected initial task
 7:         Initialize partial schedule $S_k$
 8:         while unscheduled tasks remain do
 9:             Calculate selection probabilities $p^k_i$
for available tasks:
$\tau^{\alpha}_{ij}\eta^{\beta}_{ij}$
10:             $p^k_i = \Sigma_{l \in allowed}\, \tau^{\alpha}_{il}\eta^{\beta}_{il}$
11:             Select next task $j$ based on $p^k_i$
12:             Assign task $j$ to a suitable resource
13:             Add task $j$ to schedule $S_k$
14:         end while
15:         Evaluate schedule $S_k$ and compute makespan $C_k$
16:         if $C_k < C_{best}$ then
17:             $Cbest \leftarrow Ck$
18:             $Sbest \leftarrow Sk$
19:         end if
20:     end for
21:     Evaporate pheromones for all edges:
22:         $\tau_{ij} \leftarrow (1-\rho)\tau_{ij}$
23:     for each ant $k$ do
24:         for each edge $(I, j)$ in $S_k$ do
25:             Deposit pheromone proportional to schedule quality:
26:             $\tau ij \leftarrow \tau ij + \frac{Q}{C_k}$
27:         end for
28:     end for
29: end for
```

```
30: return Best schedule $S_{best}$
```

Algorithm 2 PSO-based Workflow Scheduling

```
1:  Input: Workflow tasks (DAG), identical resources, swarm size $N$, max iterations $T_{\max}$,
    inertia weight $w$, coefficients $c_1$, $c_2$.
2:  Output: Best schedule $gBest$ (minimizing makespan and cost).
3:  for $i = 1$ to $N$ do
4:      Initialize particle $i$ with random schedule $x_i$ and velocity $v_i$
5:      Compute makespan $M_i$ and cost $C_i$ of schedule $x_i$
6:      Compute fitness $f_i = w_m \times M_i + w_c \times C_i$ (weighted sum of makespan and cost)
7:      Set personal best $pBest_i \leftarrow x_i$ and $f(pBest_i) \leftarrow f_i$
8:  end for
9:  Identify global best $gBest$ among all $pBest_i$
10: for $iter = 1$ to $T_{\max}$ do
11:     for $i = 1$ to $N$ do
12:         Update velocity: $v_i^{k+1} = \omega v_i^k + c_1 r_1 (pbest_i - x_i^k) + c_2 r_2 (gbest - x_i^k)$
13:         Update position: $x_i \leftarrow x_i + v_i$
14:         Compute makespan $M_i$ and cost $C_i$ of new $x_i$
15:         Compute fitness $f_i = w_m \times M_i + w_c \times C_i$
16:         if $f_i < g(pgood_i)$ then
17:             $pgood_i \leftarrow x_i$
18:         end if
19:         if $f(pgood_i) < f(ggood)$ then
20:             $ggood \leftarrow pgoodi$
21:         end if
22:     end for
23: end for
24: return $gBest$
```

Algorithm 3 Hybrid ACO-PSO for Workflow Scheduling

```
1:  Initialize parameters for ACO: number of ants, pheromone evaporation rate, etc.
2:  Initialize parameters for PSO: number of particles, inertia weight, cog- nitive and social
    coefficients
3:  for $iteration = 1, 2, \ldots,$ until ACO stopping criterion is met do
4:      for each ant do
5:          Construct a feasible workflow schedule based on pheromone and heuristic
    information
6:          Calculate fitness using:
$$Fitness = \alpha \times Makespan + \beta \times Cost$$
    where $\alpha$ and $\beta$ are weighting factors
7:      end for
8:      Update pheromone trails based on the fitness values
9:  end for
10: Select the best solutions found by ACO as the initial swarm for PSO
11: for $iteration = 1, 2, \ldots,$ until PSO stopping criterion is met do
12:     for each particle do
13:         Update velocity based on individual and global best positions
14:         to generate a new workflow schedule
15:         Calculate fitness using:
```

```
$Fitness = \alpha \times Makespan + \beta \times Cost$
16:     end for
17:     Update personal best and global best based on fitness values
18: end for
19: Return the best workflow schedule with minimum fitness value
```

4. Experimental Setup And Result Analysis

The proposed algorithm is implemented with 16-core virtual machines with 32 GB RAM on LINUX platform as access to the infra incurs a very high cost. Thus the proposed algorithm is implemented in a simulated environment using WorkflowSim simulation toolkit. It is an extension of the popular **CloudSim** framework, which is primarily used for cloud resource management simulations. WorkflowSim allows for simulating the execution of complex workflows.

Parameter Setting:

The parameter setting for the experiment involves configuring the cloud data center, virtual machines (VMs), and cloudlets (tasks within workflows). Specific resource allocations such as CPU, memory, and network bandwidth are predefined for each component. Workflows with varying task dependencies and execution times are simulated to assess performance. The comparison focuses on two key metrics: **cost** and **time delay** to evaluate the proposed algorithm against the PSO algorithm.

RESULTS ANALYSIS:

TABLE I

Cybershake

	Average Cost(PSO)	**Average Cost(ACO)**	**Average Cost(Hybrid PSO_ACO)**
Cybershake_30	**19442.24**	**19445.98**	**19444.11**
Cybershake_50	**38297.91**	**38302.67**	**38300.29**
Cybershake_100	**76664.48**	**76671.95**	**76668.22**
Cybershake_1000	**127811.51**	**127824.27**	**127817.89**

	Average Makespan(PSO)	Average Makespan(ACO)	Average Makespan(Hybrid PSO_ACO)
Cybershake_30	**840.95**	**577.29**	**310.14**
Cybershake_50	**1684.74**	**1296.12**	**814.87**
Cybershake_100	**3534.43**	**3021.19**	**1104.87**
Cybershake_1000	**22913.67**	**21974.07**	**5484.58**

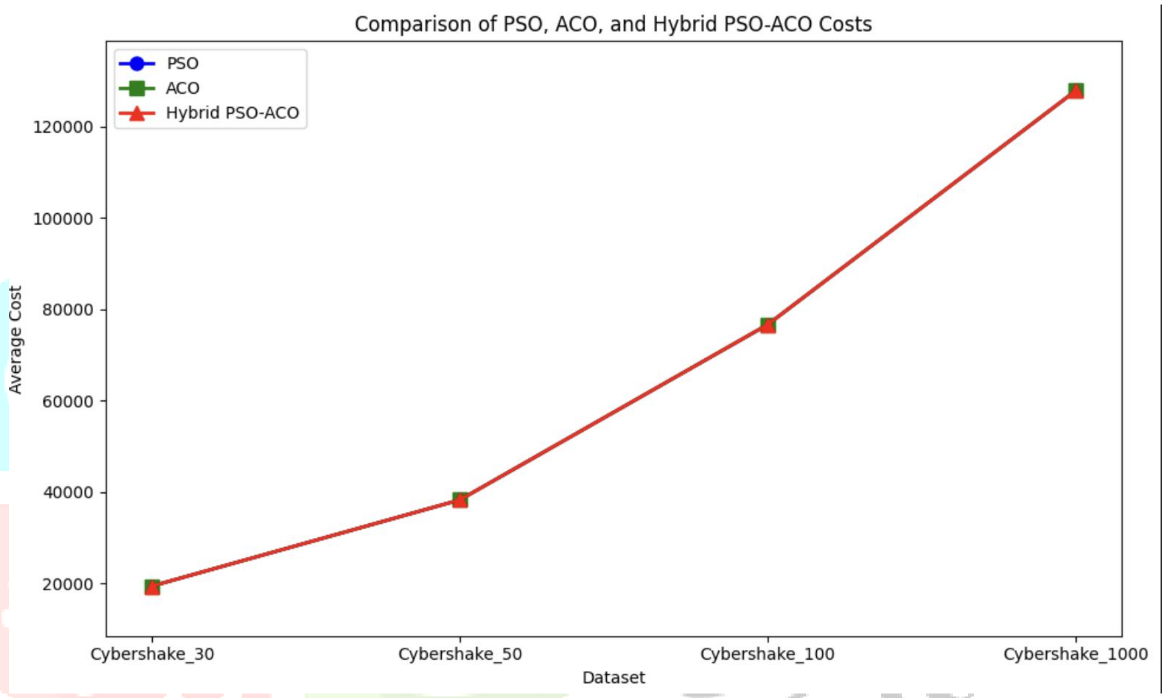

Fig. 1. Cybershake Cost comparison graph

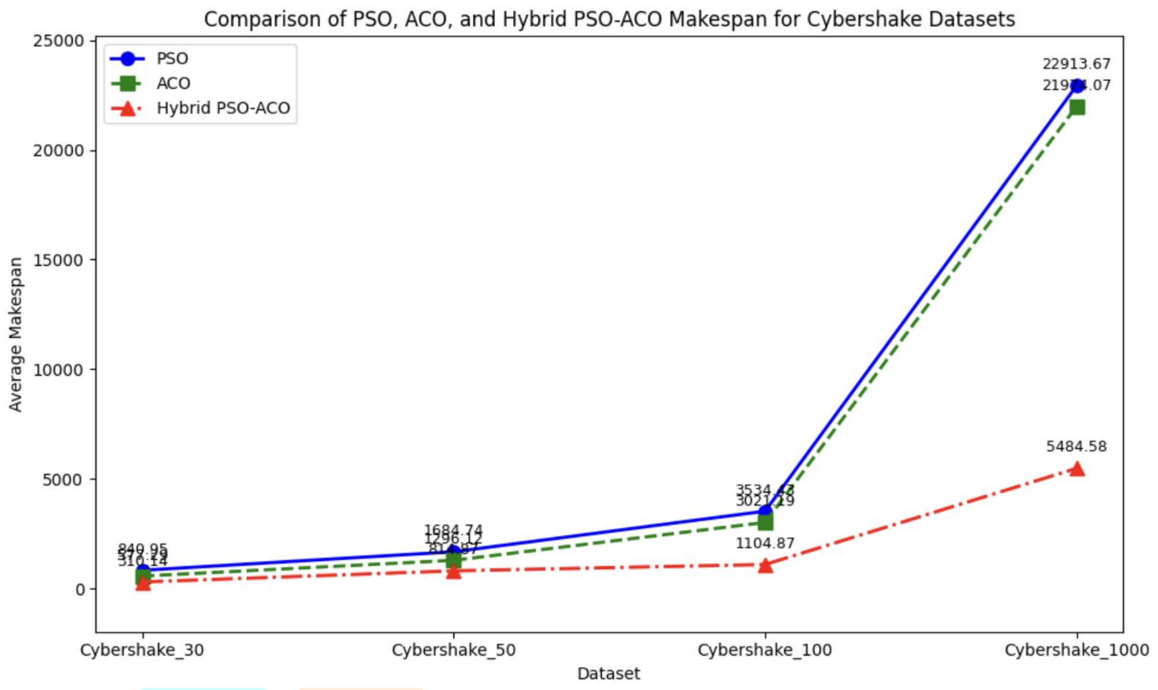

Fig. 2. Cybershake Makespan comparison graph

Key Findings:

The key findings from the experiment, which utilized the **Cybershake** and **Montage** workflows, reveal that the **Hybrid PSO-ACO algorithm** significantly outperforms both the **PSO** and **ACO** algorithms in optimizing cost and makespan. The hybrid approach consistently delivers lower makespan and cost, demonstrating superior efficiency in resource allocation and task scheduling. While **PSO** excels in reducing makespan and **ACO** performs better in minimizing cost, the hybrid method effectively balances both, providing a more optimal solution. These results highlight the advantage of combining multiple optimization techniques for enhanced performance in cloud-based scientific workflow execution.

6. Conclusion

This thesis has addressed one effective scheduling method for scientific workflows in an IaaS cloud configuration. the problem was considered as a two-part optimization problem to minimize total execution cost and makespan and to solve this problem a hybrid algorithm was developed based on Ant Colony Optimization (ACO) and Particle Swarm Optimization (PSO). The simulation experiments were conducted in the cloud infrastructure using two known scientific workflows Cybershake and Montage. It was demonstrated that the solution provided by us had gains in efficiency over the state-of-the-art algorithms that were implemented.

The outcomes acheived could be explained due to the use of PSO and ACO offering orthogonal solutions. In this case PSO was useful for global search and conergence of solutions while ACO was usefull for local exploration. With these two components a large proportion of workflows were achieved where delays were reduced, effective schedules were constructed and executed, and overall cost were reduced.

For the next steps that need to be undertaken for future research is try other mechanisms for determining the first resource pool because this aids in improving the efficiency of scheduling. We intend to study other metaheuristics and examine their performance against PSO, ACO, and hybrid versions of the two.Another promising direction is to analyze data transfer costs between geographically distributed data centers, enabling optimized VM placement. Ultimately, we aim to integrate our scheduling technique into a real-world workflow management system to facilitate practical application deployment

7. References

[1] Topcuoglu, H., Hariri, S., Wu, M.Y. (2002). Performance-effective and lowcomplexity task scheduling for heterogeneous computing. IEEE TPDS.

[2] Braun, T.D., Siegel, H.J., Beck, N. et al. (2001). A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems. JPDC.

[3] Zhang, Y., Zhang, W., & Zhang, J. (2009). Workflow scheduling using PSO algorithm in cloud computing. Proc. ICCSE. [

4] Dorigo, M., & Gambardella, L. M. (1997). Ant colony system: A cooperative learning approach to the traveling salesman problem. IEEE TEC.

[5] Qin, J., & Li, Y. (2011). An improved DE algorithm for cloud workflow scheduling. Future Generation Computer Systems.

[6] Pandey, S., Wu, L., Guru, S. M., & Buyya, R. (2010). A particle swarm optimization-based heuristic for scheduling workflow applications in cloud computing environments. AINA.

[7] Zhao, J., Liu, Q., Tang, M., & Li, K. (2015). A hybrid heuristic algorithm for workflow scheduling in cloud computing. Journal of Computer and System Sciences.

[8] Mao, H., Alizadeh, M., Menache, I., & Kandula, S. (2016). Resource management with deep reinforcement learning. HotNets.

[9] Singh, S., Chana, I. (2016). Cloud resource provisioning: survey, status and future research directions. Knowledge and Information Systems.

[10] Abrishami, S., Naghibzadeh, M., & Epema, D.H.J. (2012). Deadline-constrained workflow scheduling algorithms for Infrastructure as a Service Clouds. Future Generation Computer Systems. [

[11] Chen, W. N., & Zhang, J. (2009). An ant colony optimization approach to a grid workflow scheduling problem with various QoS requirements. IEEE TSC.

[12] Yu, J., & Buyya, R. (2005). A taxonomy of workflow management systems for grid computing. Journal of Grid Computing.

[13] Calheiros, R. N., Ranjan, R., Beloglazov, A., De Rose, C. A., & Buyya, R. (2011). CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Software: Practice and Experience. [

14] Arabnejad, H., & Barbosa, J. G. (2014). Fair resource sharing for dynamic workflow scheduling on heterogeneous systems. Future Generation Computer Systems.

[15] Dastjerdi, A. V., Tabatabaei, S. G. H., & Buyya, R. (2010). An effective architecture for automated appliance management system applying ontology-based cloud discovery. Proc. ICA3PP.