International Journal of Mathematical, Engineering and Management Sciences
Vol. 11, No. 2, 831-849, 2026
https://doi.org/10.33889/IJMEMS.2026.11.2.035

Pareto-optimized Hybrid Metaheuristic Scheduling for Energy-Efficient and Reliable Fog Computing

Shilpa Dinesh Vispute
Department of Computer Science and Engineering,
The NorthCap University, 122017, Gurugram, Haryana, India.
E-mail: shilpa20csd002@ncuindia.edu

Meghna Sharma
Department of Computer Science and Engineering,
The NorthCap University, 122017, Gurugram, Haryana, India.
E-mail: meghnasharma@ncuindia.edu

Priyanka Vashisht
Department of Computer Science and Engineering,
Amity University Haryana, 122017, Gurugram, Haryana, India.
Corresponding author: priyanka.vashisht@gmail.com

(Received on October 3, 2025; Revised on December 30, 2025; Accepted February 6, 2026)

Abstract
Excessive advancement in technology and the Internet of Things (IoT) have brought a revolution in Cloud Computing. However, a massive amount of data is generated from IoT devices, ultimately affecting the Cloud's efficiency. Fog Computing has been developed to improve efficiency. It places Fog nodes near the IOT devices to reduce the processing latency. Fog computing has achieved remarkable success; however, its nodes are resource-constrained, which makes efficient Task-scheduling. The proper scheduling of tasks among appropriate Fog nodes, considering the energy efficiency and reliability in terms of failure rate, is the main challenge for researchers. Many task scheduling algorithms have been proposed in the literature for energy efficiency and makespan; however, very little attention is paid to reliability. The high computational time of the task scheduling algorithm is another crucial aspect of these algorithms. This paper proposes a Reliable Hybrid Pareto-based Multi-objective (RHPMO) Task Scheduling approach based on metaheuristics. It combines two advanced metaheuristic techniques, named JAYA and Genetic Algorithm, to perform optimal task scheduling based on the Pareto front and explore the best global solution. A multi-objective function is formulated to minimize makespan, energy, and failure rate, thereby increasing the efficiency and reliability of task scheduling. Extensive experimentation is conducted on MATLAB R2023a on an Intel i3, 8 GB RAM. The simulation experiment results of the proposed hybrid approach are compared with the various metaheuristic approaches like JAYA, Genetic Algorithm, Particle Swarm Optimization, Bees Algorithm, and Ant Colony Optimization, and hybrid GA and PSO. The proposed approach surpasses various state-of-the-art studies and demonstrates an impressive improvement of 68.91% in computational time, 42.48% in makespan, 29.83% in energy consumption, and 12.58% in reliability, respectively.

Keywords- Energy efficiency, Multi-objective, Fog computing, Reliability, Task scheduling.

1. Introduction

Cloud Computing has been extensively used by public and private organizations to process and store their data. The Cloud infrastructure is comprised of robust processing equipment and can handle demanding requests from massive data-generating Internet of Things (IOT) devices; however, massive data transmission is quite challenging due to network bandwidth limitations. Also, it is not flexible in efficient resource allocation among various tasks, which introduces processing delay. In cloud computing, long-distance data transmission to centralized servers also increases energy consumption (Vashisht and Kumar, 2022). To handle various delay-sensitive applications, Fog computing is integrated with the Cloud (Zitar et al., 2022). In a distributed environment like Fog computing, a data grid helps manage and distribute data

efficiently across various fog nodes located closer to end devices. It ensures that data required for processing tasks is easily accessible, thereby reducing latency and enhancing real-time performance (Vashisht et al., 2019a). To increase data availability, replica management is used to create and maintain copies of data on multiple nearby fog nodes. As a result, task scheduling becomes faster, more reliable, and energy-efficient, enhancing the overall effectiveness of fog-based systems (Vashisht and Kumar, 2020). Optimization of replica management focuses on deciding how many replicas to create, where to place them, and when to update or remove them to achieve the best system performance. The goal is to balance data availability and latency (Vashisht et al., 2019b). Fog computing is applied in many areas including traffic management by avoiding critical scenario of congestion. (Singh et al., 2022). Fog computing has been widely used in many fields, such as medical, agricultural, and manufacturing etc. (Madni et al., 2024).

As Fog Computing continues to advance in the IoT application area, it faces some challenges as well. Fog nodes have limited processing capabilities, so they can handle a certain number of tasks. Hence, the researchers face a significant challenge of scheduling multiple tasks among the limited nodes available in the Fog network. Energy consumption and failure rate are two essential parameters for task scheduling, indicating the efficiency and reliability of the Fog nodes, respectively. Less energy consumption by Fog nodes helps to reduce the operational cost; however, sometimes, lower energy consumption may result in slower processing, so it is also important to focus on optimization approaches while minimizing the consumption of energy to meet application needs (Vashisht et al., 2024). Hence, energy optimization along with makespan time must be considered during task scheduling. Another important aspect is reliability, i.e., Fog nodes should demonstrate a low failure rate with minimal delay. Therefore, a task scheduling algorithm needs to fulfill multiple objectives of reliability, energy efficiency, and minimum makespan. Various conventional task scheduling algorithms like Round-Robin, First-come First-served, etc., have been used in Fog computing (Hashemi et al., 2022). Nowadays, various meta-heuristic-based scheduling algorithms are being used for the same due to their better performance. Among the meta-heuristic approaches, population-based meta-heuristic algorithms like Multi-objective Evolutionary Algorithm based on Decomposition (MOEA/D) (Zhang and Li, 2007), Non-Dominated Sorting Genetic Algorithm III (NSGA-III) (Deb and Jain, 2014), JAYA algorithm (Rao, 2016), Particle Swarm Optimization (PSO) (Wang et al., 2018), Genetic Algorithm (GA) (Katoch et al., 2021), Bees Algorithm (BA) (Ismail et al., 2020), Ant Colony Optimization (ACO) (Blum, 2024), Strength Pareto Evolutionary Algorithm II (SPEA-II) (Daghayeghi and Nickray, 2024) demonstrated better performance in task scheduling. However, these algorithms are mainly focused on energy efficiency and makespan minimization; little attention is being paid for the reliability. Reliability is a crucial aspect of task scheduling, as a high failure rate may degrade the overall Quality of Service (QOS). Reliability is important in scheduling because it ensures tasks are completed correctly and on time, even if some nodes fail. Considering reliability helps maintain system stability, minimize failures (Pradhan et al., 2023).

To address this challenge, this paper presents a hybrid multi-objective task scheduling approach. The proposed approach is based on two population-based metaheuristic techniques named GA (Katoch et al., 2021) and JAYA (Rao, 2016). GA utilizes selection, crossover, and mutation operators to solve real-world application problems. GA is applicable to the large and complex decision space. Basically, GA is suitable for applications where traditional analytical methods may be insufficient (Kumar et al., 2019). In dynamic Fog Computing environments, the GA algorithm rapidly allocates resources, with a reduction in makespan and energy consumption. By automating the search for optimal solutions, GA simplifies resource management without requiring exhaustive computations (Kumar and Sahni, 2020). But GA has a tendency to get stuck in local optima due to premature convergence. To mitigate this, the JAYA has been used as it reduces the risk of local optima trapping and maintains diversity. Furthermore, this paper also considered the reliability objective with makespan and energy consumption for task scheduling in terms of failure rate

because it ensures that tasks are executed successfully without failures. High reliability reduces the risk of task or node failures, improves the system's QoS. In a Fog environment, reliability assures tasks are assigned to appropriate Fog nodes, maintaining efficient system performance. Also, this paper utilizes a Pareto optimal front (Deb et al., 2002) with the non-dominated sorting algorithm (Deb et al., 2002), and the crowding distance algorithm (Deb et al., 2002). The Pareto front shows optimal trade-offs among objectives, ensuring better overall performance. Therefore, this Fog-based proposed hybrid multi-objective task scheduling approach is ideal for applications such as manufacturing, smart grid, hospitals, and healthcare systems that require efficient, reliable, and multi-objective criteria.

The key contribution of this paper is as mentioned below.

- A highly precise and efficient hybrid multi-objective model is proposed based on JAYA and GA, enhancing reliability in task scheduling for fog computing.
- A customized multi-objective function is introduced to minimize makespan, energy, and failure rate, increasing the efficiency of the algorithm.
- To trade off multiple objectives, a Pareto optimal front is utilized with the non-dominated sorting algorithm and the crowding distance algorithm.
- The simulation results of the proposed hybrid algorithm surpass other metaheuristic algorithms like JAYA, GA, PSO, BA, ACO, and the hybrid approach of GA-PSO (Saad et.al., 2024). The proposed algorithm has shown an outstanding enhancement of computational time by 68.91% and 12.58 % in reliability.
- The proposed algorithm also significantly reduced energy consumption by 29.83%, and 42.48% in makespan, making it highly recommendable for real-world applications.

The paper mainly consists of the following sections. The related research articles on task scheduling in Fog Computing are explained in Section 2. The introduction of the methodology for the proposed work is explained in Section 3. Section 4 elaborates on the proposed task scheduling, including the proposed algorithm. The analysis of simulation results is explained in Section 5. The conclusion of this research is drawn in Section 6.

2. Literature Survey

Numerous task scheduling algorithms have been proposed for Fog Computing using traditional scheduling algorithms and meta-heuristic algorithms. A brief overview of these studies is presented in **Table 1**.

Daghayehi and Nickray (2024) proposed a task scheduling model based on the Evolutionary Algorithm. The authors proposed a modified Pareto Evolutionary algorithm to minimize energy consumption and response time. Authors also addressed deadline requirements and load balancing issues. They utilized a specialized crossover and mutation operators to obtain an optimal scheduling policy. This algorithm surpasses existing benchmark algorithms in minimizing response time and energy usage. However, the failure of fog nodes has not been considered, which can significantly influence the reliability of the system, leading to service disruption. Moreover, the evaluation has been carried out with a limited number of evaluation parameters (Daghayeghi and Nickray, 2024).

Husain et al. (2024) introduced a prioritized task scheduling approach, designed for a Fog environment. This approach was simulated in iFogSim, and the result was evaluated in terms of task deadlines, cost, and makespan. But several crucial QoS metrics, such as energy consumption and reliability of the Fog node, have not been considered. Including these factors would provide a more diverse assessment of scheduling efficiency.

Lera and Guerrero (2024) designed an optimization solution based on deep reinforcement to solve a challenge of multi-objective placement applications in the Fog area. The authors implemented actor–critic mechanisms to prioritize interconnected services within an application. Service dependencies are placed directly into the learning process, ensuring interdependent services are prioritized during placement decisions. The evaluation results for execution time achieve a better Pareto set compared to the traditional techniques. However, the implementation does not focus on multiple constrained simple reward functions.

Rao and Qin (2024) proposed a Fog-Cloud framework along with a mathematical model for scheduling tasks. The authors presented a Hybrid Equilibrium Optimizer to achieve task deadlines. This approach demonstrated superior performance in makespan and average waiting time. However, the efficiency of task scheduling is affected as the proposed model does not consider dynamic task characteristics.

Bhaskar et al. (2025) developed an Efficient Optimization approach based on the Moth-Flame for task scheduling. The method prioritized critical tasks to ensure the deadline, optimizing energy consumption. The chaos-based logistic map is used to increase diversity. Simulation results show better results than baseline optimization approaches for execution time, energy usage, and makespan. But a hybrid task scheduling approach with novel optimization algorithms, such as metaheuristics, can give better performance.

Sui et al. (2025) proposed a multi-strategy technique to solve the task scheduling issue for Cloud and Fog platforms. This algorithm focuses on insufficient population diversity and low efficiency. The algorithm exploration coefficient is used to remove the local optima problem. This research aims to schedule tasks efficiently, achieving minimum execution time and operating cost. However, the efficiency of the proposed algorithm is limited to single-objective optimization.

Nkenyereye et al. (2025) implemented an orchestration prototype by integrating a lightweight Kubernetes platform for Edge Computing using a Raspberry Pi 4 simulation. This research work is based on the Argo workflow for preparing the sequence of applications. Evaluation results indicate a 15% reduction in average scheduling time compared to the baseline scheduling approach. However, the proposed algorithm is not scalable and reliable because of the lack of integration of Argo workflows and Argo events.

Sehgal et al. (2025) proposed an Optimal Energy Security Aware Scheduling (OESAS) based on awareness regarding security and energy in the Fog network. The credit-based system is implemented to balance the load across the network. The OESA algorithm demonstrated better performance in energy efficiency and cost reduction. However, performance enhancement can be achieved by focusing on dynamic resource scheduling. The integration of trust mechanisms while maintaining efficiency can better support real-time decision-making.

Chen et al. (2025) developed an improved genetic algorithm combined with the sparrow search algorithm (SSA). The authors implemented this integrated approach, reducing the delay and obtaining the maximum system utility. A space-air-ground integrated network for a vehicular edge platform is developed to enhance the network capability and privacy protection. The urgency level is used to fulfill the delay needs of tasks. The proposed approach shows better performance in terms of the total time and the system utility; however, the system performance is not evaluated regarding energy consumption and system reliability.

The studied literature review shows that very few studies have addressed energy consumption and the reliability of the task scheduling algorithm. Energy consumption is a vital parameter in task scheduling for Fog Computing. The Fog network is often resource-constrained with limited capacity. Energy consumption

reduces operational costs with increasing Fog node lifespan. One crucial parameter in Fog Computing, while task scheduling is reliability. To handle latency-sensitive tasks of Fog Computing, minimizing Fog node failure is important. Thus, energy consumption and reliability are important metrics during the design of task scheduling approaches in Fog environments.

The traditional task scheduling algorithms have demonstrated limited performance in energy consumption and reliability. There is a clear need for energy-efficient and reliable task scheduling methods, such as metaheuristic algorithms, to solve complex optimization problems of task scheduling. To overcome these limitations, this paper presents a hybrid task scheduling model integrating JAYA and GA metaheuristic approaches. Hybrid metaheuristics algorithms integrate the strengths of two or more optimization algorithms to overcome the limitations of individual approaches, offering faster optimization for multi-objective problems such as task scheduling. The proposed algorithm considers non-dominated solutions using the Pareto front to remove dominated solutions, so it generates non-dominated solutions for makespan, energy, and reliability. Furthermore, a multi-objective function is formulated to increase the energy efficiency and reliability of task scheduling.

3. Proposed Methodology

This paper proposes a Reliable Hybrid Pareto-based multi-objective (RHPMO) model to address the complex problem of reliability in task scheduling of the Fog environment. The proposed approach integrates the JAYA and GA metaheuristic algorithms for enhanced reliability. The JAYA algorithm iteratively moves candidate solutions nearer to the best value and far from the worst value, so it is more effective for multi-objective optimization problems. In the proposed approach, the JAYA is applied for the exploration phase. Another population-based GA is utilizing selection, crossover, and mutation operators to explore the solution space and identify near-optimal solutions for multi-objective complex problems. In the proposed model, GA is used for exploitation to find solutions from more probability regions. Hence, the hybrid mechanism leverages JAYA for fast convergence and GA for diversity preservation, producing efficient and robust scheduling solutions for Fog environments.

Table 1. Summary of literature review.

Authors	Optimization technique	Primary objectives	Research gap
Daghayeghi and Nickray (2024)	Pareto Evolutionary Algorithm	service response time and energy consumption	Consider fewer number of evaluation parameters
Hussain et al. (2024)	Resource Aware Prioritized Task Scheduling (RAPTS)	response time, task deadlines, cost, and makespan	The approach doesn't consider node failure
Lera and Guerrero (2024)	a multi-objective optimization for applications in fog computing	cost	Considers the solution front with a lack of a larger set of objectives
Rao and Qin (2024)	An enhanced hybrid Equilibrium Optimizer (EHEO)	makespan, total energy consumption, success rate, and average waiting time	Does not consider dynamic characteristics
Bhasker et al. (2025)	An optimization based on efficient Augmented Moth-Flame	energy consumption, makespan, and total time of execution	Can give better performance with hybrid optimized approach
Sui et al. (2025)	A multi-strategy fusion based on Mayfly Algorithm	transmission energy consumption, processing time and operating cost, convergence rate	Efficiency of proposed algorithm limited to single objective optimization
Nkenyereye et al. (2025)	An algorithm based on Argo workflow engine	execution time, a makespan, and energy usage	limited scalability
Sehgal et al. (2025)	Optimal Energy Security Aware Scheduling (OESAS) framework	Energy consumption, success ratio, makespan	Scheduler's performance can be improved using integration with Metaheuristic algorithm
Chen et al. (2025)	an integration of GA and the sparrow search algorithm (SSA)	System utility, total processing time	Doesn't consider reliability and energy consumption

The proposed approach simultaneously optimizes energy consumption and reliability by evaluating candidate solutions on these objectives. It is done by using a Pareto optimal front, ensuring that the objectives do not dominate each other. Pareto optimal font chooses the best solution among non-dominating solutions, providing enhanced reliability and reduced energy consumption. The proposed system model is shown in **Figure 1** and described in the following sub-section.

3.1 System Model

In a Fog computing environment, the Fog nodes receive the user request from the IoT devices and schedule the task according to the processing requirements; however, task scheduling is complex due to the limited network resources. To fulfill the user task in the least feasible amount of time, known as makespan, an effective and reliable task scheduling strategy is of utmost importance. Along with makespan, the scheduling algorithm must also ensure energy efficiency and failure rate of the system. **Figure 1** shows the proposed system model based on multiple objectives of minimum makespan, energy consumption, and failure rate. Users of the IoT network submit various tasks to the fog broker at the edge of the IoT layer. A fog broker consists of three separate but related components: resource monitor, task monitor, and task scheduler. The resource monitor stores the information about the available capacity of the Fog nodes. The progress of task execution is tracked by the task monitor, which sends the processing result to the IoT users. The resource monitoring component's job is to keep information about Fog nodes and assess them before allocating tasks. The resource monitor then periodically provides the task scheduler unit with a report on the condition of the resources so that the task can be scheduled according to their availability. The task scheduler is the main component of the fog broker. To efficiently schedule tasks, the Fog node condition and user processing needs must be considered. The proposed model introduces a Reliable Hybrid Pareto-based multi-objective (RHPMO) algorithm to schedule tasks.

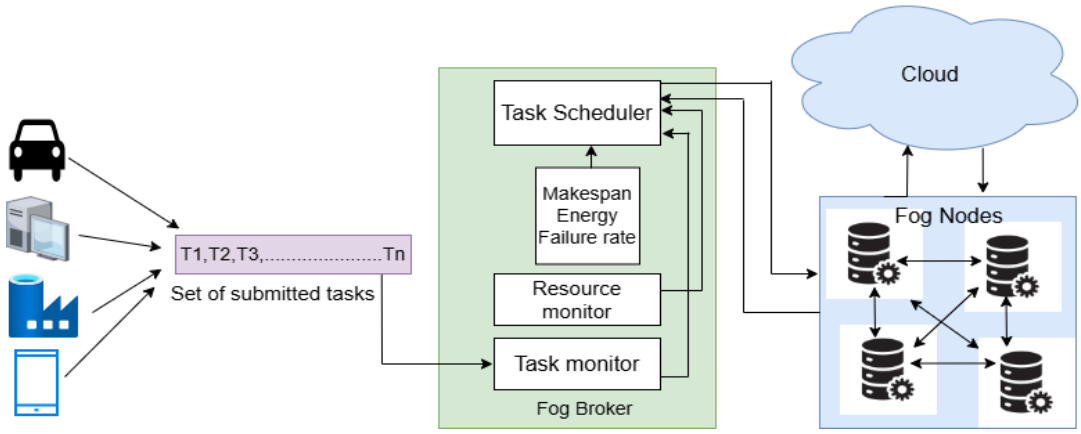

Figure 1. The proposed system model for RHPMO task scheduling.

3.2 Problem Formulation

The main aim of the proposed algorithm is to minimize makespan and energy consumption, and maximize reliability that depends on the task failure rate.

Consider, N is set of Fog nodes.

$$N = \{FN_1, FN_2, \ldots\ldots, FN_i\},$$

T is set of tasks.

$$T = \{T1, T2, \ldots., Tj\},$$

The following evaluation metrics are formulated to find out the fitness value. Makespan is measured in terms of the total time needed to complete a set of tasks. It is calculated from the beginning of the first task to the completion of the last task. So, it is the maximum execution time required to execute a set of tasks on a Fog node. Execution time for a task is formulated using Equation (1). where, TL(T_j^i) is Task length of task *j* assigned to Fog node *i*. It shows number of instructions required to complete tasks. CPURate(FN_i) shows the processing speed of Fog node *i*. Makespan is calculated using Equation (2).

$$\text{Exe}(T_j^i) = \text{TL}(T_j)/CPURate(FN_i) \quad (1)$$

$$M = \max_{i \in FN, j \in T} (Exe(T_j^i)) \quad (2)$$

Energy consumption is a vital parameter in task scheduling because it is related to the lifespan of Fog nodes. It also affects the overall operational cost of the scheduling process. So, considering energy consumption in task scheduling leads to cost-effective computing. Energy in the proposed algorithm is calculated using Equation (3).

$$E = \sum_{FN_i \in N, T_j \in T} Pow(FN_i) * [Exe(T_j) * T_s * T_r] \quad (3)$$

where, $Pow(FN_i)$ is the power consumption rate of Fog node *i*, $Exe(T_j)$ is the execution time required to execute task *j*, as shown in Equation (1). T_s is the transmission time needed to send data, and T_r is the transmission time needed to receive data.

Fog node reliability is calculated as shown in Equation (4), in which FR is the rate of failure for the Fog node. The failure rate of the Fog node is calculated by using Equation (5), where T_{fail} is the number of tasks the Fog node failed to serve, and T_{tot} is the total number of tasks. A lower failure rate means higher reliability. Efficient task scheduling algorithms aim to minimize failure rate by assigning tasks to more reliable nodes. The total number of failed tasks is the number of tasks that fail to complete successfully on a Fog node. A task is considered failed if it misses the deadline. So, if $Exe(T_j)$>Deadline(T_j), then the task is added to the set of failed tasks.

$$R=1\text{-}FR \quad (4)$$

$$FR = \frac{T_{fail}}{T_{tot}} \quad (5)$$

To formulate objective functions for minimizing makespan, energy, and failure rate, a module is proposed shown in Module 1. The fitness function of the proposed problem is expressed as,

$$Fit_{fun} = \text{Min}[f_1(x), f_2(x), f_3(x)] \quad (6)$$

where,
f_1=Makespan, f_2=Energy Consumption, f_3=Failure rate of node.

subject to:
Each task must be allocated to exactly one node.
$T_j \in N_i$,

The capacity of node C_i should be greater than the total load assigned to the node C_i.
$\sum T_j^i \leq C_i$.

4. Proposed Reliable Hybrid Pareto-based Multi-Objective (RHPMO) Task Scheduling Algorithm

This section presents a reliable hybrid Pareto-based multi-objective task scheduling approach with the JAYA algorithm and GA. The working process of the proposed algorithm is given below:

1) Initial population generated randomly
2) Calculate the fitness value
3) JAYA algorithm (Exploration phase)
   a. Select Best and Worst solutions
   b. Update population using the JAYA update function
4) Genetic algorithm (Exploitation phase)
   a. Initial solution is JAYA-generated population
   b. Evaluate the Fitness value of each solution
   c. Generate a new solution using Selection, Crossover, and Mutation operators
   d. Merge selected parent and new offspring to generate a new generation
5) Perform non-dominated sorting and calculate crowding distance
6) Repeat steps 2- 6 until the maximum iteration is reached
7) Return the best solution from the first Pareto front

The proposed algorithm is presented in Algorithm 1. This section explains the JAYA exploration phase and the GA exploitation phase. The flowchart for the JAYA-GA integration is shown in **Figure 2**. This section also introduces the non-dominating sorting (Deb et al., 2002) and calculation of crowding distance (Deb et al., 2002) to implement the Pareto front.

4.1 JAYA Exploration Phase

The proposed algorithm updates the initial population by using the JAYA algorithm. Let Best be the best value of a candidate obtained in the entire solution. Let Worst be the worst value of a candidate obtained in the entire solution. And, r_1 and r_2 are the two random numbers in the range [0, 1]. The solution is modified depending on the best and worst values using Equation (7) (Rao, 2016).

$$New \leftarrow Old + r_1(Best - |Old|) - r_2(Worst - |Old|) \quad (7)$$

The updated solution generated in the JAYA phase becomes the input to the GA phase.

4.2 GA Exploitation Phase

The GA (Katoch et al., 2020) relies on a population that utilizes the principle of "survival of the fittest." The approach includes mainly three genetic operators. The genetic operators are selection, mutation, and crossover. These operators are useful to produce better solutions.

Suppose P_n is the initial population generated in the JAYA phase. The selection operation selects P_1 as a set of first parents, and P_2 is selected as a set of second parents. The crossover operator works on P_1 and P_2 with a crossover rate C_r is as given below in Equation (8). C_r is the Crossover rate, which is set to 0.8 to produce a new solution with the best features.

$$P(n)=Crossover((P_{1(n-1)},P_{2(n-1)}),C_r) \quad (8)$$

The mutation operation is used to limit the crossover operator using searching for the optimal solution. The mutation operator rate is set too low to mutate offspring a little bit differently from parents. The mutation operator is as given below in Equation (9), in which M_r is the mutation rate, which is set to 0.05.

$$P(n)=Mutation(P(n)),M_r) \quad (9)$$

The next generation is formed using the parents and the mutated offspring. The generated solutions are then sorted out using the non-dominated sorting algorithm.

4.3 Non-Dominated Sorting Algorithm

In multi-objective optimization, non-dominated sorting is used to find solutions based on Pareto dominance as presented in Algorithm 2 (Deb et al., 2002). It shows the best trade-off between objectives. The first Pareto front includes the best solution, showing a non-dominated solution for all objectives. The second front contains solutions dominated only by the first front, and so on.

Algorithm 1. RHPMO algorithm.

Input:
Set of Fog nodes $\{FN_1, FN_2, \ldots, FN_i\}$
Set of Tasks $\{T_1, T_2, \ldots T_j\}$
Number of Iterations=MaxIt

Output:
Optimal scheduling of tasks on Fog nodes with minimum makespan, energy and failure rate

Initialize Population $P \in FN,T$
For each iteration It; It $\in 0$ *to MaxIt*
  Calculate FitVal using Module 1
  Find out Best solution
  Find out Worst solution
  Generate r_1 *and* $r_2 \in (0,1)$
  New $\leftarrow$ *Old* $+ r_1(Best - |Old|) - r_2(Worst-|Old|)$
    For each P
  Select second parents Par_{new} *from the updated population*
    For each Par_i *;* $i \in 0$ *to* Par_{new}
  Perform crossover operation
  Return offspring
    For each offspring
  Perform mutation operation
  Return muted offspring
 For each Muted offspring ; $i \in 0$ *to* Par_{new}
  Generate next population
  Perform non-dominating sorting and crowding distance
 Return best solution with respect to number of iteration

Module 1. To calculate fitness value.

Input:
Set of Fog Nodes $N = \{FN1, FN2, \dots, FNi\}$
Set of Tasks $T = \{T1, T2, \dots., Tj\}$
Output:
Fitness Values of objectives $M_{obj}, E_{obj}, F_{obj}$
Start:
For each Task $Tj \in T$
Calculate MinM $= \sum Min(M(FN_i))$
Calculate MinE $= \sum Min(E(T_j^i))$
Calculate MinF $= \sum Min(F(FN_i))$

$M_{obj} = (MinM_{obj})norm$
$E_{obj} = (MinE_{obj})norm$
$F_{obj} = (MinF_{obj})norm$
For end
End

Algorithm 2. Non-dominated sorting.

Input:
Population P
Output:
Non-dominated front $Front_i$
Start
For each $p \in P$ % *member p belongs to population P*
For each $q \in P$ % *member q belongs to population P*
if $(p < q)$ *then* % *if member p dominates member q*
$Sol_p = Sol_p \ U \{q\}$ % *add member q in* Sol_p
Else if $(q < p)$ % *if member p is dominated by member q*
$c_p = c_p + 1$ % *increase counter Cp*
if $c_p = 0$ *then*
$Front_1 = Front_1 \ U \{p\}$ % *add member p in first front*
$i = 1$
While $Front_i \neq \emptyset$
$L = \emptyset$
For each $p \in Front_i$ % For each member p belongs from $Front_i$
For each $q \in Sol_p$ % For each member from set Sol_p
$c_q = c_q - 1$ % *decrease* C_q *by 1*
if $c_q = 0$ *then* $L = L \ U \{q\}$ % *if* C_q *is zero then add q in list L*
$i = i + 1$
$Front_i = L$ % $Front_i$ *contains all members of L*
end

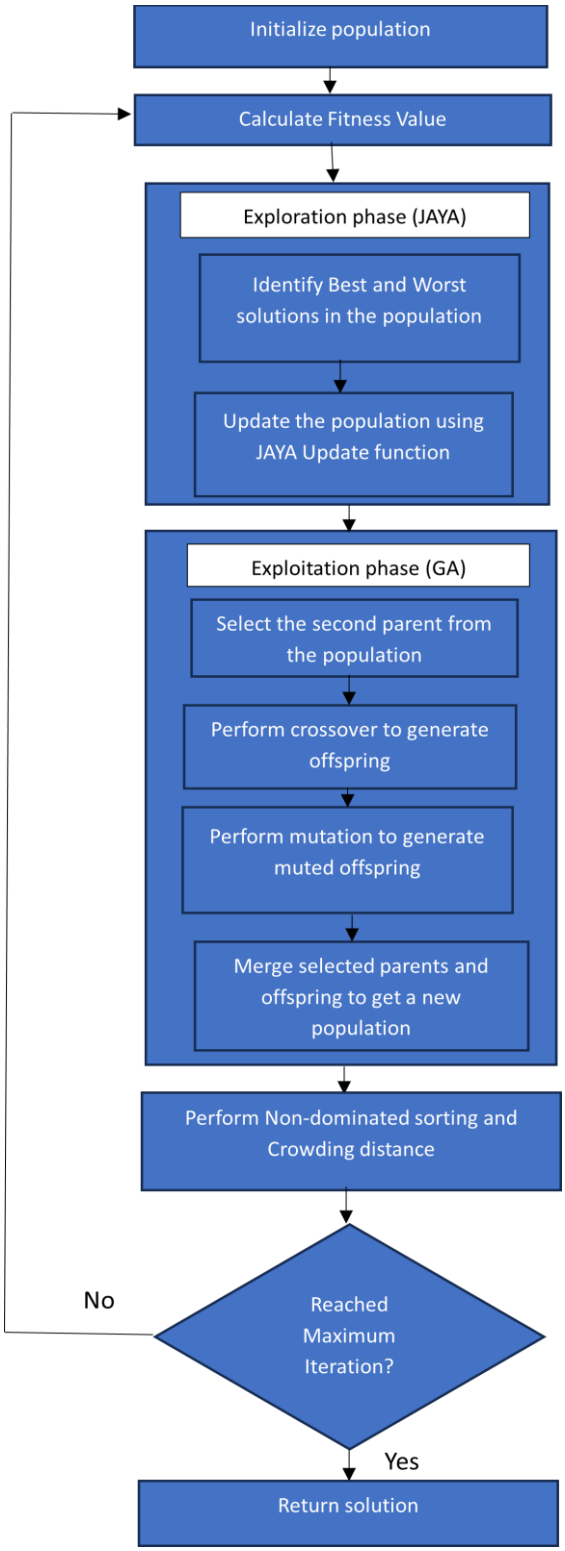

Figure 2. Flowchart for the proposed RHPMO algorithm.

4.4 Crowding Distance Calculation

In multi-objective optimization, crowding distance measures the diversity among solutions on the Pareto front, as shown in Algorithm 2 (Deb et al., 2002). Crowding distance estimates how close a solution is to its neighbors in the given objective space.

Algorithm 3. Calculating the Crowding distance.

Input:
Population parameters
Output:
Crowding distance for the solution $F[i]_{dist}$
Start
$N = |F|$ *% A Pareto front composed of N individuals*
For i= 1.....N do
$F[i]_{dist} = 0$
End for
For m= 1,...,M do *% M is total number of objectives*
SORT(F,m)
$F[1]_{dist} = F[N]_{dist} = \infty$ *%extreme solution set to infinity*
for i= 2,...,N-1 do
$F[i]_{dist} = F[i]_{dist} + (F[i+1]_{dist}.m - F[i-1]_{dist}.m)/(f_m^{max} - f_m^{min})$
% fmin_m minimum value of objective m
%fmax_m maximum value of objective m
end for
end for
end

The next section explains the simulation experiment evaluation of the proposed RHPMO algorithm. The evaluation results are compared with similar existing algorithms.

5. Analysis of Experiments and Results

This section presents an evaluation of the proposed RHPMO algorithm and its performance compared to existing algorithms. The simulation experiment of the proposed RHPMO algorithm is done in MATLAB. The configuration of the simulation is mentioned in **Table 2**. Simulation is performed with Intel(R) Core™ i3-1005G1 of 1.20GHz with 8 GB RAM. The proposed RHPMO algorithm is compared with existing algorithms like the JAYA algorithm, GA, PSO, BA, ACO, and hybrid GA-PSO, which shows better performance for the RHPMO approach. The hybrid GA-PSO was analyzed in terms of execution time, response time, and completion time. These evaluation metrics are closely associated with energy consumption and system reliability. For fair comparison with the proposed work, hybrid GA-PSO is re-evaluated using metrics, makespan, energy consumption, and failure rate.

Table 2. Simulation experiments details.

Parameter	Value
Size of population	50
Maximum iterations	100
Rate of crossover	0.8
Rate of mutation	0.05
Task set sizes	20, 40, 60, 80 tasks
Task lengths	1000–10000 MI
Processing capacity of fog nodes	200–1000 MIPS

5.1 Analysis Based on Makespan

The result for the makespan analysis is shown in **Table 3**. The analysis of the makespan for the number of tasks is presented in **Figure 3**. The makespan analysis for the proposed algorithm shows 12.44%,13.83%,9.31%,42.48%,13.36%, and 7.87% better performance than the JAYA, GA, PSO, BA, ACO, and hybrid GA-PSO algorithms. This improvement highlights the hybrid algorithm's effectiveness in balancing global exploration and local exploitation, leading to faster task completion and enhanced overall scheduling efficiency.

Table 3. Makespan analysis.

	Makespan (seconds)						
No of tasks	**RHPMO**	**JAYA**	**GA**	**PSO**	**BA**	**ACO**	**Hybrid GA-PSO**
20	20.0202	29.8849	30.4476	22.4017	60.1832	29.5487	27.22
40	30.5392	48.78	49.426	46.0141	85.3523	50.4456	40.2
60	48.97	67.88	70.8645	66.5637	128.2597	68.9067	60.55
80	82.454	87.19	89.6703	84.405	177.0676	89.2359	85.11

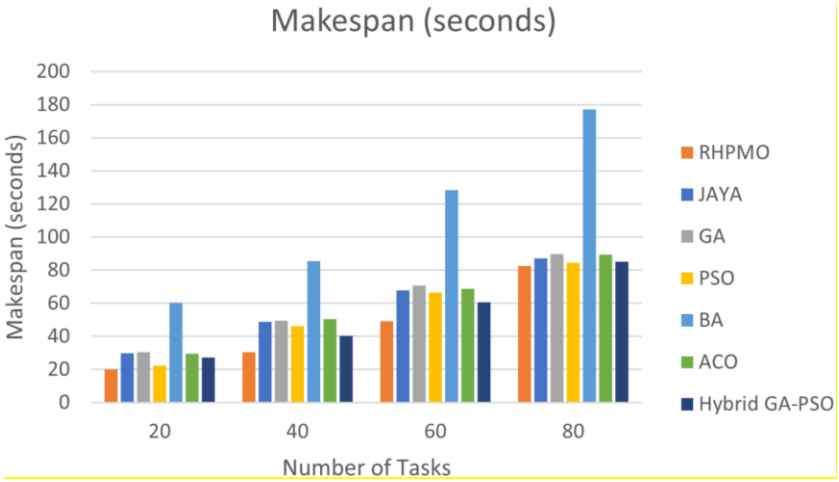

Figure 3. Comparison of the makespan result.

5.2 Analysis Based on Energy Consumption

The analysis of the energy consumption result is shown in **Table 4**. The energy consumption analysis for the number of tasks is presented in **Figure 4**. The energy analysis shows 5.92%, 11.77%, 10.95%, 29.83%, 10.51%, and 6.80% better performance for the RHPMO than the JAYA, GA, PSO, BA, ACO, and hybrid GA-PSO algorithms. This improvement indicates that RHPMO effectively balances workload distribution and resource utilization, leading to reduced energy usage and enhanced sustainability in Fog environments.

Table 4. Energy consumption analysis.

	Energy (Joule)						
No of tasks	**RHPMO**	**JAYA**	**GA**	**PSO**	**BA**	**ACO**	**Hybrid GA-PSO**
20	440.4226	498.716	611.5447	520.9331	818.1568	535.2314	500.111
40	887.1265	1100.095	1186.27	1206.792	1470.6265	1226.691	1123.8
60	1768.095	1775.56	2201.49	1792.501	3037.1278	1957.727	1780.89
80	1970.081	2328.956	2418.67	2792.071	4046.0788	2535.512	2400.9

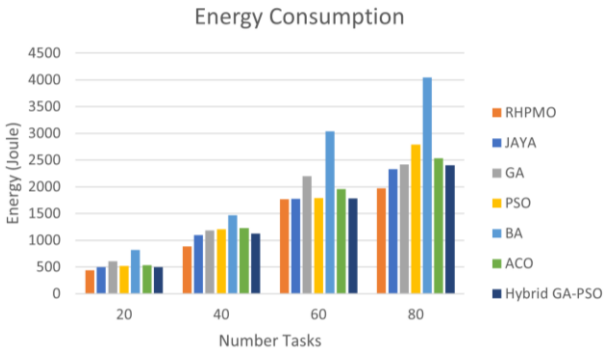

Figure 4. Comparison of energy consumption results.

5.3 Analysis based on Reliability

The analysis for the reliability result is shown in **Table 5**. The analysis of the reliability result for the number of tasks is presented in **Figure 5**. The reliability analysis shows 2.41%, 2.72%, 3.03%, 12.58%, 7.59%, and 2.10% better performance of RHPMO than the JAYA, GA, PSO, BA, ACO, and hybrid GA-PSO algorithms. This improvement demonstrates RHPMO's effectiveness in selecting reliable nodes and minimizing task failure rates, ensuring stable task execution in dynamic and distributed Fog environments.

5.4 Analysis based on Computing Time

The result for the computing time is shown in **Table 6**. The computing time analysis for the number of tasks is presented in **Figure 6**. The computation time analysis shows 11.30%, 45.81%, 50.51%, 68.91%, 61.97%, and 27.35% better performance of RHPMO than the JAYA, GA, PSO, BA, ACO, and hybrid GA-PSO algorithms. This significant reduction in computation time highlights the algorithm's capability to converge faster and execute task scheduling operations more efficiently. It makes it more appropriate for real-time Fog Computing.

Table 5. Reliability analysis.

	Reliability (%)						
No of tasks	**RHPMO**	**JAYA**	**GA**	**PSO**	**BA**	**ACO**	**Hybrid GA-PSO**
20	89	83	84	82	68	75	83
40	87	82	83	81	66	74	81
60	84	80	80	80	65	73	83
80	80	79	75	77	65	70	79

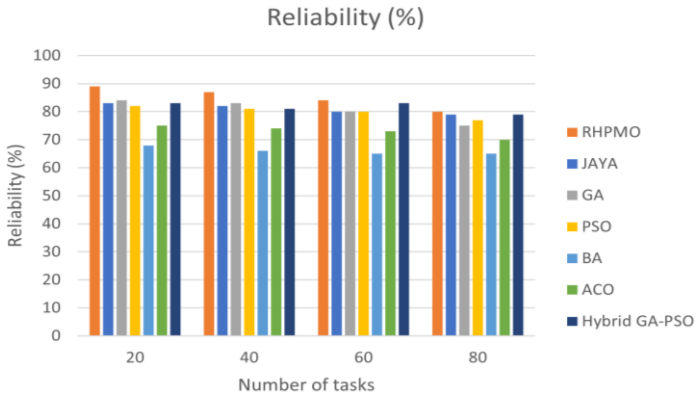

Figure 5. Comparison of reliability results.

Table 6. Computation time analysis.

	Running time (seconds)						
No of tasks	**RHPMO**	**JAYA**	**GA**	**PSO**	**BA**	**ACO**	**Hybrid GA-PSO**
20	20.93	30.22	130.17	140.23	440.45	240.47	135.2
40	90.029	130.01	278.71	290.31	590.12	390.31	150.7
60	110.15	170.85	413.19	470.2	770.47	670.29	200.34
80	288.89	308.89	550.09	650.19	970.34	870.88	407.9

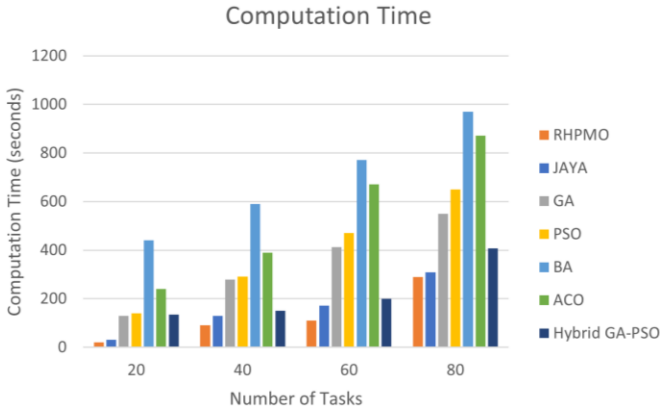

Figure 6. Analysis of computation time.

5.5 Convergence Comparison using Hypervolume and Iterations

The convergence plot for the proposed algorithm, compared to state-of-the-art algorithms, is shown in **Figure 7**. In Pareto-based multi-objective optimization, the convergence of the algorithm is plotted by using hypervolume (HV) vs iterations. HV calculates the size of the objective space dominated by the Pareto front (Deist et al., 2021). The convergence plot analysis demonstrates that the proposed RHPMO algorithm achieves the highest and fastest convergence performance compared to ACO, with 32% higher hypervolume. Furthermore, the proposed RHPMO algorithm shows significant improvement in hypervolume by 4.21%, 10.00%, 16.47%, 20.73%, and 25.32% compared to the hybrid GA-PSO, JAYA, PSO, BA, and GA, respectively. Also, it illustrates that the hybrid approach significantly enhances convergence speed compared to a single metaheuristic algorithm.

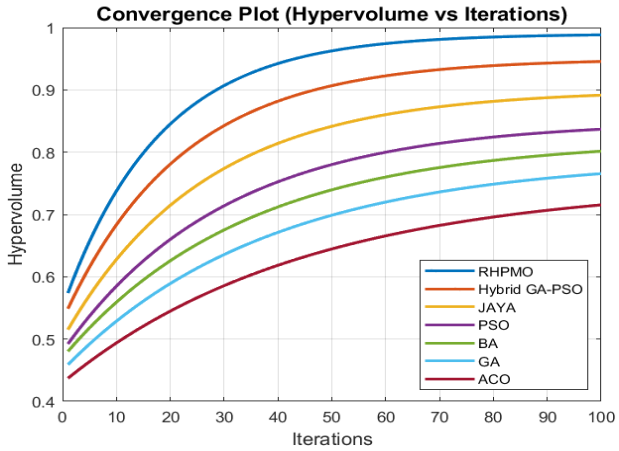

Figure 7. Convergence plot using hypervolume and iterations.

5.6 Statistical Validation of Performance Using T-Test

The proposed RHPMO algorithm was statistically analyzed against benchmark algorithms (JAYA, GA, PSO, BA, ACO, and hybrid GA-PSO) across three key objectives: Makespan, Energy, and Reliability. Paired t-test analysis shown in **Table 7**, **Table 8**, **Table 9** ($p < 0.05$) shows that RHPMO significantly reduces makespan, energy consumption compared to JAYA, GA, BA, ACO, and hybrid GA-PSO. In terms of reliability, RHPMO consistently indicates statistically significant improvement. Overall, these results confirm that RHPMO provides balanced multi-objective improvements, offering lower execution time, reduced energy consumption, and higher reliability in Fog Computing task scheduling scenarios.

Table 7. Paired t-Test Results for Makespan.

Comparison	t-Statistic	Approx. p-value
RHPMO vs JAYA	3.78	0.032
RHPMO vs GA	4.22	0.024
RHPMO vs PSO	2.24	0.011
RHPMO vs BA	5.52	0.012
RHPMO vs ACO	4.08	0.027
RHPMO vs hybrid GA-PSO	3.96	0.03

Table 8. Paired t-test results for energy consumption.

Comparison	t-Statistic	Approx. p-value
RHPMO vs JAYA	2.06	0.013
RHPMO vs GA	5.56	0.01
RHPMO vs PSO	1.74	0.018
RHPMO vs BA	2.42	0.09
RHPMO vs ACO	2.92	0.06
RHPMO vs hybrid GA-PSO	1.94	0.015

Table 9. Paired t-test results for reliability.

Comparison	t-Statistic	Approx. p-value
RHPMO vs JAYA	3.70	0.031
RHPMO vs GA	15.6	0.001
RHPMO vs PSO	5.48	0.01
RHPMO vs BA	13.87	0.002
RHPMO vs ACO	13.11	0.002
RHPMO vs hybrid GA-PSO	2.89	0.028

6. Conclusion and Future Scope

This research introduces an efficient and reliable task scheduling approach for the Fog network. A hybrid approach using the optimization algorithm RHPMO is presented to schedule tasks optimally. The proposed RHPMO algorithm is a combination of the JAYA algorithm and the Genetic algorithm for obtaining a good initial solution and achieving a fast convergence rate. A multi-objective fitness function is considered for scheduling tasks. The proposed efficient and reliable task scheduling approach schedules tasks to available Fog nodes, considering lower energy consumption, shorter makespan, and reduced failure rate. The traditional weighted sum approach for a multi-objective fitness function includes the dominated solutions. So, it degrades the performance of the one objective in comparison with the other objectives. Thus, a Pareto optimal front is used, which generates the non-dominated optimal solution. The reliable task scheduling using the proposed RHPMO shows better performance than the existing scheduling techniques. The makespan analysis for the proposed RHPMO algorithm shows 12.44%, 13.83%, 9.31%, 42.48%, 13.36%, and 7.87% better performance than the JAYA, GA, PSO, BA, ACO, and hybrid GA-PSO algorithms. The

energy analysis for the proposed RHPMO algorithm shows 5.92%, 11.77%, 10.95%, 29.83%, 10.51%, and 6.80% better performance than the JAYA, GA, PSO, BA, ACO, and hybrid GA-PSO algorithms. The reliability analysis for the proposed RHPMO algorithm shows 2.41%, 2.72%, 3.03%, 12.58%, 7.59%, and 2.10% better performance than the JAYA, GA, PSO, BA, ACO, and hybrid GA-PSO algorithms. The analysis of computation time for the proposed RHPMO algorithm shows 11.30%, 45.81%, 50.51%, 68.91%, 61.97%, and 27.35% better performance than the JAYA, GA, PSO, BA, ACO, and hybrid GA-PSO algorithms. The proposed RHPMO algorithm focuses on reliability and energy consumption, providing an efficient and sustainable solution for managing dynamic workloads in distributed IoT environments. The proposed algorithm ensures reliable task execution while minimizing energy usage, making it highly suitable for applications in smart manufacturing, healthcare, and vehicular fog networks. So, the proposed RHPMO algorithm enhances system performance, adaptability, and scalability, enabling seamless and intelligent task offloading across heterogeneous fog and edge devices in real-world IoT systems.

Future research directions include integrating additional objectives, such as cost and latency, into the optimization process. Incorporating cost-awareness can help manage financial constraints in large-scale IoT deployments, while latency optimization is essential for time-critical applications requiring real-time responses. Developing multi-objective and adaptive scheduling frameworks that balance reliability, energy, cost, and latency will further enhance the efficiency and practical applicability of task scheduling in next-generation IoT and Fog Computing systems.

Conflicts of Interest

The authors confirm that there is no conflict of interest to declare for this publication.

Acknowledgments

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

AI Disclosure

The author(s) declare that no assistance is taken from generative AI to write this article.

References

Bhasker, B., Kaliraj, S., Gobinath, C., & Sivakumar V. (2025). Optimizing energy task offloading technique using IoMT cloud in healthcare applications. *Journal of Cloud Computing Advances Systems and Applications*, *14*(1), 9. https://doi.org/10.1186/s13677-025-00733-0.

Blum C. (2024). Ant colony optimization: a bibliometric review. *Physics of Life Reviews*, *51*, 87-95. https://doi.org/10.1016/j.plrev.2024.09.014.

Chen, Q., Li, C., Chen, M., Wu, M., & Zhang, G. (2025). Digital twin assisted multi-task offloading for vehicular edge computing under SAGIN with blockchain. *IET Communications*, *19*(1), e70002. https://doi.org/10.1049/cmu2.70002.

Daghayeghi, A., & Nickray, M. (2024). Delay-aware and energy-efficient task scheduling using strength pareto evolutionary algorithm II in fog-cloud computing paradigm. *Wireless Personal Communications*, *138*(1), 409-457. https://doi.org/10.1007/s11277-024-11510-8.

Deb, K., & Jain, H. (2014). An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: solving problems with box constraints. *IEEE Transactions on Evolutionary Computation*, *18*(4), 577-601. https://doi.org/10.1109/TEVC.2013.2281535.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, *6*(2), 182-197. https://doi.org/10.1109/4235.996017.

Deist, T.M., Grewal, M., Dankers, F.J.W.M., Alderliesten, T., & Bosman, P.A.N. (2021). Multi-objective learning to predict pareto fronts using hypervolume maximization. *Machine Learning.* arXiv preprint arXiv:2102.04523.

Hashemi, S.M., Sahafi, A., Rahmani, A.M., & Bohlouli, M. (2022). GWO-SA: gray wolf optimization algorithm for service activation management in fog computing. *IEEE Access, 10,* 107846-107863. https://doi.org/10.1109/ACCESS.2022.3212439.

Hussain, M., Nabi, S., & Hussain, M. (2024). RAPTS: resource aware prioritized task scheduling technique in heterogeneous fog computing environment. *Cluster Computing*, *27*, 13353-13377. https://doi.org/10.1007/s10586-024-04612-2.

Ismail, A.H., Hartono, N., Zeybek, S., & Pham, D.T. (2020). Using the Bees algorithm to solve combinatorial optimisation problems for TSPLIB. *IOP Conference Series Materials Science and Engineering*, *847*(1), 012027. https://doi.org/10.1088/1757-899X/847/1/012027.

Katoch, S., Chauhan, S.S., & Kumar, V. (2021). A review on genetic algorithm: past, present, and future. *Multimedia Tools and Applications*, *80*(5), 8091-8126. https://doi.org/10.1007/s11042-020-10139-6.

Kumar, V., & Sahni, R. (2020). Dynamic testing resource allocation modeling for multi-release software using optimal control theory and genetic algorithm. *International Journal of Quality & Reliability Management*, *37*(6-7), 1049-1069. https://doi.org/10.1108/IJQRM-09-2019-0296.

Kumar, V., Sarkar, B., Sharma, A.N., & Mittal, M. (2019). New product launching with pricing, free replacement, rework, and warranty policies via genetic algorithmic approach. *International Journal of Computational Intelligence Systems*, *12*(2), 519-529. https://doi.org/10.2991/ijcis.d.190401.001.

Lera, I., & Guerrero, C. (2024). Multi-objective application placement in fog computing using graph neural network-based reinforcement learning. *The Journal of Supercomputing*, *80*(19), 27073-27094. https://doi.org/10.1007/s11227-024-06439-5.

Madni, S.H.H., Faheem, M., Younas, M., Masum, M.H., & Shah, S. (2024). Critical review on resource scheduling in IaaS clouds: taxonomy, issues, challenges, and future directions. *The Journal of Engineering*, *2024*(8), e12420. https://doi.org/10.1049/tje2.12420.

Nkenyereye, L., Lee, B.G., & Chung, W.-Y. (2025). Functionality-aware offloading technique for scheduling containerized edge applications in IoT edge computing. *Journal of Cloud Computing*, *14*(1), 13. https://doi.org/10.1186/s13677-025-00737-w.

Pradhan, S.K., Kumar, A., & Kumar, V. (2023). An effort allocation model for a three stage software reliability growth model. In *Predictive Analytics in System Reliability* (pp. 263-282). IEEE. Cham, Springer International Publishing. https://doi.org/10.1007/978-3-031-05347-4_17.

Rao, M., & Qin, H. (2024). Enhanced hybrid equilibrium strategy in fog-cloud computing networks with optimal task scheduling. *Computers, Materials & Continua, 79*(2), 2647-2672. https://doi.org/10.32604/cmc.2024.050380.

Rao, R.V. (2016). Jaya: a simple and new optimization algorithm for solving constrained and unconstrained optimization problems. *International Journal of Industrial Engineering Computations*, *7*(1), 19-34.

Saad, M., Enam, R.N., & Qureshi, R. (2024). Optimizing multi-objective task scheduling in fog computing with GA-PSO algorithm for big data application. *Frontiers in big Data*, *7*, 1358486. https://doi.org/10.3389/fdata.2024.1358486.

Sehgal, N., Bansal, S., & Bansal, R.K. (2025). Energy and security-aware task scheduling in fog computing: a comparative analysis of scheduling algorithms using IoT. *Procedia Computer Science*, *252*, 430-439. https://doi.org/10.1016/j.procs.2025.01.002.

Singh, S.P., Kumar, R., Sharma, A., Reddy, S.R., & Vashisht, P. (2022). Simulation and emulation tools for fog computing. *Recent Advances in Computer Science and Communications, 15*(3), 315-322. https://doi.org/10.2174/2666255813999201002152003.

Sui, X.-F., Wang, J.-S., Zhang, S.-H., Zhang, S.-W., & Zhang, Y.-H. (2025). Multi-strategy fusion mayfly algorithm on task offloading and scheduling for IoT-based fog computing multi-tasks learning. *Artificial Intelligence Review*, *58*(5), 144. https://doi.org/10.1007/s10462-025-11145-6.

Vashisht, P., & Kumar, V. (2020). Agent based optimized réplica management in data grids. *Revista Investigación Operacional, 41*(2), 232-249.

Vashisht, P., & Kumar, V. (2022). A cost effective and energy efficient algorithm for cloud computing. *International Journal of Mathematical, Engineering and Management Sciences, 7*(5), 681-696. https://doi.org/10.33889/IJMEMS.2022.7.5.045.

Vashisht, P., Bajaj, S.B., & Narang, A. (2024). Energy-efficient fog computing: a review and future directions. *International Journal of Innovative Research in Computer Science and Technology*, *12*(2), 135-139. https://doi.org/10.55524/ijircst.2024.12.2.24.

Vashisht, P., Kumar, V., Kumar, R., & Sharma, A. (2019a). Optimizing replica creation using agents in data grids. In *2019 Amity International Conference on Artificial Intelligence* (pp. 542-547). IEEE. Dubai, United Arab Emirates. https://doi.org/10.1109/AICAI.2019.8701244.

Vashisht, P., Kumar, V., Kumar, R., & Sharma, A. (2019b). Optimization of replica consistency and conflict resolution in data grid environment. *International Journal of Mathematical, Engineering and Management Sciences*, *4*(6), 1420. https://dx.doi.org/10.33889/IJMEMS.2019.4.6-112.

Wang, D., Tan, D., & Liu, L. (2018). Particle swarm optimization algorithm: an overview. *Soft Computing*, *22*(2), 387-408. https://doi.org/10.1007/s00500-016-2474-6.

Zhang, Q., & Li, H. (2007). MOEA/D: a multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, *11*(6), 712-731.

Zitar, R.A., Al-Betar, M.A., Awadallah, M.A., Doush, I.A., & Assaleh, K. (2022). An intensive and comprehensive overview of JAYA algorithm, its versions and applications. *Archives of Computational Methods in Engineering*, *29*(2), 763-792. https://doi.org/10.1007/s11831-021-09585-8.