# Hybrid scheduling algorithms in cloud computing: a review

**Neeraj Arora[1], Rohitash Kumar Banyal[2]**
[1]School of Science and Technology, Vardhman Mahaveer Open University, Kota, India
[2]Department of Computer Science and Engineering, Rajasthan Technical University, Kota, India

**Article Info**

***Article history:***

Received Mar 3, 2021
Revised Jul 18, 2021
Accepted Aug 9, 2021

***Keywords:***

Cloud computing
Hybrid algorithms
Metaheuristic algorithms
Optimization

**ABSTRACT**

Cloud computing is one of the emerging fields in computer science due to its several advancements like on-demand processing, resource sharing, and pay per use. There are several cloud computing issues like security, quality of service (QoS) management, data center energy consumption, and scaling. Scheduling is one of the several challenging problems in cloud computing, where several tasks need to be assigned to resources to optimize the quality of service parameters. Scheduling is a well-known NP-hard problem in cloud computing. This will require a suitable scheduling algorithm. Several heuristics and meta-heuristics algorithms were proposed for scheduling the user's task to the resources available in cloud computing in an optimal way. Hybrid scheduling algorithms have become popular in cloud computing. In this paper, we reviewed the hybrid algorithms, which are the combinations of two or more algorithms, used for scheduling in cloud computing. The basic idea behind the hybridization of the algorithm is to take useful features of the used algorithms. This article also classifies the hybrid algorithms and analyzes their objectives, QoS parameters, and future directions for hybrid scheduling algorithms.

***Corresponding Author:***

Neeraj Arora
School of Science and Technology, Vardhman Mahaveer Open University
Kota, Rajasthan, India
Email: narora@vmou.ac.in, neeraj.arora0007@gmail.com

## 1. INTRODUCTION

Cloud computing provides applications and services on demand through the internet. It is gaining popularity in information technology due to advancements like hiding and abstraction of complexity, virtualizes resources, and efficient use of distributed resources. The cloud computing architecture is divided into two components front-end and the back-end. The elements of both components are loosely coupled. The architecture's front-end provides the applications and interfaces for the user to use the cloud service. The back-end consists of all the resource needs in cloud computing, like data storage, processing elements, and applications. The back-end is also responsible for providing manageability and security mechanism needed in a cloud computing environment. The network or the internet does the communication between the back-end and the front-end. The abstract architecture of cloud computing is shown in Figure 1.

Cloud computing is a distributed paradigm where the IT resources are available on-demand over the internet based on a pay-per-use billing model [1]. The services delivered by the cloud is broadly categorized as software as a service (SaaS) for end-user APIs like salesforce.com, platform as a service (PaaS) for a run-time environment like Windows Azure, Google app engine, and infrastructure as a service (IaaS) for hardware resources like Google Cloud, Amazon EC2 [2]. The service of cloud computing is offered using virtualization. The physical machines are virtually divided into several virtual machines (VM) according to several parameters and resource types, like operating systems, processing elements, memory, or storage, as per the user requirement. The user's tasks are run on virtual machines. The virtual machines can share the physical machine's hardware and resources to achieve parallel and distributed computing [3]. It is essential to have a good scheduling algorithm for assigning the users tasks to the appropriate resources to optimize the various QoS parameters like cost, time, and energy consumption.

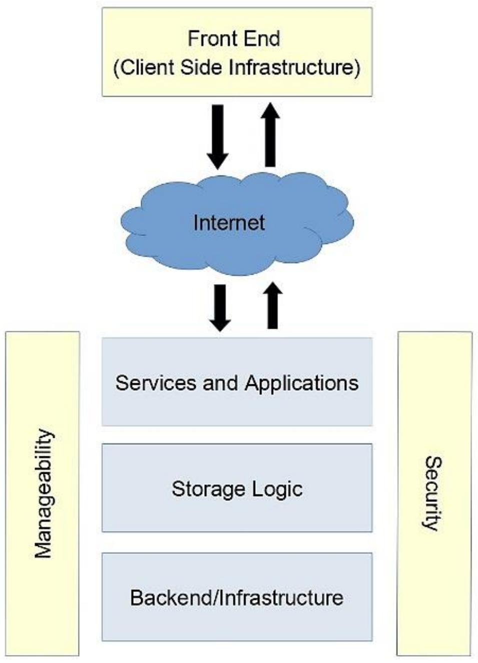


Figure 1. Cloud architecture

Scheduling is one of the prominent issues in cloud computing. The scheduling in cloud computing is different from the traditional system because of cloud computing features like elasticity, pay-per-use model, on-demand, multi-tenant. The scheduling has two phases resource provisioning and task scheduling. The resources are selected based on the quality of service (QoS) parameters and then select the suitable VM instance for those tasks in resource provision. Finally, the desired VM instance is allotted to the available hosts or physical machines. The resource provisioning is significantly related to balance the load. The optimal order of the tasks is to find according to the scheduling objectives in task scheduling.

Although several traditional algorithms like first come first serve (FCFS), and shortest job first (SJF), were proposed but not performed well as these are developed for the traditional computation. The scheduling in cloud computing is an NP-hard problem. So, the meta-heuristic algorithms like a genetic algorithm (GA), and particle swarm optimization (PSO), are a good option, where you find the near-optimal solution. But with the increase in the problem space, the meta-heuristic algorithms showed some limitations like stuck in the local optimal solution. With the further rise in the cloud's complex scheduling problem, various hybrid algorithms were proposed, combining two or more algorithms and taking the advantages of involved algorithms to efficiently solve the problem.

The major of review papers was focused on the resource scheduling based on heuristics and meta-heuristics methods like in [1], [4]-[6]. Livny *et al.* [7] the review was based on workflow scheduling according to the objective and execution model. Some other papers like [8]-[10] were also focused on workflow scheduling and develop a taxonomy of workflow management and scheduling. In this paper, we consider the hybrid scheduling algorithms in cloud computing for review. We discussed various objectives involved in optimization, the simulator used and classifies according to the involved algorithms.

The rest of the article is organized as follows. In section 2, we gave a general introduction to the scheduling problem, different types of scheduling problems in cloud computing. The various objectives or QoS parameters found in the literature are discussed in section 3 with their mathematical definition. Then, we classify the hybrid algorithms according to the involved algorithms. We also gave an overview of these hybrid algorithms along with the research directions and issues related to scheduling in cloud computing. The complete details are given in section 4. In section 5, we discuss the trends and future direction of hybrid scheduling algorithms in cloud computing. The paper is ended with conclusion as mentioned in section 6.

## 2. SCHEDULING PROBLEM

The scheduling architecture of cloud computing is illustrated in Figure 2. The users submitted the task to the scheduling server. The scheduler assigns the appropriate resources for task execution using scheduling algorithms according to the user requirements. The resources are available on the hosts or physical machines of the cloud provider. Each host is divided into various virtual machines. When the task completes its execution, the result will return to the respective user.

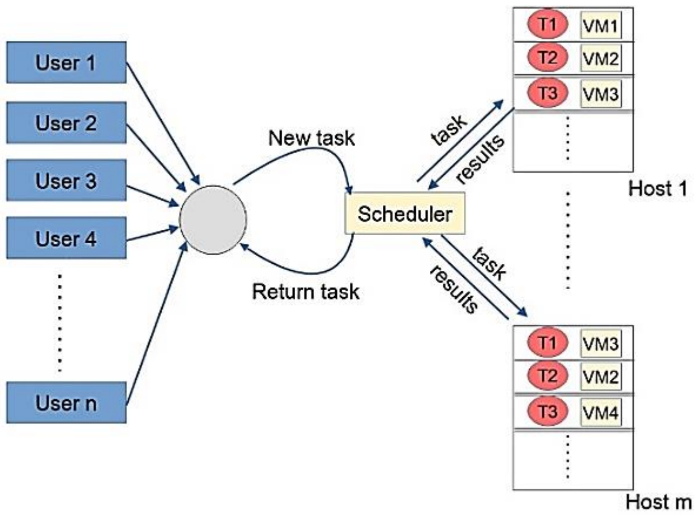


Figure 2. Scheduling in cloud computing environment

The scheduler generates the mapping between tasks and virtual machines or virtual machines and hosts for allotment according to the performance parameters. Figure 3 shows a sample mapping of $n$ number of tasks to the $m$ number of the virtual machines, then there are $m^n$ combination is possible if brute force algorithm is used. Similar is the case with virtual machines and hosts. The scheduling is considered a complex problem, and the solution is not completed in polynomial time [11]. It is good to find a near-optimal solution to the scheduling problem with the help of meta-heuristic algorithms.

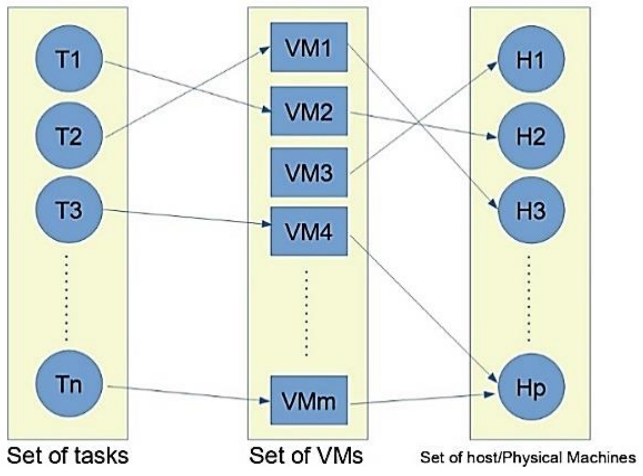


Figure 3. Mapping among tasks, VMs and hosts

The population-based meta-heuristic scheduling algorithms take a set of solutions known as population. Each member of the population represents the solution to the problem. For each iteration, the solutions will improve to move towards the near-optimal results. Figure 4 shows the sample encoding of a population of $k$ members, $n$ tasks, and $m$ virtual machines concerning the task scheduling problem. In task scheduling algorithms, each solution has the mapping of tasks and virtual machines.

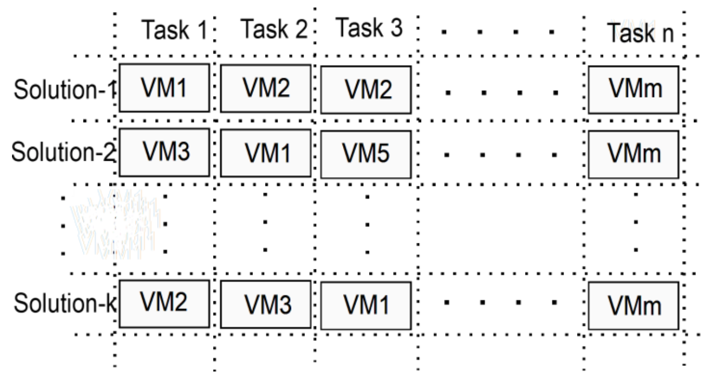


Figure 4. Encoding of scheduling problem in cloud computing

### 2.1. Basic scheduling problem

The scheduling problem in cloud computing has two parts resource provisioning and task scheduling. The task scheduling problem is considered a mapping problem. Several tasks are mapped to the various available virtual machine such that it optimizes the objective considered by the proposed algorithm. Mathematically, the problem can be seen as a mapping $f(VM, T): T \rightarrow VM$, set of virtual machines VM is $\{VM_1, VM_2, \ldots, VM_m\}$ and set of tasks T is $T = \{T_1, T_2, \ldots, T_n\}$. Each virtual machine $VM_i$ has its characteristics like id, clock speed in MIPS, RAM size, and several processors. Each task $T_j$ also has its characteristics like id, length, arrival time, and file size. Using these characteristics of tasks and virtual machines, one needs to design the scheduling algorithm for generating the mapping so that the targeted objectives will be optimized. Whereas, in resource provisioning, the VMs are allotted to available physical machines according to their characteristics. The basic intention of resource provisioning is to balance the load among the available servers for better functioning [5]. Mathematically, the problem can be seen as a mapping $f(PM, VM): VM \rightarrow PM$, set of physical machines or hosts $PM = \{PM_1, PM_2, \ldots, PM_n\}$ and set of virtual machines VM is $\{VM_1, VM_2, \ldots, VM_m\}$ as shown in the Figure 3.

### 2.2. Scheduling problem with constraints

The scheduling can be done by considering some constraints, where the values of objectives are lying between some predefined threshold limits. The most common are the budget and deadline constraints. In budget constraints, the user's tasks need to be complete in some predefined threshold of the cost, whereas in deadline constraint, the makespan value is considered [12]. Mathematically, the scheduling problem with constraints can be described using (1) and (2):

$$\min/\max \mathrm{f(x)} = \mathrm{x} \tag{1}$$

subject to:

$$\mathrm{x} \in \text{ objectives} \mid \{\mathrm{y} \leq \mathrm{x} \leq \mathrm{z}\} \tag{2}$$

where $f(x)$ is the fitness function (discussed in section 3) $x$ are the objective values and the value of $x$ should lies in between predefined threshold values $y$ and $z$

### 2.3. Workflow scheduling problem

The workflow scheduling is scheduling with precedence constraint, where the order of arrival of tasks is considered [8]. This scheduling is also called dependent task scheduling. The workflow can be represented using directed acyclic graph (DAG), where the nodes in the DAG represent the tasks (T), and the edges (E) joining the nodes represent the dependency between the tasks. A sample workflow is shown in Figure 5, containing six tasks $\{T1, T2, T3, T4, T5, T6\}$. The tasks $T1$ and $\{T4, T5, T6\}$ are the entry and exit tasks, respectively. Each edge of the DAG shows the dependencies between the tasks. For example, $T2$ executed after $T1$ is shown by the paired set $\{T1, T2\}$.

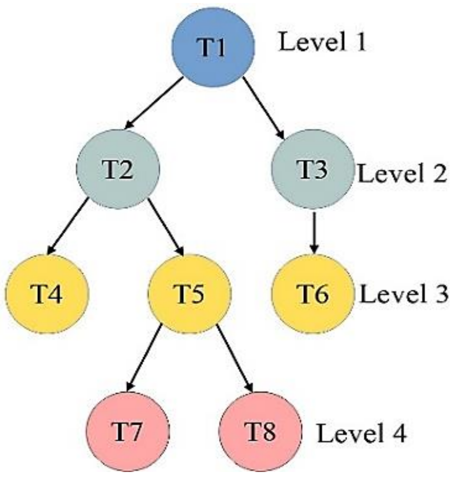


Figure 5. A simple workflow

In recent years, the proposed scheduling algorithms were tested in the real work scientific workloads. These workloads express the complex computation problems that can be solved using distributed and parallel computing [13]. There are several types of scientific workflow like CyberShake, Inspiral, Montage, and Sipth, available by Pegasus [14]. The CyberShake workflow analyzes the disaster modeling and prediction of the particular geographical location by designing the hazard maps [15]. For studying and analyzing the gravitational waveform, the inspiral workflow is used [12]. The montage workflow is used by NASA in astrophysics to create the custom mosaics of the sky by taking multiple input images [16]. Sipht workflow application is maintained by Harvard to use in the field of bioinformatics [17].

## 3. FITNESS FUNCTION

The fitness function described the targeted objectives to be optimized using the proposed scheduling algorithm [9]. If only one objective is present in the fitness function, then it is a single objective function, also known as objective function [7]. Similarly, if it contains two-objective, it is known as a bi-objective function, and it has more than two objectives and is then known as a multi-objective function. Generally, the fitness function can be represented as (3),

$$f(x_1, x_2, \dots x_n) = \alpha_1 \times x_1 + \alpha_2 \times x_2 + \dots + \alpha_n \times x_n \tag{3}$$

where $x_1, x_2, \dots x_n$ are the objectives and $\alpha_1, \alpha_2, \dots\dots \alpha_n$ are the weight assigned for each objectives. Figure 6 shows the different objectives considered in the existing scheduling algorithms. The different objectives considered in the literature will be explained in the following sub-sections.

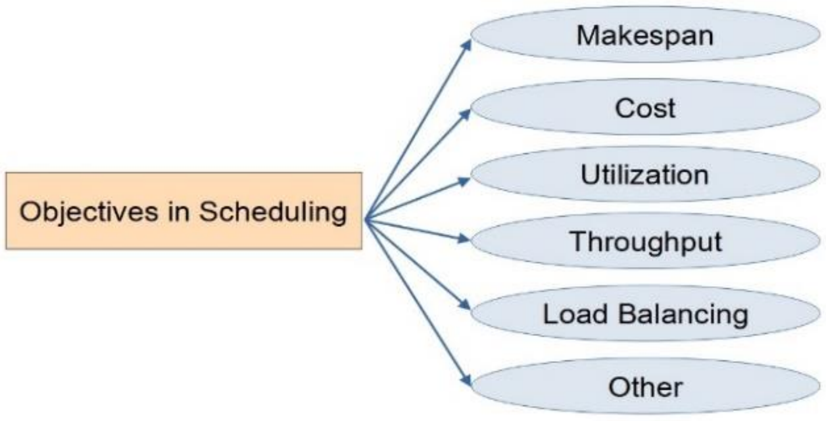


Figure 6. Objectives of scheduling algorithms

### 3.1. Makespan

The makespan is the maximum completion time taken by virtual machines. In other words, the time taken when all the tasks are finished its execution on the virtual machine is the makespan [18]. Mathematically, the makespan can be derived using (4),

$$Makespan = max\{CT_j \,|j = 1,2,\ldots m\} \tag{4}$$

where $CT_j$ is the completion time of virtual machine $VM_j$. The completion time is the maximum execution time of tasks. In case tasks are dependent, then the waiting time of tasks is also considered. The completion time $CT_j$ is depicted in (5).

$$CT_j = \begin{cases} \max(ET_i) & \text{iff } pred(T_k) = \emptyset \\ max(WK_i + ET_i) & \text{iff } pred(T_k) \neq \emptyset \end{cases} \tag{5}$$

The waiting time of task $T_i$ is the maximum completion time of all the predecessor tasks of a workflow, as shown in (6). The execution time of a task can be calculated using (7),

$$WK_i = \begin{cases} 0 & \text{iff } pred(T_k) = \emptyset \\ \max(CT_k) & \text{iff } pred(T_k) \neq \emptyset \end{cases} \tag{6}$$

$$ET_i = \frac{SZ_{Task}}{Num(PE_i) \times PE_{Unit}} \tag{7}$$

where $SZ_{Task}$ is the size of the task in a million instruction (MI), $Num(PE_i)$ is the number of core assigned to the VM, size of each core in MIPS.

### 3.2. Cost

The cost is the crucial objective to be optimized, as cloud computing follows a pay-as-you-go billing scheme, using an efficient scheduling algorithm. Generally, the cloud service providers charges for some specific time interval based on the amount of storage. Execution costs, communication costs, and storage costs are considered in cloud computing. VM's total execution cost is the cost charged of VM per unit interval and the execution time of tasks on that VM. Mathematically, the total execution cost (TEC) of VM is shown in (8).

$$TEC_{vm} = \sum_{j \in vm, j=1}^{k} \frac{ET_{ki}}{\tau} \times CO_i : i \in I_i \tag{8}$$

Similarly, the total execution cost of workflow W is given in (9) [7],

$$TEC_W = \sum_{j \in W, j=1}^{k} \frac{ET_{ki}}{\tau} \times CO_i : i \in I_i \tag{9}$$

where $CO_i$ is the cost of type-i VM instance for a unit time in the cloud data center. $\tau$ is the time period for which the user uses the resources. $ET_{ki}$ is the execution time of task $T_k$ by type-i VM instance.

When the task is allotted on the different machines, the communication cost is also included. It is also known as data transfer cost and mathematically shown in (10) [19],

$$CC_{ij} = \frac{data_{(i,j)}}{BW_{(i,j)}} \tag{10}$$

where $CC_{ij}$ is the communication cost between tasks $t_i$ and $t_j$. $data_{(i,j)}$ is the length of the output file of task $t_i$ and $BW_{(i,j)}$ is the bandwidth between the resources. If the tasks are on the same machine, the communication cost $CC_{ij}$ become zero. Other types of costs include the cost of storing the tasks on the resource. This type of cost depends upon the size of the task allotted to the resources, the cloud provider charge, according to the volume of data files stored onto the machines.

### 3.3. Utilization

The VM utilization is the ration of processing time and makespan and expressed using (11),

$$Utilization_{vm_j} = \frac{\sum_{i=1}^{n} PT_{ij}}{makespan} \tag{11}$$

where, $PT_{ij}$ is the processing time of task $T_i$ on $VM_j$.

### 3.4. Throughput

The throughput is calculated as the number of task instructions completed in a unit time [20]. The throughput of $VM_k$ is calculated as per (12),

$$throughput = \frac{\sum_{i=1}^{n} SZ(task_i)}{CT_k} \quad (12)$$

where $CT_k$ is the completion time of the $VM_k$ and $SZ(task_i)$ is the size of $task_i$ in million instructions.

### 3.5. Load balancing

The Load balancing is measured using the degree of imbalance as mentioned in the (13) [21],

$$DI = \frac{ET_{max} - ET_{min}}{ET_{avg}} \quad (13)$$

where $T_{max}$, $ET_{min}$ and $ET_{avg}$ are maximum, minimum, and average total execution times among all VMs, respectively. ET can be find using (13).

### 3.6. Other

There are other QoS parameters mentioned in [5]:

- Scalability: This metrics is used to check the performance of an algorithm in the increasing number of nodes.
- Fault tolerance: It is used to check the capability of the algorithm working in under some failure like links, and processing units.
- Energy consumption: It shows the total amount of energy consumed by the system. This parameter is used to avoid overheating of a particular node by using an efficient energy-saving load balancing algorithm.

## 4. HYBRID SCHEDULING ALGORITHMS

From the literature, we found that most of the proposed hybrid algorithms were the combination of the popular meta-heuristic algorithms like particle swarm optimization (PSO) [22], genetic algorithms (GA) [23], ant colony optimization (ACO) [24], or its variants. We classify the hybrid algorithm according to the involved algorithms during the hybridization, as shown in Figure 7. The following sub-section will review the hybrid algorithms are per the classification.

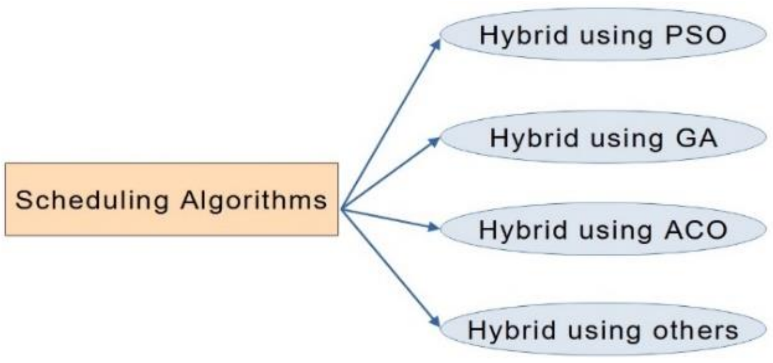


Figure 7. Classification of hybrid scheduling algorithms in cloud computing

### 4.1. Hybrid using PSO

Shirvani [25] proposed a hybrid discrete particle swarm optimization (HDPSO) algorithm, which was designed using a discrete particle swarm algorithm and hill-climbing algorithm. The basic idea was to use a hill-climbing algorithm for local search to balance between exploration and exploitation. The objective was to minimize the makespan. The HDPSO performed better in schedule length ratio (SLR), Speed-Up, and efficiency than the other exiting heuristics approaches and meta-heuristics approaches like GA, PSO. In the near future, the authors will consider the makespan and cost objectives for optimizing the scheduling.

Domanal *et al.* [26] presented the algorithm, which was the hybrid version of modified PSO and Modified Cat Swarm Algorithm; the algorithm reduced the average response time. It increased resource utilization compared with round-robin (RR), modified PSO (MPSO), modified cat swarm optimization (MCSO), ACO, and Exact algorithm. The simulator used was PySim to validate the results. The authors will consider the dynamic scheduling in which tasks will be entering the cloud with different inter-arrival times in future work.

Jena *et al.* [27] was consider modified particle swarm optimization (MPSO) and improved the Q-learning algorithm to propose a hybrid algorithm named QMPSO for load balancing in cloud computing. The velocity in MPSO is adjusted using the best action generated by the improved Q-learning algorithm. The objectives of the algorithm were to optimize the makespan, throughput, and energy utilization. The results were validated using the CloudSim 3.0.3 simulator. The results showed a reduction in the waiting time of the tasks concerning MPSO and Q-learning algorithms. Independent tasks were considered for the implementation in the near future as the authors considered only the dependent tasks.

Firefly algorithm was combined with an improved PSO algorithm (IPSO) to make another hybrid algorithm named IPSO-Firefly algorithm [28]. The basic idea was to use the firefly algorithm for providing the initial solution of IPSO. The IPSO-Firefly algorithm performed well in terms of average load, average turnaround time, average response time compared to RR, FCFS, SJF, GA, IPSO, PSO, Firefly. The proposed algorithm converged fast in comparison to other state-of-art algorithms. The simulator used was MATLAB software for carried out work.

Mansouri *et al.* [29] proposed a hybrid task scheduling algorithm using the fuzzy system and modified particle swarm optimization techniques. The objectives were to minimize makespan and maximize resource utilization. The proposed algorithm (FMPSO) increased the exploration ability by introducing four modified velocity updating methods. Further, the crossover and mutation operator is combined with the PSO algorithm for more improvement. The results showed improvement in the degree of imbalance, makespan, execution time, efficiency, and improvement ratio compared to standard-PSO (SPSO), standard-GA (SGA), modified-PSO (MPSO), and standard GA with fuzzy theory (FUGE) algorithm. The precedence of tasks and fault tolerance parameter will be considered for further study in the future.

Dordaie and Navimipour [30] used a hill-climbing algorithm for local search to modify the existing PSO algorithm and proposed a hybrid particle swarm optimization and hill-climbing algorithm for task scheduling in the cloud environment. The aim was to reduce the makespan of task scheduling. The authors used C# in an Azure cloud environment to generate the experiment results. The proposed algorithm and some existing algorithms were tested under random and scientific DAG to show the efficiency in terms of makespan and found beneficial compared to other algorithms. In the future, real-world DAG was used to test the proposed algorithm by considering load balancing.

Jana and Poray [31] used two popular bio-inspired meta-heuristics algorithms, genetic and PSO algorithm, for task scheduling in cloud computing. The objective is to reduce the response time of the VM. The authors used the CloudSim simulator to simulate the proposed algorithm and evaluated with Max-Min and minimum execution time algorithm. The results showed that the proposed algorithm minimizes the waiting time and response time. The author will consider dynamic task allocation, minimization of electric cost, and internal communications into account in the near future.

Verma and Kaushal [32] proposed a hybrid multi-objective particle swarm optimization (HPSO) for scientific workflow scheduling problems in the IaaS cloud environment. The proposed algorithm (HPSO) used a multi-objective particle swarm optimization algorithm with a list-based heuristic i.e., budget and deadline constrained heterogeneous earliest finish time (BDHEFT). The fitness function was composed of two conflicting objectives makespan and cost under the deadline and budget constraints. The authors extend the functionality of CloudSim for workflow scheduling for simulation and analysis of results. The simulation results showed that the proposed HPSO converged fast and has a uniform spacing among the solutions compared with other state-of-art multi-objective meta-heuristics like non-dominated sort genetic algorithm-II (NSGA-II), multi-objective PSO (MOPSO), and fuzzy dominance sort based discrete PSO ($\epsilon$-FDPSO). Some other QoS constraints, like reliability, trust management, VM migration, will be considered for future work. Concepts like neural networks, and fuzzy logic can be tested to improve the proposed algorithm.

A hybrid heuristic based on particle swarm optimization (PSO) and gravitation search algorithms (GSA) was proposed in [33] for workflow scheduling in the cloud environment. The simulation was done using the CloudSim toolkit with the amazon EC2 referencing price. The proposed algorithm notifies the significant reduction in cost compared to existing non-heuristic, PSO, and GSA algorithm under deadline constraint. One can improve the proposed algorithm by choosing the VM number according to the historical data in future work.

Authors proposed Two-hybrid algorithms in [34] named best fit PSO (BFPSO) and PSO-tabu search (PSOTS). The basic idea of BFPSO was to initialize the PSO using the Best fit algorithm instead of random values. In PSOTS, the local search ability was improved by apply Tabu search (TS) to PSO. Both the proposed hybrid algorithms performed better in terms of makespan, cost, and resource utilization compared to standard PSO when simulated using the CloudSim toolkit. The authors plan to improve the standard PSO using other greedy methods in the future.

George [35] proposed a hybrid particle swarm optimization, and multi objective bat algorithm (PSO-MOBA) algorithm for minimizing the profit using resource scheduling, low power consumption in cloud computing. The proposed algorithm was the combination of PSO and multi-objective bat algorithm (MOBA) and resolved the global convergence problem. In the future, other factors will be included to reduce the cost and maximize the profit.

### 4.2. Hybrid using GA

The algorithm hybrid electro search with a genetic algorithm (HESGA) was proposed by Velliangiri *et al.* [36] to optimize the results in terms of makespan, execution time, and cost. The proposed algorithm takes advantage of both the involved algorithms. The GA was adequate to achieve the local optimization results while the electro search algorithm best achieved the global results. The proposed algorithm is tested using CloudSim 3.0.3 simulator and finds that the proposed algorithms outperform that GA, ES, ACO, and hybrid particle swarm optimization genetic algorithm (HPSOGA). The dependency of tasks and scientific workflow will evaluate other parameters like the degree of imbalance and energy efficiency in the future works.

Aziza and Krichen [37] consider the dependency of tasks and proposed a hybrid algorithm with two popular Heterogeneous earliest finish time (HEFT) and GA algorithms, named hybrid of HEFT and GA (HEFT-GA). The initial population of GA is initialized using the result generated by the HEFT algorithm. The HEFTGA experimented with real work scientific workflows like Montage, Cybershake, Epigenomics, Laser interferometer gravitational-wave observatory (LIGO), and sRNA identification protocol using high-throughput technologies (SIPHT) to optimize the results in terms of execution time and execution cost using the workflowsim simulator. Power consumption will be considered in future work.

The Genetic algorithm was combined with the gravitational search algorithm to overcome the drawback of gravitational search by storing the best particle position [38]. Chaudhary and Kumar [38] proposed a hybrid genetic-gravitational search algorithm (HG-GSA) to reduce the total cost. The proposed algorithm reduces the total cost and increases utilization compared with PSO, Cloud-GSA, and linear improved GSA (LIGSA-C) approaches. The simulator used was CloudSim 3.0.3. In the future, the authors will focus on the new hybrid algorithm based on the bio-inspired heuristics. The authors will also work on the concepts based on the center of mass-based crossover and diversity-based crossover techniques to reduce costs in the future.

Natesan and Chokkalingam [39] proposed a hybrid multi-objective task scheduling algorithm combining whale optimization algorithm (WOA) with GA to optimize makespan and cost, enactment amelioration rate (EAR). The basic idea was first to use WOA and update the worst chromosome using the GA algorithm's operations like crossover and mutation. The proposed algorithm, named Whale Genetic Optimization Algorithm, was simulated using a CloudSim simulator and found that it gave optimized results in comparison to other standard algorithms like first come first serve, Min-Min, Max-Min algorithm. The proposed algorithm was also performed better in comparison to standalone GA and WOA. The authors will consider other parameters like energy consumption, security, and reliability in the near future.

Srichandan *et al.* [40] proposed a hybrid multi-objective genetic and bacterial foraging algorithm named MHBFA, for task scheduling in cloud computing. The proposed algorithm takes account of the advantages and disadvantages of the involved algorithms. The genetic algorithm has low local search capability but excellent in global search while bacterial foraging was good in local search but deficient in global search. The two objectives, makespan, and energy, were considered for the minimization of the fitness function. The simulation experiments were carried out using Matlab R2013a. The proposed algorithm minimizes the makespan and reduces energy consumption in comparison with GA, PSO, bacterial foraging algorithm (BFA). The same work can be carried out under dependent tasks. The further improvement in the proposed algorithm's convergence rate was possible by reducing extra timing taken by the crossover and mutation operations in the future.

Manasrah and Ali [41] used the two most popular bio-inspired algorithms, the genetic algorithm and the particle swarm optimization algorithm, to propose a new hybrid GA-PSO algorithm for works flow scheduling in cloud computing. The proposed algorithm started with the random population, then applied the genetic algorithm for the first half of the iteration. The solutions generated by the genetic algorithm were initiated as the initial population of PSO. For the latter half of the iteration, PSO was run to create the optimal solution. The proposed algorithm simulates using the WorkflowSim simulator under the real scientific workflows like Montage, CyberShake, Epigenomics, and LIGO. The simulation results showed the proposed algorithm outperformed existing state-of-art algorithms like GA, PSO, HSGA, Workflow Scheduling for public cloud using genetic algorithm (WSGA), and Min-min based time and cost tradeoff (MTCT) algorithm in total execution time and total execution cost. The same work can be extended by considering more than one data center in a heterogeneous environment. The same would be improved by considering the dynamic workflow that can change the tasks' characteristics during the run-time.

Loheswaran and Premalatha [42] proposed a hybrid algorithm combining the genetic algorithm (GA) and invasive weed optimization (IWO) named GA-IWO. The authors improved the genetic algorithm as GA has premature convergence and complexity by combining with the IWO algorithm. The authors take schedule length as an objective and found that the proposed algorithm lowers the average schedule length compared to other existing algorithms.

Another hybrid evolutionary workflow scheduling algorithm named linewise earliest finish time (LEFT) was proposed by Nasonov *et al.* [43]. The proposed algorithm used the HEFT and GA algorithm's best characteristics and used an alternative for existing HEFT in the initial population generation for GA. The objective taken into account was makespan. Simulation experiment results in computational environment simulator show that the proposed algorithm works better than the traditional HEFT algorithm in a dynamic heterogeneous distributed computational environment. The work can be extended to a multi-heuristics scheme where different heuristics will be used parallelly in tournament mode for better results.

Kamalinia and Ghaffari [11] proposed a hybrid meta-heuristic task scheduling method combining the Genetic and differential evolution (DE) algorithms. The proposed novel hybrid genetic along with the DE algorithm. The basic idea was to apply GA, and the solution generated by GA was the initial population of the DE algorithm. The simulation was done using Visual Studio 2013 and the C#.net programming language. The proposed algorithm improves resource efficiency and minimizes the makespan compared with HEFT (UpRank, DownRank, and LevelRank) and Binary genetic algorithm (BGA). The proposed novel approach also reduces the critical path and reduce communication cost among the processors. In future work, the hybridization can be done using other meta-heuristic algorithms, and the performance can be analyzed using simulation results.

Ahmad *et al.* [44] used HEFT generated solutions to the initial value of GA to propose a hybrid genetic algorithm for workflow scheduling in cloud computing. The objective was to reduce the makespan. The proposed algorithm was tested against heuristics algorithms like Modified critical path (MCP) and HEFT, a generic evolutionary algorithm (PEGA), and recently proposed hybrid genetic algorithms like multiple priority queues genetic algorithm (MPQGA) and hybrid successor concerned heuristics genetic scheduling (HSCGS). The results showed improvement in average schedule length, load balancing, and communication to computation ratio. In the future, authors try to optimize the scheduling using more data-intensive and complex workflows like real-world scientific workflows.

Delava and Aryan [45] proposed a hybrid heuristic algorithm (HSGA) for finding the optimal solution in terms of makespan and load balancing of workflow scheduling in a cloud computing environment. The proposed hybrid algorithm used Best-Fit and Round Robin algorithms to obtain the genetic algorithm's initial population. HSGA produced better results in comparison to variants of the GA algorithm with an increasing number of tasks.

## 4.3. Hybrid using ACO

Kaur and Kaur [46] developed a VM load balancing framework named Hybrid approach based Deadline constrained, dynamic VM provisioning, and load balancing (HDD-PLB). The HDD-PLB was used for evaluating the two proposed hybrid algorithms called predict earliest finish time-ACO (PEFT-ACO) and heterogeneous earliest finish time-ACO (HEFT-ACO) using Cloud Workflow Simulator. PEFT-ACO was the hybridization of predict earliest finish time (PEFT) heuristics and ant colony optimization (ACO). HEFT-ACO was the hybridization of Heterogeneous Earliest Finish Time (PEFT) heuristics and ant colony optimization (ACO). The objectives include makespan and cost. The results show that the PEFT-ACO gives optimal results in CyperShake and LIGO workflow.

The ant colony optimization (ACO) algorithm was combined with a genetic algorithm in [47] for solving the task scheduling problem in cloud computing to propose a genetic-ant-colony hybrid Algorithm. The basic idea was to initialized the pheromone in ACO with the best chromosome generate by GA. The aim of the proposed algorithm was load balancing and optimal timespan. The simulation was done using the CloudSim simulator and found that the proposed algorithm produced good results in terms of accounted objectives compared to standard GA and ACO. In the future, the same experiments will be conducted in the real cloud environment to form more practical results.

Ghumman and Kaur [48] proposed a hybrid scheduling algorithm using improved max-min and ACO algorithm for load balancing in cloud computing. The objective was to minimize the makespan.

Simulation results using CloudSim suggested that the proposed hybrid algorithm performs better in terms of total processing time and processing cost compared to the original max-min algorithm. The same work can be carried out in the real environment for tests, and the cost model can be considered in the future.

### 4.4. Other hybrid algorithms

Hariharan and Raj [49] used Firefly (FF) and BAT algorithms to develop a new hybrid algorithm named Hybrid FF-BAT. The proposed algorithm's basic idea was to give the best solution to the FF algorithm to the initial solution of the BAT algorithm. The simulator used was CloudSim 3.0.3 for evaluation purposes. The proposed algorithm minimized the total execution cost. The proposed algorithm also reported converging fast than standalone FF and BAT algorithms and BAT-FF.

Eng *et al.* [50] presented the hybridization of great deluge (GD) and variable neighborhood descent (VND) meta-heuristics for reducing the completion time of the tasks. The proposed algorithm, named GDVND, was an extended version of the VND algorithm, which has the GD's excellent diversification capability. The proposed algorithm was useful in finding the optimal solution compared to other algorithms using a Grid simulator. The authors also reported that the proposed algorithm was not trapped in the local optimal solution easily. In the future, inconsistent and partial consistent properties like dynamic processing power will be included in the study.

Gao *et al.* [51] proposed a multi-objective hybrid algorithm using two well-known algorithms, genetic algorithm (GA) and artificial bee colony (ABC), to solve workflow scheduling problem. In the proposed algorithm, the resultant chromosome generated by the GA algorithm in the first half of the iteration was passed to ABC for the latter half of the proposed algorithm to optimize the workflow makespan and cost simultaneously. The proposed algorithm was simulated using WorkflowSim-1.0 with some existing algorithms like multi-objective ACO (PBACO), hybrid multi-objective PSO (HPSO), multi-objective GA (MGA), multi-objective artificial bee colony (MABC), and heterogeneous earliest finish time (HEFT). The proposed algorithm has an overall better performance under four well-known scientific workflows: Montage, LIGO, CyberShake, and Sipht.

Alazzam *et al.* [52] used two popular bio-inspired algorithms, tabu search (TS) and harmony search (HS), to propose a new hybrid tabu-harmony task scheduling algorithm (THTS) in cloud computing. The harmony search is one of the well-known algorithms for achieving global optimization, and Tabu search memorized the previously searched results, thus avoiding the repetition. The proposed algorithm (THTS) takes the advantages of both the involved algorithms. The objective was to optimize the results in terms of throughput, makespan, and cost. The proposed THTS algorithm was evaluated using CloudSim V 4.0.3 with existing Tabu, Harmony search, and round robin algorithm and found that it was better. The proposed algorithm (THTS) will be implemented using more complex dependent tasks like scientific workflows in the future.

Elaziz *et al.* [53] used the differential evolution (DE) to improve the exploration ability of the moth search algorithm (MS) to design the new task scheduling algorithm in a cloud environment. The proposed algorithm's basic idea was to divide the population into half based on the fitness value and apply the DE algorithm to the latter half to improve the exploration ability of the MS algorithm. The results were compared with existing heuristics methods like SJF and RR and some meta-heuristic methods like PSO, Whale Optimization Algorithm (WOA), and moth search algorithm (MSA) and found the reduction in makespan. Since the proposed algorithm was evaluated under only one objective that is makespan, more than one objective can be considered in the future. The workflow schedule can also be viewed with some other QoS parameters like memory usage, the peak of demands, and overload in the near future.

Choudhary *et al.* [54] introduced a gravitational search algorithm (GSA) based hybrid algorithm for workflow scheduling. The basic idea was to replace one of the agents generated by heterogeneous earliest finish time (HEFT) into GSA's population. The author considered two objectives that were makespan and total cost for optimization. The authors used the C++ coding environment for simulation experiments. They found that the proposed algorithm was better than standard GSA, hybrid genetic algorithm (HGA), and the HEFT in terms of monetary cost ratio (MCR) and schedule length ratio (SLR). The authors evaluated the results using five popular scientific workflows: CyberShake, LIGO, Montage, SIPHT, and Inspiral, and validate the results using the ANOVA statistical test. The available bandwidth between the VM can be considered variable during the simulation, and VM failure will be considered during the task execution in the future works.

Another multi-objective bio-inspired meta-heuristic hybrid algorithm proposed by the Anwar and Deng [55]. The proposed algorithm uses symbiotic organisms search (SOS) and predict earliest finish time (PEFT) to solve scientific workflow in a cloud environment. The Montage, LIGO, SIPHT, CyberShake, and Epigenomics scientific workflow were used for scheduling. In the proposed algorithm, the global exploration ability of symbiotic organisms search (SOS) was increased by the predicted earliest finish time (PEFT) algorithm to enhance the convergence rate and diversity towards the true Pareto optimal front. The objective of the proposed algorithm was to optimize the results in teams of makespan and cost. The authors found that the proposed algorithm performed better in comparison to MOHEFT, MOPSO, NSGA-II. The author will consider the other parameters like energy and carbon dioxide CO2 emissions for optimization in the near future.

Li *et al.* [56] proposed a local search enhanced hybrid artificial bee colony algorithm (LABC) with three objectives, i.e., minimize the makespan, maximize the workload, and the total workload was considered simultaneously. A deep exploration function was developed to enhance the local search ability of the Artificial Bee Colony algorithm. To simulate the detailed encoding and decoding mechanism of the problem, the authors used the C++ environment. The experiment results found that the proposed LABC algorithm outperformed the other existing algorithms.

Teylo *et al.* [57] introduced a hybrid evolutionary algorithm (HEA) for solving task scheduling and data assignment problem (TaSDAP) of scientific workflow on clouds called HEA-TaSDAP, which includes an evolutionary algorithm (EA), local search, and a path relinking method. The objective was to minimize the execution time. The author used C/C++ environment for simulation and found that the proposed algorithm outperformed the other classical approaches such as Min-Min and HEFT. The authors also considered real executions in the Amazon EC2 cloud using real scientific workflows.

Abdullahi and Ngadi [58] proposed a hybrid symbiotic organisms search (SOS) optimization algorithm to improve the local search ability of standard simulated annealing (SA). The simulation results showed that the proposed hybrid SOS algorithm performs better than SOS in convergence speed, response time, degree of imbalance, and makespan. The well-known CloudSim simulator was used for the analysis of results. The other existing meta-heuristics algorithm can enhance the local search ability of proposed hybrid SOS in the future.

## 5. DISCUSSION

In this section, we discuss the current trends of the hybrid algorithm from the literature. Figure 8 shows the number of proposed hybrid algorithms as per the classification mentioned in section 4. It is noted that most of the proposed algorithms used particle swarm optimization (PSO), genetic algorithm (GA), and ant colony optimization (ACO) algorithms or its variant in hybridization.

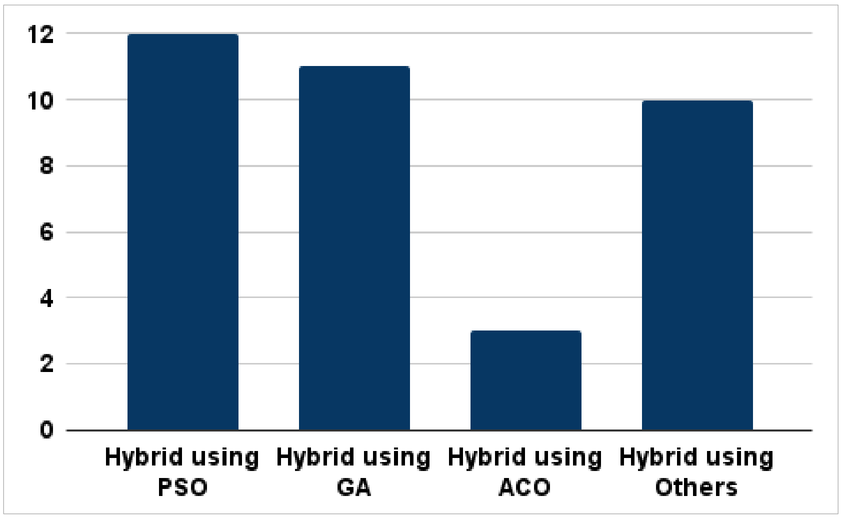


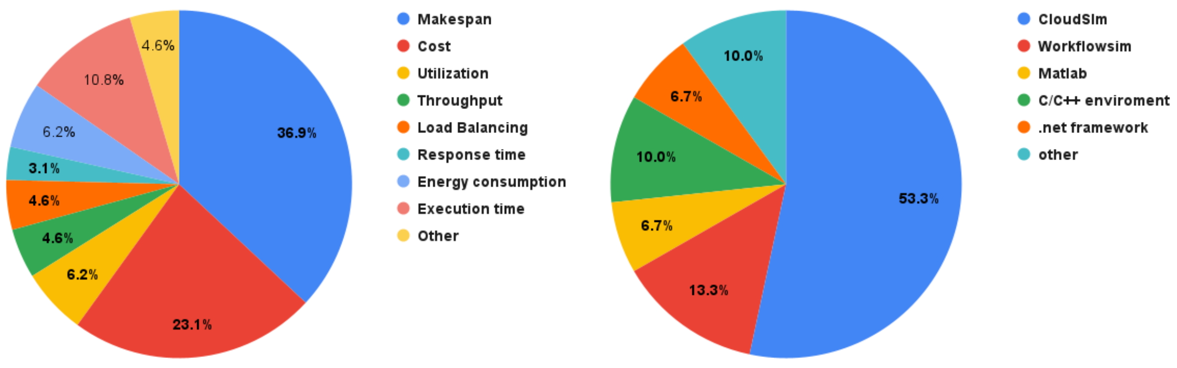
Figure 8. Classification of hybrid scheduling algorithms in cloud computing

Table 1 shows the various scheduling objectives involved in the proposed hybrid algorithms from the literature. Most of the proposed hybrid algorithms considered makespan and cost objectives for optimization in the scheduling problem. Makespan and Cost are used in 36.9% and 23.1% of the proposed algorithms, respectively, as depicted in Figure 9(a).

Several simulators are available for simulating the results of scheduling algorithms in cloud computing like CloudSim, workflowSim, and MATLAB. CloudSim [59] is the most popular, about 53.3% of the published papers from the literature using CloudSim. WorkflowSim [60] is an extension of CloudSim generally used for workflow scheduling. WorkflowSim is used in 13.3% of the involved papers from the literature, as shown in Figure 9(b).

Table 1. Scheduling objectives used in proposed algorithms

| Reference | Makespan | Cost | Utilization | Throughput | Load Balancing | Response time | Energy consumption | Execution time | Other |
|---|---|---|---|---|---|---|---|---|---|
| [25] | ✓ | ✓ | | | | | | | |
| [26] | | | ✓ | | | ✓ | | | |
| [27] | ✓ | | | ✓ | | | ✓ | | |
| [36] | ✓ | ✓ | | | | | | ✓ | |
| [37] | | ✓ | | | | | | ✓ | |
| [28] | ✓ | | ✓ | ✓ | | | | ✓ | |
| [46] | ✓ | ✓ | | | | | | | |
| [49] | | | | | | | | ✓ | |
| [38] | | ✓ | | | | | | | |
| [50] | ✓ | | | | | | | | |
| [39] | ✓ | ✓ | | | | | | | ✓ |
| [29] | ✓ | | ✓ | | | | | | |
| [52] | ✓ | ✓ | | ✓ | | | | | |
| [53] | ✓ | | | | | | | | |
| [51] | ✓ | ✓ | | | | | ✓ | | |
| [54] | ✓ | ✓ | | | | | | | |
| [30] | ✓ | | | | | | | | |
| [31] | | | | | | ✓ | | | |
| [40] | ✓ | | | | | | ✓ | | |
| [41] | | ✓ | | | | | | ✓ | |
| [55] | ✓ | ✓ | | | | | | | |
| [56] | ✓ | | | | | | | | ✓ |
| [32] | ✓ | ✓ | | | | | | | |
| [42] | | | | | | | | | ✓ |
| [47] | | | | | ✓ | | | ✓ | |
| [43] | ✓ | | | | | | | | |
| [57] | | | | | | | | ✓ | |
| [11] | ✓ | | | | | | | | |
| [44] | ✓ | | | | ✓ | | | | |
| [33] | | ✓ | | | | | | | |
| [34] | ✓ | ✓ | ✓ | | | | | | |
| [35] | | ✓ | | | | | ✓ | | |
| [45] | ✓ | | | | ✓ | | | | |
| [48] | ✓ | | | | | | | | |



(a) (b)

Figure 9. Proposed hybrid scheduling algorithms: (a) objectives, (b) simulators

## 6. CONCLUSION

Several heuristic and meta-heuristic methods were proposed to solve the scheduling problem in cloud computing. With the increasing number of tasks, virtual machines, and physical machines, the scheduling problem becomes more complicated. Even though the meta-heuristics methods fail to generate near-optimal solutions, a well-known PSO algorithm was reported to be stuck in the local optimal results. In recent years, the hybrid scheduling algorithms were more popular that combines the best properties of two or more algorithms to find the optimal solution.

In this paper, we survey the hybrid scheduling algorithms in cloud computing. Most of the proposed hybrid algorithms were combined with the existing or modified PSO, GA, and ACO versions. So, we classified the hybrid algorithms according to the combined algorithms. It is found that most of the hybrid scheduling algorithms used makespan and cost as an objective function, and the CloudSim simulator is most popular among all for generating the simulation results. In the future, a more multi-objective version of the hybrid algorithms can be proposed. Although a few works were done with the dependent tasks, this can be studied in the future. The hybrid algorithms can be tested in the real world cloud environment. More hybridization can be simulated using the different combinations of meta-heuristics algorithms in the future.

## REFERENCES

[1] V. Vinothina, R. Sridaran, and P. Ganapathi, "A survey on resource allocation strategies in cloud computing," *International Journal of Advanced Computer Science and Applications (IJACSA),* vol. 3, no. 6, pp. 97-104, 2012, doi: 10.14569/IJACSA.2012.030616.

[2] M. R. Belgaum, S. Musa, M. S. Mazliham, and M. Alam, "A behavioral study of task scheduling algorithms in cloud computing," *International Journal of Advanced Computer Science and Applications (IJACSA),* vol. 10, no. 7, pp. 498-503, 2019, doi: 10.14569/ijacsa.2019.0100768.

[3] X. F. Liu, Z. H. Zhan, J. D. Deng, Y. Li, T. Gu, and J. Zhang, "An energy efficient ant colony system for virtual machine placement in cloud computing," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 1, pp. 113-128, 2018, doi: 10.1109/TEVC.2016.2623803.

[4] S. Singh and I. Chana, "A survey on resource scheduling in cloud computing: Issues and challenges," *Journal of Grid Computing,* vol. 14, no. 2, pp. 217-264, 2016, doi: 10.1007/s10723-015-9359-2.

[5] E. J. Ghomi, A. M. Rahmani, and N. N. Qader, "Load-balancing algorithms in cloud computing: A survey," *Journal of Network and Computer Applications,* vol. 88, pp. 50-71, 2017, doi: 10.1016/j.jnca.2017.04.007.

[6] P. Singh, M. Dutta, and N. Aggarwal, "A review of task scheduling based on meta-heuristics approach in cloud computing," *Knowledge and Information Systems,* vol. 52, no. 1, pp. 1-51, 2017, doi: 10.1007/s10115-017-1044-2.

[7] M. Adhikari, T. Amgoth, and S. N. Srirama, "A survey on scheduling strategies for workflows in cloud environment and emerging trends," *ACM Computing Surveys,* vol. 52, no. 4, pp. 1-36, 2019, Art. no. 68, doi: 10.1145/3325097.

[8] S. Yassir, Z. Mostapha, and T. Claude, "Workflow scheduling issues and techniques in cloud computing: A systematic literature review," *International Conference of Cloud Computing Technologies and Applications,* vol. 49, 2019, pp. 241-263, doi: 10.1007/978-3-319-97719-5_16.

[9] F. Wu, Q. Wu, and Y. Tan, "Workflow scheduling in cloud: a survey," *The Journal of Supercomputing,* vol. 71, pp. 3373-3418, 2015, doi: 10.1007/s11227-015-1438-4.

[10] L. Liu, M. Zhang, Y. Lin, and L. Qin, "A survey on workflow management and scheduling in cloud computing," *2014 14th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, 2014, pp. 837-846, doi: 10.1109/CCGrid.2014.83.

[11] A. Kamalinia and A. Ghaffari, "Hybrid task scheduling method for cloud computing by genetic and DE algorithms," *Wireless Personal Communications*, 2017, doi: 10.1007/s11277-017-4839-2.

[12] M. Kalra and S. Singh, "Multi-criteria workflow scheduling on clouds under deadline and budget constraints," *Concurrency and Computation: Practice and Experience*, vol. 31, no. 17, pp. 1-16, 2019, doi: 10.1002/cpe.5193.

[13] M. A. Rodriguez and R. Buyya, "A taxonomy and survey on scheduling algorithms for scientific workflows in IaaS cloud computing environments," *Concurrency and Computation: Practice and Experience*, vol. 29, no. 8, pp. 1-23, 2017, doi: 10.1002/cpe.4041.

[14] E. Deelman *et al.*, "Pegasus, a workflow management system for science automation," *Future Generation Computer Systems,* vol. 46, pp. 17-35, 2015, doi: 10.1016/j.future.2014.10.008.

[15] R. Graves *et al.*, "CyberShake: A physics-based seismic hazard model for southern california," *Pure and Applied Geophysics,* 2011, doi: 10.1007/s00024-010-0161-6.

[16] Q. Jiang, Y. C. Lee, M. Arenaz, L. M. Leslie, and A. Y. Zomaya, "Optimizing scientific workflows in the cloud: A montage example," *2014 IEEE/ACM 7th International Conference on Utility and Cloud Computing, UCC 2014,* 2014, pp. 517-522, doi: 10.1109/UCC.2014.77.

[17] J. Livny, H. Teonadi, M. Livny, and M. K. Waldor, "High-throughput, kingdom-wide prediction and annotation of bacterial non-coding RNAs," *PLoS One*, vol. 3, no. 9, 2008, Art. no. e3197, doi: 10.1371/journal.pone.0003197.

[18] F. Ebadifard and S. M. Babamir, "A PSO-based task scheduling algorithm improved using a load-balancing technique for the cloud computing environment," *Concurrency and Computation: Practice and Experience*, vol. 30, no. 12, pp. 1-16, 2017, Art. no. e4368, doi: 10.1002/cpe.4368.

[19] J. Sahni and P. Vidyarthi, "A cost-effective deadline-constrained dynamic scheduling algorithm for scientific workflows in a cloud environment," *IEEE Transactions on Cloud Computing*, vol. 6, no. 1, pp. 2-18, 2018, doi: 10.1109/TCC.2015.2451649.

[20] A. Khalili and S. M. Babamir, "Optimal scheduling workflows in cloud computing environment using Pareto-based grey wolf optimizer," *Concurrency and Computation: Practice and Experience*, vol. 29, no. 11, pp. 1-11, 2017, doi: 10.1002/cpe.4044.

[21] D. Gabi, A. S. Ismail, A. Zainal, Z. Zakaria, and A. Abraham, "Orthogonal Taguchi-based cat algorithm for solving task scheduling problem in cloud computing," *Neural Computing and Applications*, vol. 30, no. 6, pp. 1845-1863, 2018, doi: 10.1007/s00521-016-2816-4.

[22] L. Zhang, Y. Chen, R. Sun, and B. Yang, "A task scheduling algorithm based on PSO for grid computing," *International Journal of Computational Intelligence Research*, vol. 4, no. 1, 2008, pp. 37-43, doi: 10.5019/j.ijcir.2008.123.

[23] K. Dasgupta, B. Mandal, P. Dutta, and S. Dam, "A genetic algorithm (GA) based load balancing strategy for cloud computing," *Procedia Technology,* vol. 10, pp. 340-347, 2013, doi: 10.1016/J.PROTCY.2013.12.369.

[24] M. Tawfeek, A. El-Sisi, A. Keshk, and F. Torkey, "Cloud task scheduling based on ant colony optimization," *2013 8th International Conference on Computer Engineering & Systems (ICCES)*, 2015, pp. 64-69, doi: 10.1109/ICCES.2013.6707172.

[25] M. H. Shirvani, "A hybrid meta-heuristic algorithm for scientific workflow scheduling in heterogeneous distributed computing systems," *Engineering Applications of Artificial Intelligence*, vol. 90, 2020, Art. no. 103501, doi: 10.1016/j.engappai.2020.103501.

[26] S. G. Domanal, R. M. R. Guddeti, and R. Buyya, "A hybrid bio-inspired algorithm for scheduling and resource management in cloud environment," *IEEE Transactions on Services Computing,* vol. 13, no. 1, pp. 3-15, 2020, doi: 10.1109/TSC.2017.2679738.

[27] U. K. Jena, P. K. Das, and M. R. Kabat, "Hybridization of meta-heuristic algorithm for load balancing in cloud computing environment," *Journal of King Saud University - Computer and Information Sciences,* 2020, doi: 10.1016/j.jksuci.2020.01.012.

[28] M. M. Golchi, S. Saraeian, and M. Heydari, "A hybrid of firefly and improved particle swarm optimization algorithms for load balancing in cloud environments: Performance evaluation," *Computer Networks*, vol. 162, 2019, Art. no. 106860, doi: 10.1016/j.comnet.2019.106860.
[29] N. Mansouri, B. M. H. Zade, and M. M. Javidi, "Hybrid task scheduling strategy for cloud computing by modified particle swarm optimization and fuzzy theory," *Computers and Industrial Engineering*, vol. 130, pp. 597-633, 2019, doi: 10.1016/j.cie.2019.03.006.
[30] N. Dordaie and N. J. Navimipour, "A hybrid particle swarm optimization and hill climbing algorithm for task scheduling in the cloud environments," *ICT Express*, vol. 4, no. 4, pp. 199-202, 2018, doi: 10.1016/j.icte.2017.08.001.
[31] B. Jana and J. Poray, "A hybrid task scheduling approach based on genetic algorithm and particle swarm optimization technique in cloud environment," *Intelligent Engineering Informatics*, vol. 695, pp. 607-614, 2018, doi: 10.1007/978-981-10-7566-7_61.
[32] A. Verma and S. Kaushal, "A hybrid multi-objective particle swarm optimization for scientific workflow scheduling," *Parallel Computing*, vol. 62, pp. 1-19, 2017, doi: 10.1016/j.parco.2017.01.002.
[33] S. Mirzayi and V. Rafe, "A hybrid heuristic workflow scheduling algorithm for cloud computing environments," *Journal of Experimental & Theoretical Artificial Intelligence*, vol. 27, no. 6, pp. 721-735, 2015, doi: 10.1080/0952813X.2015.1020524.
[34] H. M. Alkhashai and F. A. Omara, "BF-PSO-TS: Hybrid heuristic algorithms for optimizing task schedulingon cloud computing environment," *International Journal of Advanced Computer Science and Applications (IJACSA)*, vol. 7, no. 6, pp. 207-212, 2016, doi: 10.14569/ijacsa.2016.070626.
[35] S. George, "Hybrid PSO-MOBA for profit maximization in cloud computing," *International Journal of Advanced Computer Science and Applications*, vol. 6, no. 2, pp. 159-163, 2015, doi: 10.14569/ijacsa.2015.060223.
[36] S. Velliangiri, P. Karthikeyan, V. M. A. Xavier, and D. Baswaraj, "Hybrid electro search with genetic algorithm for task scheduling in cloud computing," *Ain Shams Engineering Journal*, vol. 12, no. 1, pp. 631-639, 2020, doi: 10.1016/j.asej.2020.07.003.
[37] H. Aziza and S. Krichen, "A hybrid genetic algorithm for scientific workflow scheduling in cloud environment," *Neural Computing and Applications*, vol. 8, pp. 15263-15278, 2020, doi: 10.1007/s00521-020-04878-8.
[38] D. Chaudhary and B. Kumar, "Cost optimized hybrid genetic-gravitational search algorithm for load scheduling in cloud computing," *Applied Soft Computing*, vol. 83, 2019, Art. no. 105627, doi: 10.1016/j.asoc.2019.105627.
[39] G. Natesan and A. Chokkalingam, "Multi-objective task scheduling using hybrid whale genetic optimization algorithm in heterogeneous computing environment," *Wireless Personal Communication*, vol. 110, pp. 1887-1913, 2019, doi: 10.1007/s11277-019-06817-w.
[40] S. Srichandan, T. A. Kumar, and S. Bibhudatta, "Task scheduling for cloud computing using multi-objective hybrid bacteria foraging algorithm," *Future Computing and Informatics Journal*, vol. 3, no. 2, pp. 210-20, 2018, doi: 10.1016/j.fcij.2018.03.004.
[41] A. M. Manasrah and H. B. Ali, "Workflow scheduling using hybrid GA-PSO algorithm in cloud computing," *Wireless Communications and Mobile Computing*, vol. 2018, 2018, Art. no. 1934784, doi: 10.1155/2018/1934784.
[42] K. Loheswaran and J. Premalatha, "A hybrid GA-IWO scheduling algorithm," *2017 International Conference on Algorithms, Methodology, Models and Applications in Emerging Technologies (ICAMMAET)*, 2017, pp. 1-5, doi: 10.1109/ICAMMAET.2017.8186732.
[43] D. Nasonov, A. Visheratin, N. Butakov, N. Shindyapina, M. Melnik, and A. Boukhanovsky, "Hybrid evolutionary workflow scheduling algorithm for dynamic heterogeneous distributed computational environment," *Journal of Applied Logic*, vol. 24, pp. 50-61, 2017, doi: 10.1016/j.jal.2016.11.013.
[44] S. G. Ahmad, C. S. Liew, E. U. Munir, T. F. Ang, and S. U. Khan, "A hybrid genetic algorithm for optimization of scheduling workflow applications in heterogeneous computing systems," *Journal of Parallel and Distributed Computing*, vol. 897, pp. 80-90, 2016, doi: 10.1016/j.jpdc.2015.10.001.
[45] A. G. Delavar and Y. Aryan, "HSGA: A hybrid heuristic algorithm for workflow scheduling in cloud systems," *Cluster Computing*, vol. 17, pp. 129-137, 2014, doi: 10.1007/s10586-013-0275-6.
[46] A. Kaur and B. Kaur, "Load balancing optimization based on hybrid heuristic-metaheuristic techniques in cloud environment," *Journal of King Saud University-Computer and Information Sciences*, 2019, doi: 10.1016/j.jksuci.2019.02.010.
[47] Z. Wu, S. Xing, S. Cai, Z. Xiao, and Z. Ming, "A genetic-ant-colony hybrid algorithm for task scheduling in cloud system," *International Conference on Smart Computing and Communication*, vol. 10135, 2017, pp. 183-193, doi: 10.1007/978-3-319-52015-5_19.
[48] N. S. Ghumman and R. Kaur, "Dynamic combination of improved max-min and ant colony algorithm for load balancing in cloud system," *2015 6th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, 2016, pp. 1-5, doi: 10.1109/ICCCNT.2015.7395172.
[49] B. Hariharan and D. P. Raj, "A hybrid framework for job scheduling on cloud using firefly and BAT algorithm," *International Journal of Business Intelligence and Data Mining*, vol. 15, no. 4, 2019, doi: 10.1504/IJBIDM.2019.102811.
[50] K. L. Eng, A. Muhammed, M. A. Mohamed, and S. Hasan, "A hybrid heuristic of variable neighbourhood descent and great deluge algorithm for efficient task scheduling in grid computing," *European Journal of Operational Research*, vol. 284, no. 1, pp. 75-86, 2020, doi: 10.1016/j.ejor.2019.12.006.
[51] Y. Gao, S. Zhang, and J. Zhou, "A hybrid algorithm for multi-objective scientific workflow scheduling in IaaS cloud," *IEEE Access*, vol. 7, pp. 125783-125795, 2019, doi: 10.1109/ACCESS.2019.2939294.
[52] H. Alazzam, E. Alhenawi, and R. Al-Sayyed, "A hybrid job scheduling algorithm based on Tabu and Harmony search algorithms," *The Journal of Supercomputing*, vol. 76, no. 12, pp. 7994-8011, 2019, doi: 10.1007/s11227-019-02936-0.
[53] M. A. Elaziz, S. Xiong, K. P. N. Jayasena, and L. Li, "Task scheduling in cloud computing based on hybrid moth search algorithm and differential evolution," *Knowledge-Based Systems*, vol. 169, pp. 39-52, 2019, doi: 10.1016/j.knosys.2019.01.023.
[54] A. Choudhary, I. Gupta, V. Singh, and P. K. Jana, "A GSA based hybrid algorithm for bi-objective workflow scheduling in cloud computing," *Future Generation Computer Systems*, vol. 83, pp. 14-26, 2018, doi: 10.1016/j.future.2018.01.005.
[55] N. Anwar and H. Deng, "A hybrid metaheuristic for multi-objective scientific workflow scheduling in a cloud environment," *Applied Sciences* vol. 8, no. 4, pp. 1-21, 2018, doi: 10.3390/app8040538.
[56] J. Q. Li, Y. Y. Han, and C. G. Wang, "A hybrid artificial bee colony algorithm to solve multi-objective hybrid flowshop in cloud computing systems," *International Conference on Cloud Computing and Security*, vol. 10602, pp. 201-213, 2017, doi: 10.1007/978-3-319-68505-2_18.
[57] L. Teylo, U. de Paula, Y. Frota, D. de Oliveira, and L. M. M. A. Drummond, "A hybrid evolutionary algorithm for task scheduling and data assignment of data-intensive scientific workflows on clouds," *Future Generation Computer Systems*, vol. 76, pp. 1-17, 2017, doi: 10.1016/j.future.2017.05.017.

[58] M. Abdullahi and M. A. Ngadi, “Hybrid symbiotic organisms search optimization algorithm for scheduling of tasks on cloud computing environment,” *PLoS One*, vol.11, no. 6, 2016, Art. no. e0158229, doi: 10.1371/journal.pone.0158229.
[59] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. De Rose, and R. Buyya, “CloudSim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms,” *Software-Practice & Experience*, 2011, doi: 10.1002/spe.995.
[60] W. Chen and E. Deelman, “WorkflowSim: A toolkit for simulating scientific workflows in distributed environments,” *2012 IEEE 8th International Conference on E-Science*, 2012, pp. 1-8, doi: 10.1109/eScience.2012.6404430.

## BIOGRAPHIES OF AUTHORS

**Neeraj Arora** was born in Jodhpur, Rajasthan, India, in 1989. He received the B.Tech. degree from the Jodhpur Institute of Engineering and Technology, Jodhpur, in 2001 and the M.E. degree from the M.B.M. Engineering College, Jodhpur, in 2013, both in computer science and engineering. He is currently an Assistant Professor of Computer Science at Vardhman Mahaveer Open University, Kota and pursuing the Ph.D. degree with the Department of Computer Science and Engineering, Rajasthan Technical University, Kota. His research interests include Optimization Algorithms, Cloud Computing, Resource Scheduling, and Workflow Scheduling. He can be contacted at email: narora@vmou.ac.in, neeraj.arora0007@gmail.com.

**Rohitash Kumar Banyal** has been working as Associate Professor in the Department of Computer Science and Engineering in Rajasthan Technical University, Kota, India. He has published many research papers in national and international journal of repute. He has received his Doctorate of Philosophy (Ph.D.) in Computer Science & Engineering from SLIET, Longawal, Sangrur, and Punjab, India. Prior to this, he has obtained his M Tech in Computer Science and Engineering from IIT, Delhi. His areas of interest are Cloud computing, Big Data, and High-Performance Computing. He has published many research paper in various national and international journal of repute. He is a reviewer of the various national and international journal of repute. He has conducted many workshops, seminar, and conferences in the department and university level as well. He has been invited as a subject expert, speaker at various levels and occasion in different universities. He has more than 20 years of teaching experience in engineering and technology. He has guided more than 20 M Tech students and currently 6 students are pursuing doctorate under his supervision. He can be contacted at email: rkbanyal@rtu.ac.in.