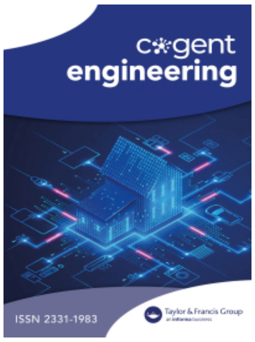


Taylor & Francis
Taylor & Francis Group

Cogent Engineering

ISSN: (Print) (Online) Journal homepage: www.tandfonline.com/journals/oaen20

Toward optimizing scientific workflow using multi-objective optimization in a cloud environment

Shabina Ghafir, M. Afshar Alam, Farheen Siddiqui, Sameena Naaz, Shahab Saquib Sohail & Dag Øivind Madsen

To cite this article: Shabina Ghafir, M. Afshar Alam, Farheen Siddiqui, Sameena Naaz, Shahab Saquib Sohail & Dag Øivind Madsen (2024) Toward optimizing scientific workflow using multi-objective optimization in a cloud environment, Cogent Engineering, 11:1, 2287303, DOI: 10.1080/23311916.2023.2287303

To link to this article: https://doi.org/10.1080/23311916.2023.2287303

Published online: 19 Dec 2023.

cogent engineering

COMPUTER SCIENCE | RESEARCH ARTICLE

Toward optimizing scientific workflow using multi-objective optimization in a cloud environment

Shabina Ghafir[1], M. Afshar Alam[1], Farheen Siddiqui[1], Sameena Naaz[1], Shahab Saquib Sohail[1] and Dag Øivind Madsen[2*]

Received: 13 August 2023
Accepted: 18 November 2023

*Corresponding author: Dag Øivind Madsen, USN School of Business, University of South-Eastern Norway, Hønefoss, Norway
E-mail: dag.oivind.madsen@usn.no

Reviewing editors:
Jenhui Chen, Computer Science and Information Engineering, Chang Gung University, Taiwan
Ahmed Haj Darwish, Faculty of Informatics Engineering, University of Aleppo, Aleppo, Syrian Arab Republic

Additional information is available at the end of the article

Abstract: Scientific workflows are a common and critical part of scientific computing, involving complex computations and oversized and distributed computing resources. Efficient workflow execution requires scheduling algorithms considering task dependencies, resource requirements, and deadlines. Cloud computing provides an innovative architecture for extensive heterogeneous computing services. However, scheduling hybrid cloud resources with deadline restrictions while observing QoS standards is an NP-complete task. Mapping workflow tasks to virtual machines and determining the optimal schedule order is a challenging aspect of cloud computing. By executing task requests on the most advantageous virtual machine in the resource pool, energy consumption, overall execution time, and computing costs can be reduced. This research aims to identify the best location to process applications using user's demand and priority. A multi-objective genetic algorithm is proposed to achieve this objective, which considers conflicting objectives such as time, energy, cost, and deadline. The algorithm initializes an efficient ranking heuristic approach and predicts the earliest finish time (PEFT) using the Bayesian approach to improve the Pareto fronts. This approach enhances the VM migration of cloud-based tasks and optimizes the search space for conflicting objectives. Experimental findings show that the proposed approach reduces cost by 5–6% and time delay by 8% compared to existing approaches. The proposed approach offers an effective solution for scheduling scientific workflows on cloud computing resources while considering various QoS standards. The results demonstrate the potential of multi-objective genetic algorithms for optimizing workflow scheduling in cloud computing environments.

Subjects: Algorithms & Complexity; Artificial Intelligence; Computer Science (General)

Keywords: workflow scheduling; load balancing; Artificial Intelligence; optimization; ranking; ChatGPT; Pareto; PEFT

1. Introduction

Cloud computing is a method of providing users with resources via the Internet and centralized remote servers. It involves using a large number of shared heterogeneous resources to deliver a wide range of benefits to its users, each with its own set of quality of service (QoS) needs (Farid et al., 2020; Ismayilov & Topcuoglu, 2020; S. Kaur et al., 2019; Krishan & Kumar, 2020; Masdari et al., 2016; Singh & Singh, 2013; Wu et al., 2015). The most popular cloud platforms include GoGrid, Amazon EC2, Microsoft Azure, and Google App Engine. Private, public, hybrid and community clouds are the most common clouds. Any subscriber can use a public cloud; however, private clouds and their architecture are controlled and used by certain businesses. Some organizations share community clouds and can be administered by them or third-party service providers. Hybrid clouds make use of both private and public cloud resources. In addition, due to the unavailability of a single cloud, a trend towards multi-clouds has emerged, which involves the federation of multiple clouds.

Furthermore, cloud-based services can be classified as infrastructure as a service (IaaS), platform as a service (PaaS), or software as a service (SaaS) provider (Gupta et al., 2019; Nirmala & Bhanu, 2016). Customers lease enterprise software as a service from SaaS providers, whereas PaaS providers provide wide infrastructure resources such as storage, computing, and networks. Virtualization is often a critical enabler of cloud computing because it allows multiple virtual machines to coexist on a single physical computer. A virtual machine (VM) simulates a computer network and executes user-defined functions. By instantiating, clients can deploy the apps on several diversely performing and cost-effective resources (Alkhanak et al., 2015; Ghasemi & Hanani, 2019). Every physical server or machine is managed by a software layer known as the VM monitor, which allows for the production and operation of the Workflow scheduling (Bhasker & Murali, 2023; Gupta & Mangla, 2019; P. Kaur & Sachdeva, 2016; Kousalya et al., 2017; Sohail, Javed, et al., 2023). It is a crucial topic in cloud technology because it attempts to map workflow tasks to virtual machines based on various functional and non-functional needs. A workflow is a collection of interdependent actions linked by functional or data-based requirements; these connections must be considered while scheduling.

Additionally, workflow scheduling is an NP-hard optimization problem in cloud technology, making attaining an optimum schedule challenging (Afzal & Kavitha, 2019; E. A. Kaur, 2017; S.Nagadev et al., 2013; Patel & Bhoi, 2013; Zhao et al., 2013). Since a cloud contains many virtual machines and diverse user workloads, scheduling must consider various scheduling goals and scenarios. The common goal of a workflow is to reduce the time required to complete a task by allocating tasks to appropriate virtual resources.

Cloud computing has revolutionized the way businesses operate and manage their IT infrastructure (Katyal & Mishra, 2014; Kumar & Kumar, 2019). One of the key benefits of cloud computing is the ability to provision computing resources on demand, enabling businesses to scale up or down their computing resources as needed. However, managing cloud resources can be challenging, particularly when optimizing scheduling workflows (Chawla & Bhonsle, 2012; Hariri et al., 2022; Shah et al., 2018).

Workflows are related tasks that must be executed to achieve a specific goal. In cloud computing, workflows can include provisioning virtual machines, launching applications, and processing data. Optimizing the scheduling of these workflows can help businesses reduce costs, improve performance, and enhance the overall efficiency of their IT operations (Areeb et al., 2023; S. Li et al., 2021). There are several critical motivations for optimizing workflows in the cloud. One of the most important is cost reduction. By optimizing workflows, businesses can minimize the time and resources required to execute tasks, thereby reducing costs. Particularly important for businesses with fluctuating workloads and the need to scale their computing resources up and down as needed.

Another motivation for optimizing workflows is performance improvement and security (B. Li et al., 2019; M. Li et al., 2023; Shishido et al., 2018; Wang et al., 2020). By scheduling workflows more efficiently, businesses can ensure that tasks are completed on time, reducing processing time and improving overall performance. This is especially important for businesses that rely on real-time data processing or need to meet strict service level agreements. Optimizing workflows can enhance the overall efficiency of IT operations. Businesses can free up their IT staff to focus on more strategic initiatives by automating repetitive tasks and reducing manual intervention. This can help businesses become more agile and responsive to changing business needs while improving the quality of service they provide to their customers.

To that end, this work primarily fulfills the need of an optimize work scheduling procedure which is lacking in the literature, and it adds value to the literature by convincingly proposing a better approach with an optimized result. In this work, our main work is to focus on designing an optimization framework for scientific workflow. The study leverages the Bayesian framework and the maximum likelihood technique to enhance the accuracy of its results through optimal estimations and predictions, while addressing the complexity of multi-objective optimization problems by incorporating a random distribution element that introduces variability into the optimization process, thus allowing the model to explore a broader solution space and potentially uncover more diverse and effective solutions; furthermore, the research adopts a synergistic approach by fusing multiple heuristic techniques, which are efficient and effective problem-solving strategies, to enhance the model's ability to navigate complex optimization landscapes, and despite incorporating randomness, the study strategically reduces it to enhance the reliability of its outcomes, ensuring that they are not overly influenced by unpredictable factors. Precisely, the prime contributions can be outlined as -

- Utilization of Bayesian framework and maximum likelihood technique.
- Incorporation of Random Distribution in Multi-Objective Optimization.
- Synergistic Fusion of Heuristic Techniques.
- Enhanced reliability through reduced randomness.

The rest of the paper is organized as follows: in the next section, we have reviewed the works that focus on optimization problem for scientific workflow. In Section 3, material and methods are discussed. In Section 4, the results are presented and have been discussed in details. We have concluded the article in Section 5, additionally, limitation and future directions are also stated.

2. Related work

Authors in Qin et al. (2023) emphasize a workflow scheduling issue with many objectives in multi-cloud systems (MOWSP-MCS). The time, cost, and dependability are regarded as the optimization goals from the user's standpoint. In contrast to conventional multi-objective scheduling problems inside the cloud, MOWSP-MCS enables customers to implement the backup strategy to increase reliability. This research offers a reliability-aware multi-objective mimetic algorithm (RA MOMA) with an intensification approach and a diversification strategy to tackle the MOWSP-MCS. The diversification technique incorporates many problem-specific genetic algorithms to generate offspring with a wide range of characteristics. In the case of the intensification technique, four kinds of problem-specific neighbourhood operators are built upon the resource utilization rate and critical path in order to increase the archive set's quality. A complete statistical experiment is performed to measure the efficacy of RA-MOMA. RA-MOMA is better than various comparable approaches to solving the MOWSP-MCS, as demonstrated by comparisons with these algorithms.

In Mangalampalli et al. (2023) the authors suggest a new workflow scheduling system that takes into account the order of importance of tasks and puts them on the right virtual resources. This algorithm was modelled using the whale optimization technique as the procedure. A workflow sim simulator was used to run a lot of simulations. It was tested against the CS, PSO, ACO, and GA algorithms that were already in use. Lastly, the simulation findings demonstrated that migration time, makespan, and energy use were all kept to a minimum.

In their work authors in Zhou et al. (2022) suggest a security-aware and a makespan workflow scheduling scheme that utilizes a revised firefly optimizer with a novel solution initializing plan, a position-updating method for fireflies, two task-to-VM assessment strategies, and a firefly solution map-based operator. To prove that the researchers proposed workflow scheduling technique works, designers run many simulations of actual workflows. When contrasted with a base classifier and two cutting-edge methodologies, the results indicate that this strategy can reduce costs by up to 54.0%.

An approach based on time, cost, and average use is suggested in Pillareddy and Karri (2023). The authors recommend MONWS, which utilizes the min-max technique to minimize cycle time and optimize resource usage by determining the optimum time to run tasks on virtual machines and the corresponding adaptive threshold value for prioritizing tasks in workflow scheduling. Compared to state-of-the-art algorithms, MONWS reduced Makespan by35%, increased maximum cloud capacity by 8%, and reduced costs by 4%.

Authors in Arora and Banyal (2022) introduced the PSO–GWO hybrid meta-heuristic algorithm. The proposed methodology is a mixture of two optimization algorithms, GWO and PSO. For the scientific workflows Cybershake, Montage, SIPHT, and Spiral, the PSO–GWO algorithm is evaluated. The reason for this new approach is to decrease the total cost of execution. The PSO-GWO method speeds up typical total execution time while reducing overall operatingcost, in comparison to the standard PSO and GWO algorithms.

In their work Qin et al. (2022) proposed a new hybrid collaborative multi-objective fruit fly optimization algorithm (HCMFOA) to minimize the amount of cost and time spent on the execution. The recommended HCMFOA uses a cluster technique based on points of reference to split the swarm into smaller groups. In addition, a hybrid starting approach is developed by combining a non-linear weight vector with two principles for task allocation. Each fruit fly within the subspace has a starting position determined by using this method. In coordinated smell-based forage, three problem-solving operators living in the same neighborhood examine the global scope in collaboration. The sub-swarms-based crossover operator is made for local exploitation in multi-objective vision-based foraging. Lastly, a large-scale computational test is done to test how well HCMFOA works. The statistical findings indicate that HCMFOA does a lot better than the current top-of-the-line methods.

Authors in Mangalampalli, Swain, et al. (2022) assessed the task priority among all tasks entering the cloud interface, and effectively mapped tasks to VMs. Cat swarm optimization (CSO) was utilized by researchers to address the scheduling challenge, finally addressing the consumption of resources and Makespan. Designers ran simulations on workflow sim and compared the effectiveness of the approach to that of known algorithms CS and PSO. Based on simulation findings, designers discovered that the approach outperforms existing algorithms for the variables specified.

In their work, authors in F. Li et al. (2022) employ a three-step scheduling scheme to combine the scheduling of container-based workflow processes and the execution of containers in a cloud-edge context. In the first phase, every container is assigned a virtual CPU (vCPU). This enables CPU sharing amongst containers. Then, a two-area deployment is used to place containers on virtual machines and virtual machines on physical equipment, either in the cloud or at the edge. From the point of view of cloud-edge resources and containerized workflows, numerous goals are taken into account. These include minimizing Makespan, energy use, and load imbalance (Naaz et al., 2012; Shafiq et al., 2022). Two different multi-objective algorithm architectures are used in conjunction with three different evolutionary approaches to generate a collection of non-dominated options. These are the basic non-co-evolution strategy (B-NCS), the co-evolution strategy (CES), and the hybrid non-co-evolution strategy (H-NCS). The results of the simulations show that H-NCS is more effective than other techniques, and the proposed approach yields better results as compared to the current two-step scheduling paradigm.

A modified version of the Firefly method for addressing workflow scheduling issues in a cloud-edge scenario is proposed in Bacanin et al. (2022). By combining genetic operators and a quasi-reflection-based learning technique, the suggested model improves upon the traditional firefly meta heuristics' errors. To start, the recommended improved approach was validated on 10 recently released benchmark test instances, and its effectiveness was assessed in relation to the baseline and to other enhanced state-of-the-art meta-heuristics. Secondly, simulations were conducted to examine a workflow scheduling problem with dual primary goals and a timeline. The researchers conducted a comparison analysis with other cutting-edge methods tested under identical experimental circumstances. With respect to output clarity and fast convergence, the algorithm shown here greatly exceeds the classic Firefly method and other outstanding meta-heuristics. Results from simulations show that the revised firefly algorithm provides a significant improvement over existing solutions to the scheduling problem at the cloud's edge, both in terms of cost a Makespan.

Authors in Belgacem and Beghdad-Bey (2022) look at the possible trade-off between virtual machine utilization cost and Makespan. Using the ant colony algorithm (ACO) and the heterogeneous earliest end time (HEFT) to minimize them, researchers present a HEFT-ACO strategy. Considering the features of Amazon EC2 cloud service, three distinct kinds of genuine scientific workflows are simulated experimentally while taking into account the Amazon EC2 cloud infrastructure. The test findings demonstrate that the suggested method outperforms ant colony algorithm (ACO), Fuzzy resource utilization with multi-objective scheduling (FR-MOS), and Predict the Earliest Finish Time-Ant Colony Algorithm (PEFT-ACO).

The research articles which are discussed above offer diverse perspectives on the optimization of task scheduling in cloud computing environments. To further extending the work, Bezdan et al. (2022). introduced a hybridized bat optimization algorithm, aiming for efficient multi-objective task scheduling with superior results compared to similar methods. Following a different approach, Mirmohseni et al. (2022) developed a hybrid Fuzzy Particle Swarm Optimization Genetic Algorithm (FPSO-GA) specifically for load balancing in cloud networks, combining fuzzy particle swarm optimization and genetic algorithm techniques. Aktan and Bulut (2022) contributed to the field by discussing the challenges inherent in scheduling applications in cloud computing. They underscored the critical need for capturing and disseminating scientific information within a collaborative context, recognizing the broader implications of scheduling in this domain. Addressing the dynamic nature of cloud resources and workload demands, Mangalampalli, Karri, et al. (2022) presented a multi-objective task scheduling Grey Wolf Optimization (MOTSGWO) algorithm. Zivkovic et al. (2022) proposed an improved Harris Hawks Optimization algorithm for workflow scheduling in a cloud-edge environment, outperforming existing approaches by reducing cost and improving makespan performance metrics (Miftakhov et al., 2021; Thammarak et al., 2022).

2.1. Research gap

Task scheduling is one of the critical issues in cloud computing due to the heterogeneous nature of cloud resources. Many algorithms in a real-time scenario have been put forward to solve the task scheduling problem in the cloud computing domain. So far, many task scheduling techniques have been discovered to solve the scheduling problem, and each scheduling technique has its advantages and limitations due to considering different QoS parameters. In general scheduling-based techniques are classified into two major categories: heuristic, and meta-heuristic (Hariri et al., 2022; R. Kaur & Singh Dhindsa, 2018; Kumar & Kumar, 2019; B. H. Malik et al., 2018; Naz et al., 2023; Rodriguez & Buyya, 2014; Shah et al., 2018; Shameer & Subhajini, 2017; Sohail et al., 2023; Sreenu & Sreelatha, 2019).

Prior research focused on task mapping. On VM, single, hybrid, or multi-objective optimization can be used. However, during this process, the initiation of optimization can be random, or some researchers use other heuristics such as HEFT. In random processes, the maximum

Table 1. Review of latest appraoches

Ref	Aim	Approach	Dataset/Tool	Inference	Limitation
Pillareddy and Karri (2023)	Planning Cloud-Based Workflows with Multi-Objectives Normalization	Multi-Objective Normalization Workflow Scheduling (MONWS): Montage, Cybershake, LIGO, and Epigenomics.	Cloudsim tool	According to the findings, the suggested algorithm achieved better results than the state-of-the-art algorithm when used in real-world scientific computing applications.	When it comes to handling VM failure and monitoring energy use, the proposed algorithm requires some work.
Arora and Banyal (2022)	The metaheuristic methodologies utilized to determine optimal workflow scheduling.	Particle Grey Wolf Hybrid Algorithm for Workflow Scheduling (PSO-GWO)	WorkflowSim-1.1 toolkit	The PSO–GWO algorithm reduces the average overall execution cost and time compared to PSO and GWO.	No overlapping interests
Bacanin et al. (2022)	To adapt firefly algorithm for cloud-edge workflow scheduling	Genetic operators quasi-reflected FA (GOQRFA).	Workflow models dataset	The GOQRFA scheduler is able to satisfy the real-time requirement while simultaneously attempting to find the corresponding balance between the makespan and cost criterion.	The proposed GOQRFA has the drawback of having more control parameters which should be adjusted by the researcher, as any other hybridized or upgraded metaheuristic approach.
Bhasker and Murali (2023)	Smart Irrigation System Workflow Scheduling Using FMMEHO in a Cloud-Based Virtualization Environment	Membership Mutation Elephant Herding Optimization (FMMEHO), Optimum Energy and Resource Aware Workflow Scheduling (OERES) scheme	Publicly Available dataset Aneka Platform (5.0)	The findings show how successful of the proposed OERES algorithm is compared current approaches extremely well.	The drawbacks of irrigation systems include the risk of diseases from fluid overflow, their high initial cost, and the fact that they

probability of a false positive threshold does not enhance resource usage, and the heuristic approach only works for a single target. It only works for small workflows or jobs. Thus, the approach proposed in this research addresses these drawbacks and demonstrates the improvement over prior approaches in various workflows. Heuristic approaches use the prediction to reach the near-optimal solution with low time complexity and high scheduling length. At the same time, meta-heuristics directly find the solution with more efficient results than the heuristic technique. Heuristic-based solutions are unsuitable for task scheduling problems for independent tasks due to the higher cost and energy consumption for processing the submitted jobs.Meta-heuristic-based techniques are the most suitable options for such problems by generating a cost and energy-efficient solution without pre-information of tasks and resources.

3. Material and methods

3.1. Problem formulation

Scheduling algorithms provide a mapping solution for workflow tasks to the resources like virtual machines in cloud computing that obeys the user's specified QoS constraints like a deadline and budget. The proposed method focused on scheduling the workflow tasks onto the virtual machines using the Pareto-PEFT ranking with a multi-objective genetic algorithm approach to reduce the total execution time and cost. We focus on finding a schedule = (AV M, MAP, TIME, COST) consisting of resources VMs, a mapping, total execution time, and total lease cost for a workflow application. The equation determines the total lease cost and execution time

$$EC = \sum_{X=1}^{X=N} \frac{l(n)}{Resource} * \left(R_p + R_M\right) \tag{1}$$

The scheduled cost is computed by adding the cost of every VM leased based on time interval μ. The lease cost of VM is calculated by multiplying cost per unit time $Z(V\ M_k)$ with 214 the time for which the VM type $V\ M_k$ is allocated to the task using equation 2. Every VM has its start time ST $(V\ M_k)$ and finish time FT $(V\ M_k)$ for which it is leased for executing the task.

$$TIME = max\{FT(qexit)\} \tag{2}$$

The scheduling problem can be formulated as: to generate workflow schedule WS with minimum COST and TIME is within the specified deadline constraint as depicted in Equation (3)

$$\text{Minimize : COST Subject to : TIME} \leq \text{deadline} \tag{3}$$

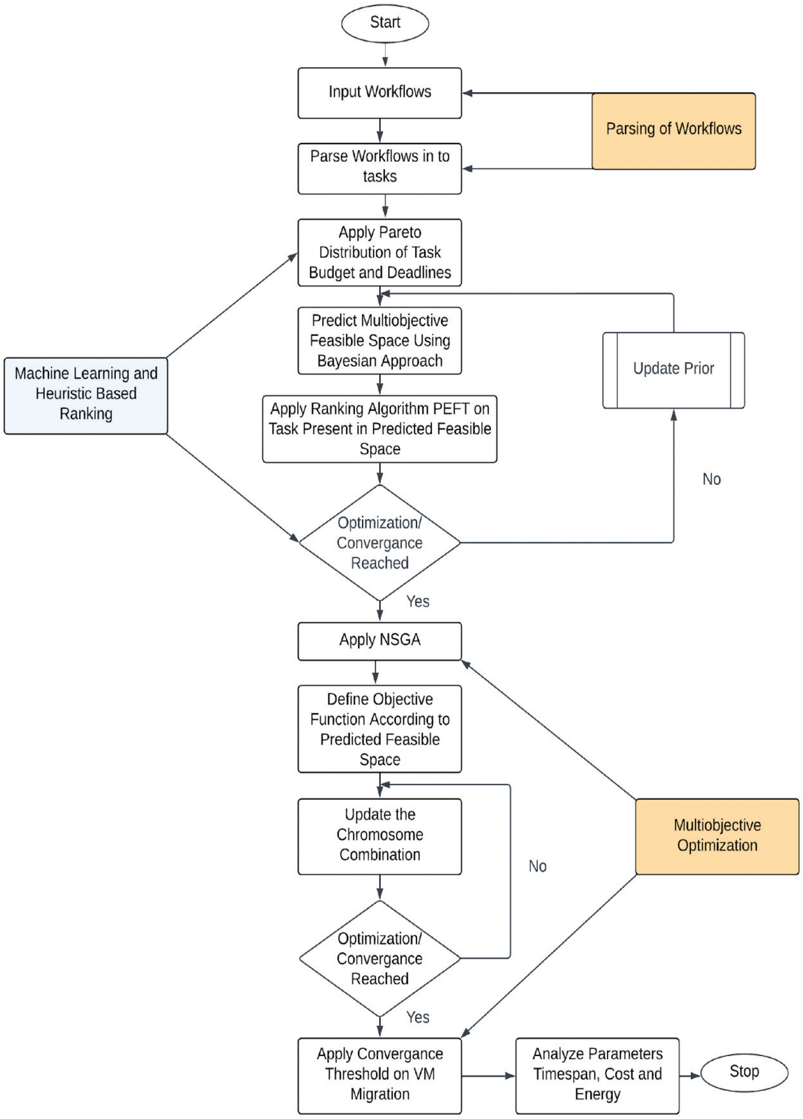
3.2. Adopted methodologies

The flow chart of the adopted methodologies is shown in Figure 1. The proposed approach makes use of the number of user workflows, physical hosts, and virtual machines (VMs) on each physical host. We can perceive the architecture into four parts:

3.2.1. Parsing workflows

The parent-child connection is referred to in the DAG workflow architecture. If there is an edge from Parent I to Child J within the DAG, we can conclude that the Child J task is a successor to a Parent I task. Due to the task precedence limitation, it can begin its execution only once the predecessor Parent I completes its activity and sends the message to its successor, Child J. Hence, even though multiple jobs will appear in the series, parse the workflows according to the parent-child connection and the stated order. We proceed to the following phase and assign an optimal ranking on the same level.

Figure 1. Flow chart of the proposed methodologies.


Start
Input Workflows
Parsing of Workflows
Parse Workflows in to tasks
Apply Pareto Distribution of Task Budget and Deadlines
Predict Multiobjective Feasible Space Using Bayesian Approach
Update Prior
Machine Learning and Heuristic Based Ranking
Apply Ranking Algorithm PEFT on Task Present in Predicted Feasible Space
No
Optimization/ Convergance Reached
Yes
Apply NSGA
Define Objective Function According to Predicted Feasible Space
Update the Chromosome Combination
No
Multiobjective Optimization
Optimization/ Convergance Reached
Yes
Apply Convergance Threshold on VM Migration
Analyze Parameters Timespan, Cost and Energy
Stop

Fitness function

$$f(P_M) = \alpha_1 * \lambda_{wi}\, P_l * ET + \alpha_2 * \lambda_{wi}\, P_l * EC + \alpha_3 * \lambda_{wi}\, P_l * E \tag{4}$$

3.2.2. Optimization of the ranking

There are three components in this section, the first of which is the optimized space, where tasks are ranked in the optimized space. It is determined by our three research objectives: time, cost, and energy. As a result, the first stage determines the Pareto front, followed by a PEFT-based ranking of that region in the second phase. Determine the probability distribution correlation between these task ranks behind ranking the whole process, and optimize using a Bayesian approach.

3.2.2.1. Pareto Front. In this part, discuss formal definitions of multi-objective optimization problems, Pareto frontier, and Pareto dominance to facilitate comprehension of this work. Since a minimization problem may describe each maximizing problem, we assume that minimizing is the goal for all objectives. Pareto's set of trade-off options is a collection of non-dominant solutions. The value of a Pareto set's solutions is called the Pareto frontier. A Pareto frontier is a helpful tool for discovering preferences and decision support.

Algorithm-1 Pareto Front

```
Input: Define Objective
Output: Optimize Pareto front
  1. D <- find Dependency in tasks on the Basis of Time and Energy
  2. If $t_i > t_l$
 $dominate\ t_i\ by\ eq(1) > domination\ (t_1)$    (2) Run step2
 If Converge
 Stop
  3. Front according to Converge
  4. run according to eq(2)
```

3.2.2.2. PEFT Ranking. Predict the Earliest Finish Time (PEFT) is a scheduling method that works with a limited number of heterogeneous processors. This algorithm works in two steps: Task Prioritizing (for computing job prioritizing) and Processor Selection (for choosing the most suitable processor for processing the current task). The task mapping is set up for optimization in this step.

Algorithm-2 PEFT RANKING

```
Input: Pareto Front
Output: Optimize Efficient ranking of tasks
  1. D<- Find Dependency in Tasks by Pareto front
  2. Compute Task ranking according to Pareto front and make list
  3. While (List >0)
  4. Compute $EFT_i$ 5. $EFT_i = Pareto\ f\ ront\ (T_i) + Resource\ time$ 6. Ranking <$MAX(EFT)$
```

3.2.2.3. Bayesian Optimization (BO). BO is indeed a black-box optimization technique with no gradients. BO has been demonstrated to be effective in optimizing complicated real-world systems that are frequently non-convex and noisy (Chawla & Bhonsle, 2012; N. Malik et al., 2021; Verma & Kaushal, 2017; Vijayalakshmi & Vasudevan, 2015). Algorithm 3 describes the BO algorithm. We first utilized the Gaussian Process (GP) approach to fit a surrogate model. The target function is assumed to be a series of jointly Gaussian random variables with covariance among arbitrary locations determined by a covariance kernel function k(xi). Using the Gaussian assumption, we may anticipate the mean (x|D of a point. The predicted uncertainty is expressed as prediction variance σ2 (x|D). For estimating the covariance, we have used a Squared-Exponential Co variance Kernel.

```
Algorithm-23: Bayesian Optimize (BO)
Input: DAG WORK FLOW WITH PEFT RANKING
Output: Optimize Ranking
    1. While
$(Task > mean\ \mu(x|D)*N)\mu$
Start
$T_C$ <-Calculate Task Time and Energy
$T_U$ <-Compared Upward
     2. If $T_U > T_C$
     3. $Rank <Bayes(T_U)$
     4. Stop
```

3.2.3. Task optimize scheduling: NSGAII

In this step, we use the data from the previous steps to create an efficient task map. As a result, NSGA I is used to learn through multi-objective optimization using Equation 2, monitor resource usage, and provide an optimal task scheduling threshold for virtual machines. One of the most often used EMO algorithms is NSGAII. The NSGAII algorithm generates a set of fronts solutions that are non-dominated. The Pareto front uses an elitist approach to maintain non-dominated solutions from every generation. The NSGAII algorithm works as follows, as shown in the diagram: The initialization process begins by producing a population of N possible solutions in this manner. After assessing the objective functions of each solution, mainly in newly created populations, the sorting operation would then execute. Individuals would be ranked into a set of fronts depending on their level of non-domination. Individuals with the same rank (but not necessarily on the same front) will also be sorted using the crowding distance values. A packed tournament selection is frequently used to identify a generation of parental individuals over whom genetic actions will be performed to establish an offspring population. As a result, the crossover procedure is carried out with a crossover probability of two parents. The children would then be modified using a mutation operator and a mutation probability. A similar process will be performed on various parents until a population of 2N population. Moreover, as with the previous population, this will be sorted and truncated to create a new community of N people. Finally, these steps would be repeated until a stop condition was met or maximum iterations were reached.

3.2.3.4. Roulette Selection. The Roulette approach is often used to select parents from the existing population according to their ranking and crowding distance. Whenever two parental chromosomes with equal rank are assessed in Roulette selection, the crowding distance has a significant influence. The crowding distance quantifies an individual's proximity to its neighbors. When a chromosome has a long crowding distance, this would increase population diversity.

In addition to this, workflow scheduling involves determining the order and allocation of tasks that make up a workflow to resources so as to optimize multiple objectives. In a cloud computing context, this might mean distributing tasks among available servers, VMs, or containers. The solution can be represented as:

Each chromosome in the population represents a potential scheduling solution. And each gene in the chromosome represents a task in the workflow. The value of each gene specifies the resource (like a specific VM or server) to which the task is assigned.

Example: If we have 4 tasks and 3 resources, a chromosome might look like this: [R1, R3, R2, R1]. This means Task 1 is assigned to Resource 1, Task 2 to Resource 3, and so forth.

We should create a population of such chromosomes, typically initialized randomly while ensuring feasibility. To evaluate, for each scheduling solution (chromosome) in the population, you'll calculate its fitness by evaluating it based on multiple objectives. Common objectives in cloud workflow scheduling include minimizing execution time, minimizing cost, maximizing reliability, etc. Since it is multi-objective, you will often rank solutions using methods like non-dominated sorting, where solutions are categorized into different fronts based on how many other solutions they dominate or are dominated by.

Based on their fitness, select chromosomes to be parents for crossover. Techniques like tournament selection can be useful where a subset of chromosomes is chosen, and the best among them (based on non-domination rank and crowding distance) is selected as a parent.

Furthermore, crossover (recombination) combines two parent chromosomes to produce offspring. This might involve swapping genes (task-resource assignments) between parents. However, mutation introduces small random changes in the offspring. In this context, it might mean changing the resource assignment for a task in a chromosome. After generating offspring, decide which solutions should continue to the next generation. This could involve selecting the best solutions from the combined set of parents and offspring, or other replacement strategies. Moreover, the algorithm continues iterating through the selection, crossover, mutation, and replacement steps for a pre-defined number of generations or until a convergence criterion is met. The final set of non-dominated solutions provides a trade-off between the objectives and can be presented to the decision-maker.

3.2.4. Parameter evaluation

- U number of users = $\{\boldsymbol{user}_1, \boldsymbol{user}_2, \ldots\ldots, \boldsymbol{user}_{\boldsymbol{n}}\}$
- Each user submits n tasks of Workflows = $\{\boldsymbol{W}_1, \boldsymbol{W}_2, \boldsymbol{W}_3 \ldots\ldots, \boldsymbol{W}_{\boldsymbol{n}}\}$
- Length l(n)
- Total Number of VM = $\{V_1, V_2, V_3 \ldots\ldots, V_n\}$
- ET $= \sum_{X=1}^{X=N} \frac{l(n)}{Resource}$

- EC $= \sum_{X=1}^{X=N} \frac{l(n)}{Resource} * (R_p + R_M)$

Algorithm 4: PPBMGA

```
INPUT: DAG Work-Flows
Output: Optimize Scheduling
1. $T$<- Parse-task (DAG)
2. $T_P$ <-$Pareto\ Front(T)$
3. $T_{PF}$ <-$PEFT(T_P)$
4. $T_{OR}$ <-$BO\ (T_{PF})$
5. $VMmapping$<-$TOR$
6. $Population$ <-$Crossover\ (VM_{mapping})$
7. $Mutation$ <-$Roulette\ selection\ (Population)$
8. If Mutation optimize and converge
9. $Threshold$ <-$Roulette\ (mutation)$ else go to step5
10. Apply threshold on VM and VM migration
11. Analysis performance metrics
```

Algorithm 5: Roulette Selection

```
Input: Crossover or mutation Population
Output: Select an efficient candidate
1. Apply Fitness Equation 1 for Selection
2. Find average of All population according to fitness value
3. If average (candidate) < average (candidate-previous)
4. else go to step2
```

- $E = \sum_{X=1}^{X=N} \frac{l(n)}{Resource} * (E_p + E_M)$

  ET <- Execution Time
  EC <-Execution Cost
  E <- Energy
- E_p*<- energy of Process or*
  E_M*<- energy of Memory*
  R_p*<- Resource use of processor*
  R_M*Resource use of Memory*

4. Result and analysis

In this section, the proposed method is used to calculate the performance of different workflows on virtual machines based on average cost, time, and energy.

Experimental parameter-based values are shown in Table 2. Review of latest approaches are shown in Table 1. VM here has values ranging from 20 to 200. The experiment involves four different types of procedures and two types of hosts. The performance metrics are based on cost, time, and energy. The highest achieved VM frequency is 1.6 GHz. The MIPS rate is 2500, and the memory capacity is 200 MB.

4.1. Dataset

The Pegasus project has described and made public the methodology for several real-world applications, including LIGO, CyberShake, GENOMIC, and LIGO SIPHT. The information presented includes the DAG, the quantity of transmitted data, and the reference execution time based on Xeon@2.3 GHz Processors. Each process has a set of these parameters. Here are the descriptions of pegasus workflow used in the article.

4.1.1. LIGO

The Laser Interferometer Gravitational Wave Observatory (LIGO) is a network of gravitational wave detectors with observatories located in Livingston, LA and Hanford, WA (Figure 2a). The mission of the observatories is to detect and measure gravitational waves predicted by the theory of general relativity—Einstein’s theory of gravity—where gravity is described as arising from the curvature of the fabric of time and space. One well-studied phenomenon expected to be the source of gravitational waves is the inspiration and merger of dense, massive astrophysical objects such as neutron stars and black holes. Such binary inspiral signals are the most promising sources of LIGO. Gravitational waves interact very weakly with matter, and when they pass through ground instruments, the measurable effects are small. A large amount of data, including a stress signal measuring the passage of gravitational waves, must be acquired and analyzed to increase the probability of detection. LIGO applications often require terabytes of data to produce meaningful results.

Table 2. Experiment setup

PARAMETERS	VALUES
Number of VM	20–200
Number of Host	2
Type of Workflows	4
Performance Metrics	Cost, Time and Energy
Maximum VM Frequency	1.6 GHz
MIPS Rate	2500 MIPS
Memory	200MB

Figure 2a. DAG for LIGO.

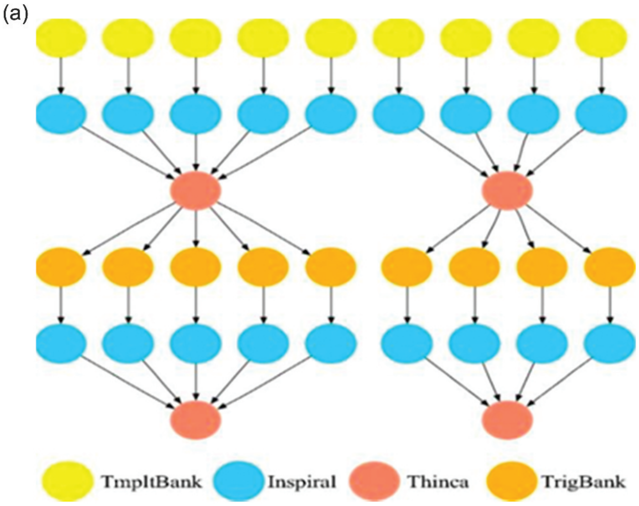
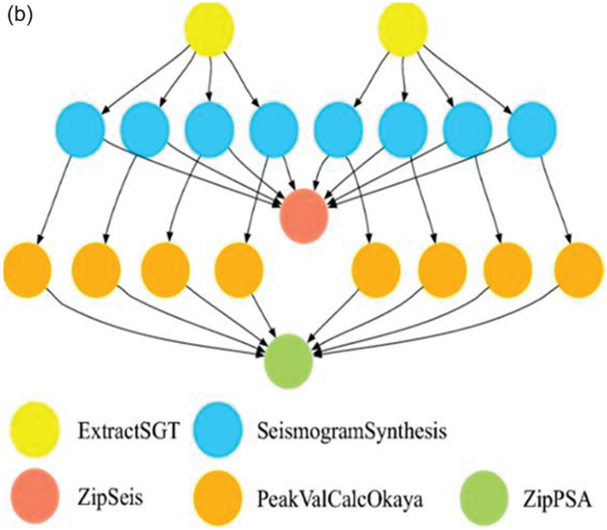


Figure 2b. DAG for Cybershake.



4.1.2. Cybershake

The Southern California Earthquake Center uses the CyberShake workflow to characterize earthquake hazards in the region. These workflows are from 2011 productions that contain high frequency codes (Figure 2b).

4.1.3. Epigenomics

An epigenomics workflow is used to automate various functions of genome sequence processing. The epigenomics workflow also has a pipeline structure and eight levels. The total input to the workflow is sequential data obtained from the genetic analysis process in multiple “lanes” (Figure 2c).

4.1.4. SIPHT

The Harvard Bioinformatics Project’s SIPHT workflow is used to automate the search for bacterial replicon untranslated RNAs (sRNA) in the NCBI database (Figure 2d).

Figure 2c. DAG for Epigenomics.

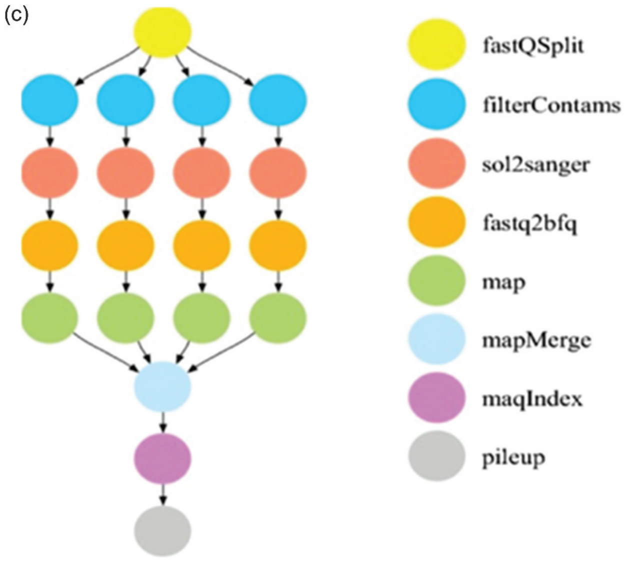
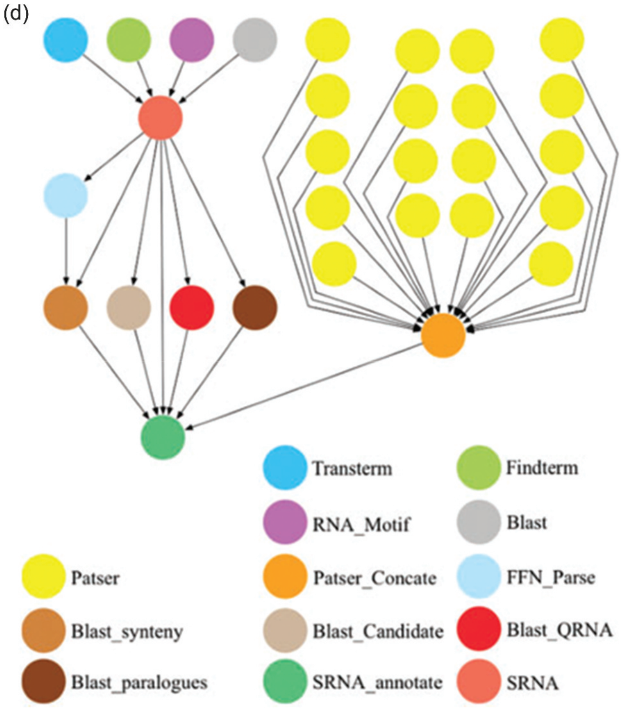


Figure 2d. DAG for SIPHT.



4.2. CYBERSHAKE workflows performance parameter analysis

Figures 3-5 compare the proposed approach to the existing approach in Cybershake Workflows with respect to cost, time, and energy usage. The approaches used are Artificial bee colony (ABC) search, Ant Colony Optimization (ACO), Grey Wolves Optimization (GWO), Tabu-GWO-ACO and the proposed method. In Figure 4, the execution cost results for 20, 40, 80, 100, 120, and 200 number of VMs is compared using the above-mentioned techniques. We find the optimum cost optimizer. The methodologies are compared. Compared to all other alternatives, the proposed approach achieves a lower execution cost for each task. In other words, the proposed method is less

Figure 3. Comparison of Energy parameter of Proposed and Existing Approach in Cyber shake Workflows.

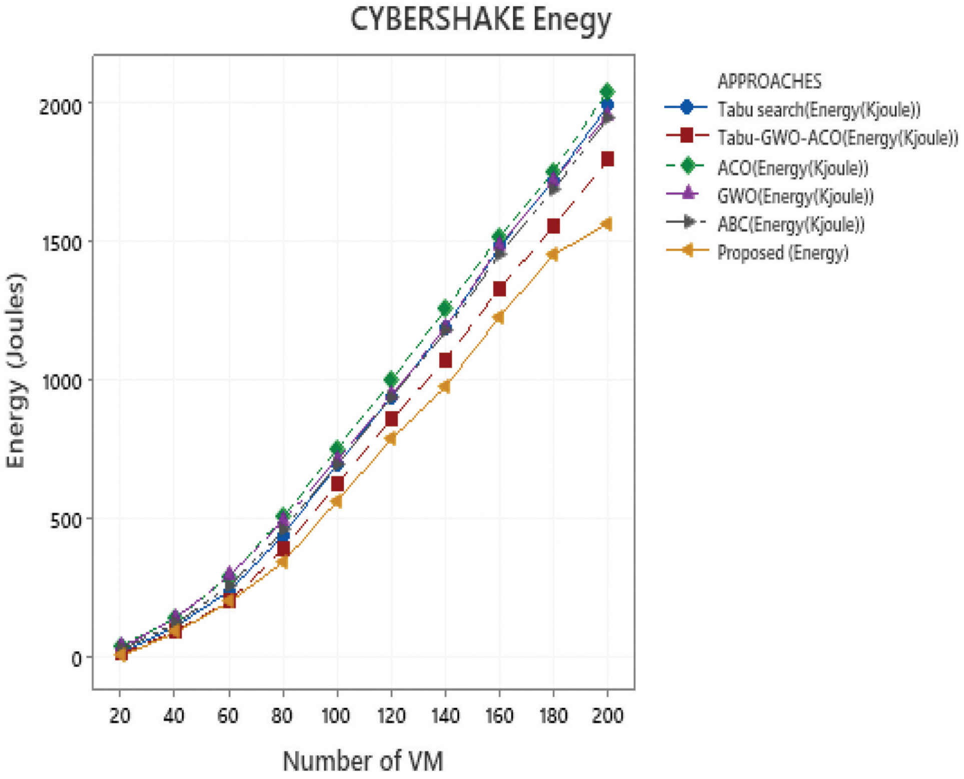
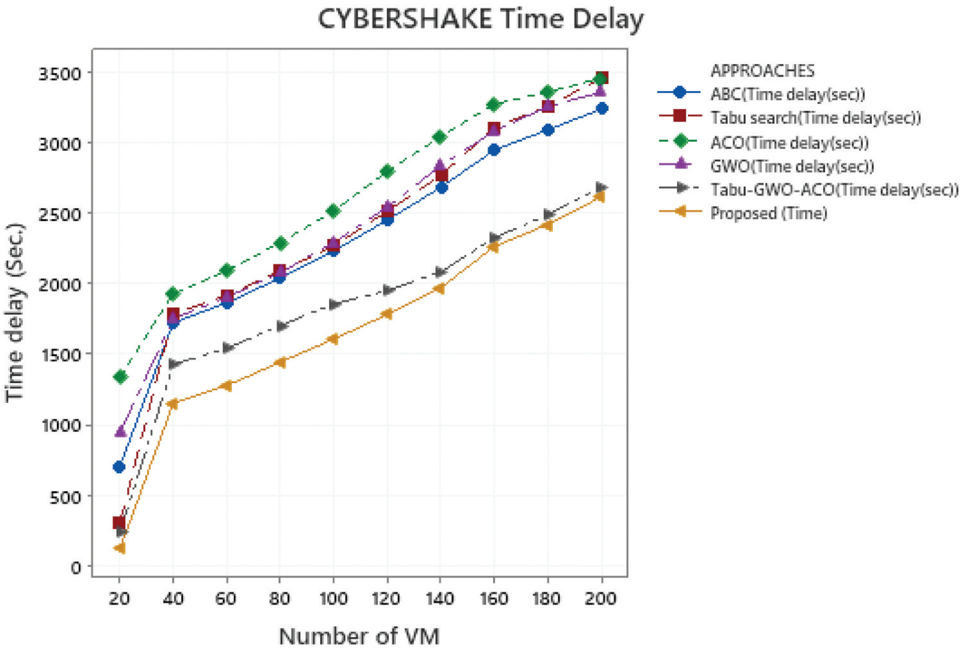
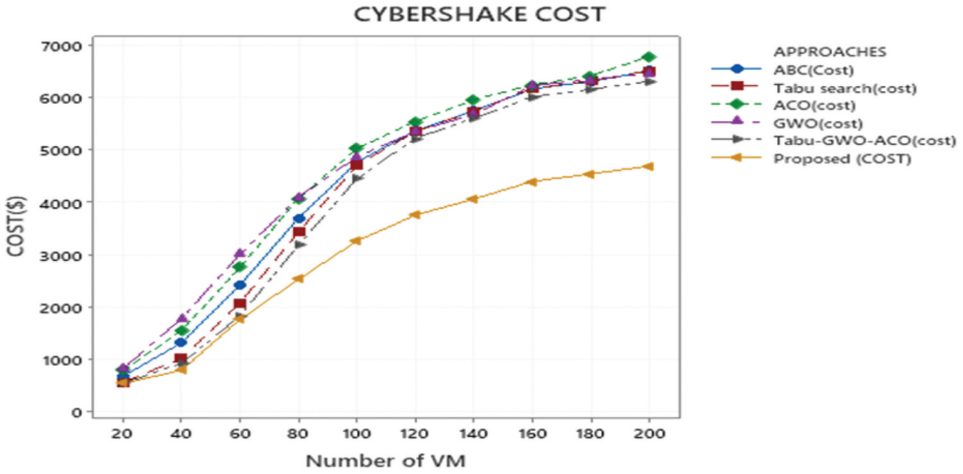


Figure 4. Comparison of Time delay parameter of Proposed and Existing approach in Cybershake Workflows.



expensive than the alternatives for creating a schedule. ACO had the highest execution cost among the other techniques, while ABC search, and GWO were virtually identical. In contrast, the Tabu-GWO-ACO (Hybrid) approach has the second lowest execution cost

Figure 5. Comparison of Cost parameter of Proposed and Existing approach in Cybershake Workflows.



In contrast, Figure 5 compares the time delay outcomes for different approaches. The proposed approach has a minor time delay compared to the other methods, with Tabu-GWO-ACO (Hybrid), ABC, GWO, and ACO having the most delay. The proposed method for shake uses the least amount of energy, followed by Tabu-GWO-ACO (Hybrid), ABC, GWO, and ACO.

4.3. LIGO Workflows Performance Parameter Analysis

Figures 6-8 illustrate how the proposed approach compares cost, time, and energy to the existing approach in LIGO Workflows. The approaches employed are ABC search, ACO, GWO, Tabu-GWO-ACO (cost), and the proposed method.

In Figure 7, the execution cost results for 25, 40, 80, 100, 120, and 200 are compared using the above-mentioned techniques. We find the optimum cost optimizer, and the approaches are

Figure 6. Comparison of Time Delay parameter of Proposed and Existing approach in LIGO Workflows.

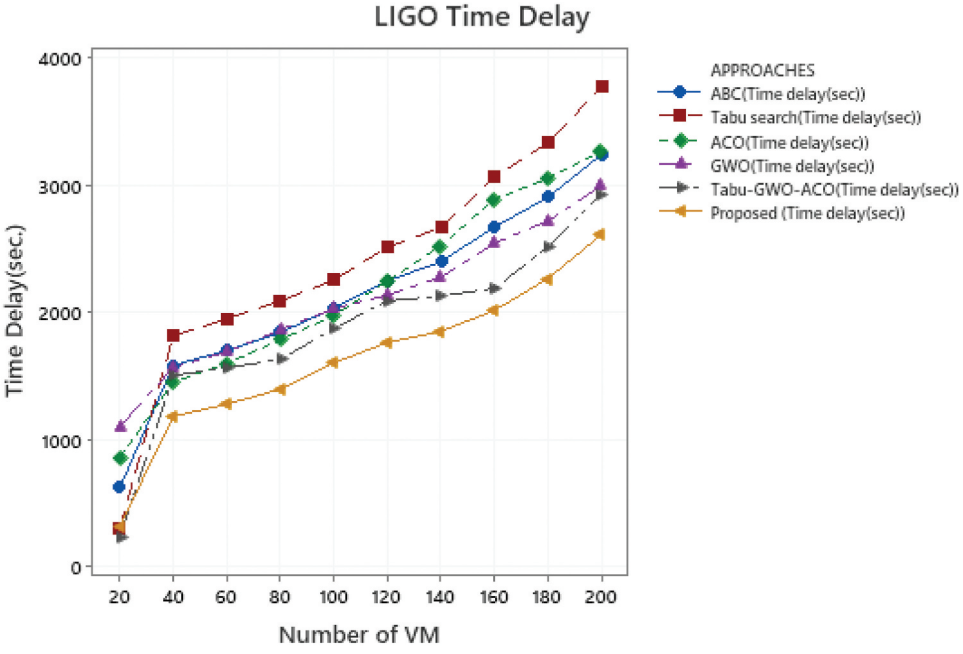
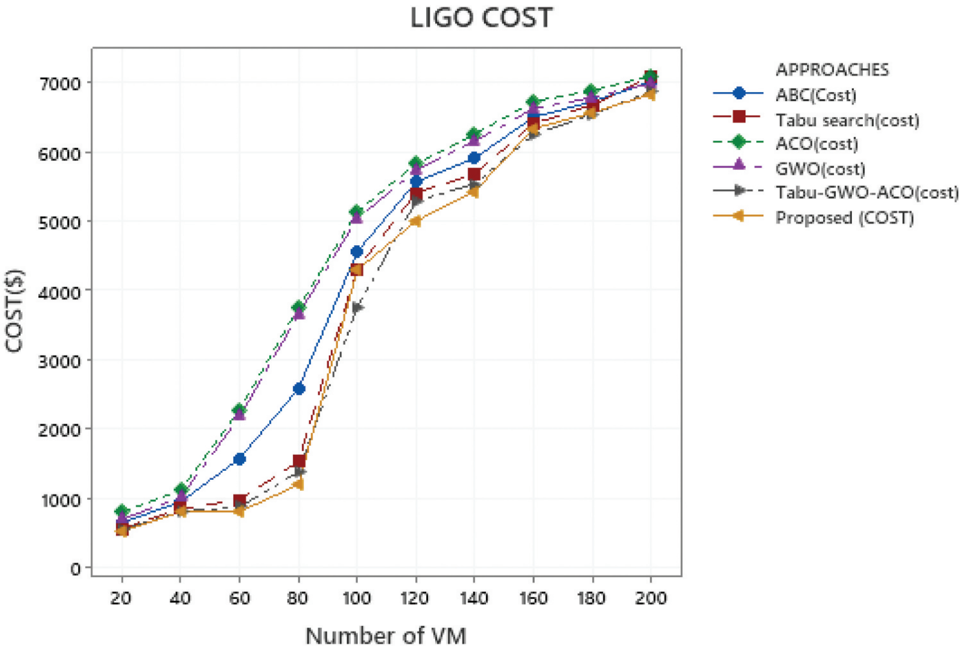


Figure 7. Comparison of Cost parameter of Proposed and Existing approach in LIGO Workflows.



contrasted. Compared to all other alternatives, the proposed method achieves lower per-task execution costs. ACO had the highest execution cost among the other approaches, followed by GWO, ABC, search, and Tabu-GWO-ACO. Hence, Tabu-GWO-ACO (Hybrid) has the second lowest execution cost. On the other hand, Figure 5 also compares the time delay results for various LIGO. The proposed approach has less time delay than the other approaches followed by Tabu-GWO-ACO (Hybrid), GWO, ABC, and ACO, and possesses the highest delay among all. The proposed approach for LIGO has a lower energy usage, followed by Tabu-GWO-ACO (Hybrid), GWO, ABC, and ACO, with the highest energy usage.

Figure 8. Comparison of Energy parameter of Proposed and Existing approach in LIGO Workflows.

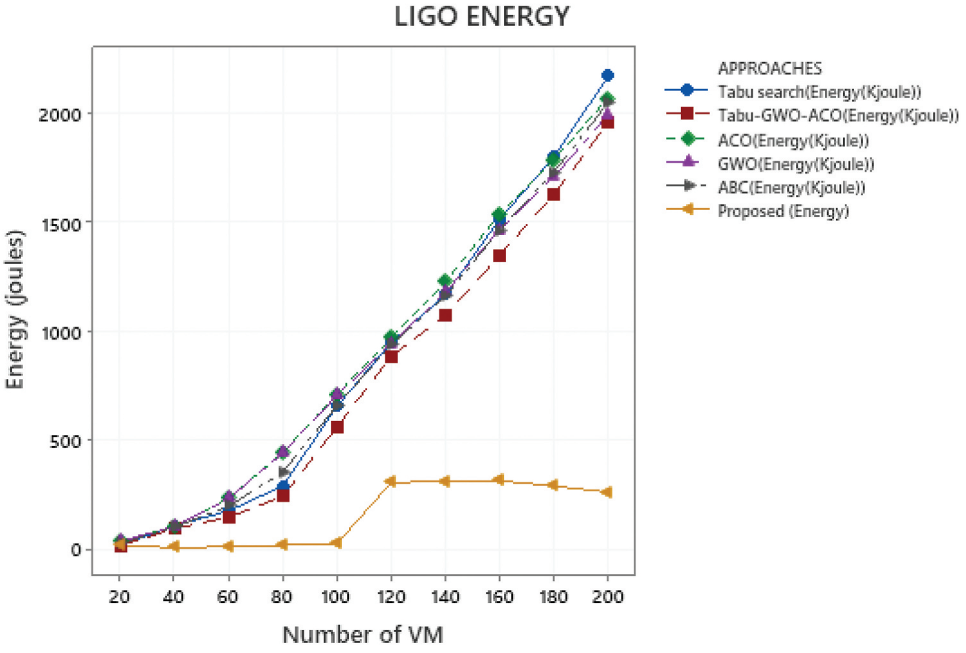
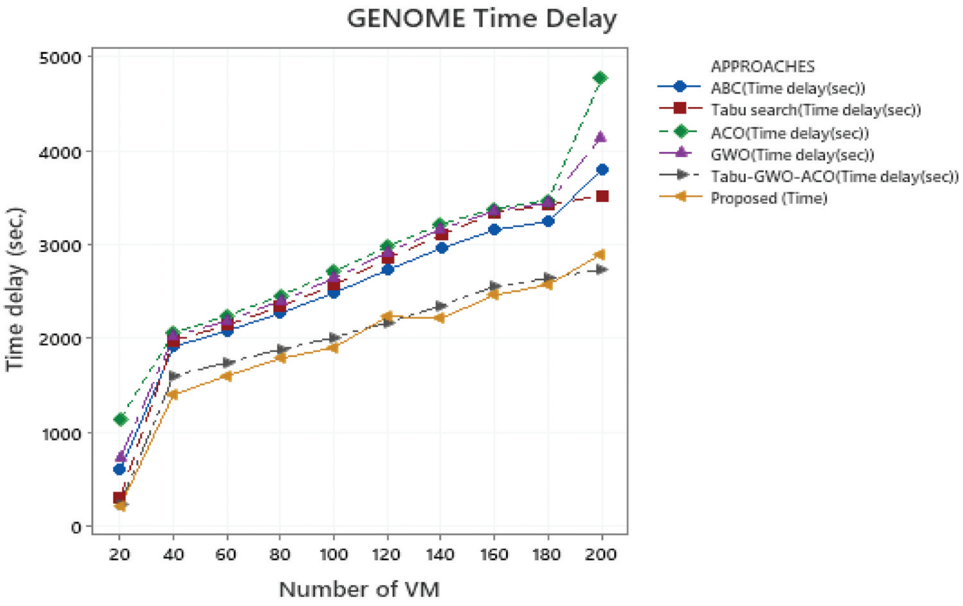


Figure 9. Comparison of Time delay parameter of Proposed and Existing approach in GENOME Workflows.



4.4. *GENOME workflows performance parameter analysis*

Figures 9-11 show how the proposed method compares to the existing approach regarding cost, time, and energy in GENOME Workflows. In comparison to all other alternatives, the proposed approach achieves a lower execution cost for each task, followed by Tabu-GWO-ACO, search, ABC, ACO, and GWO having the highest execution cost among all. On the other hand, Figure 8 also compares the time delay results for various GENOME. The proposed approach has less time delay than the other approaches, followed by Tabu-GWO-ACO (Hybrid), ABC, search, GWO, and ACO possessing the highest delay among all. The proposed approach for GENOME has a lower energy usage, followed by Tabu-GWO-ACO (Hybrid), search and ABC possessing almost the same values, GWO, and ACO, having the highest energy usage.

Figure 10. Comparison of Cost parameter of Proposed and Existing approach in GENOME Workflows.

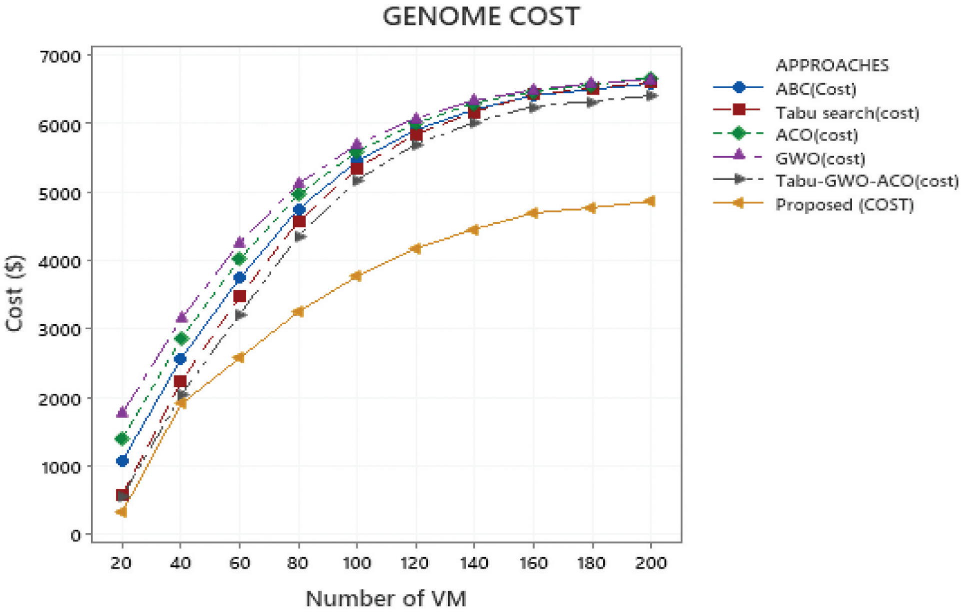
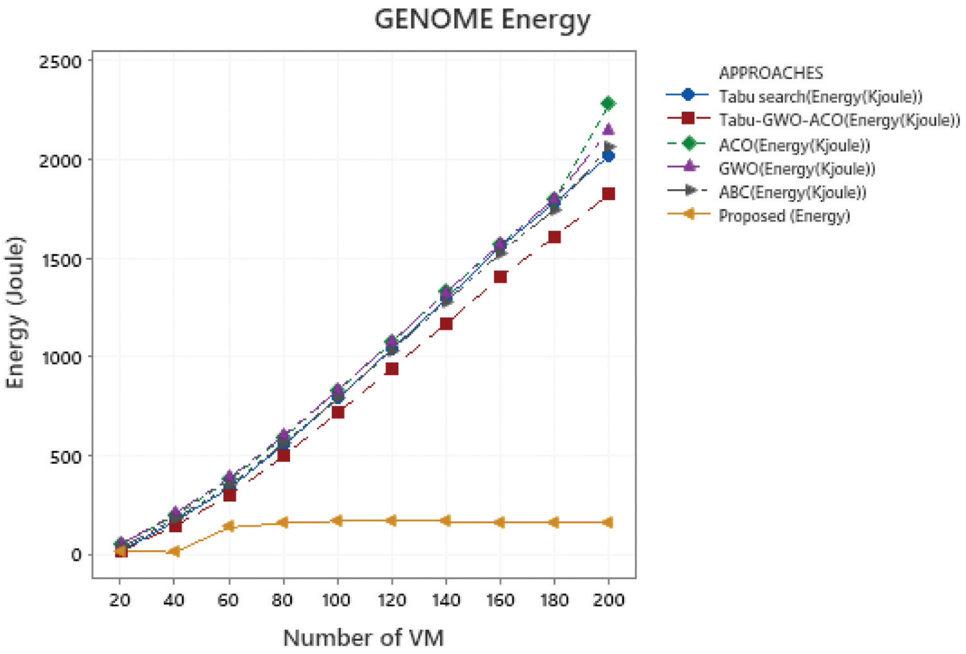


Figure 11. Comparison of Energy parameter of Proposed and Existing approach in GENOME Workflows.



4.5. SIPHT workflows performance parameter analysis

Figures 12-14 show how the proposed approach compares to the existing approach in SIPHT Workflows regarding cost, time, and energy. Compared to all other alternatives, the proposed approach achieves a lower execution cost for each task, followed by Tabu-GWO-ACO, search, ACO, ABC and GWO, almost overlapping each other, and ACO having the highest execution cost among all. On the other hand, Figure 12 also compares the time delay results for various SIPHTs. The proposed approach has less time delay than the other approaches, followed by Tabu-GWO-ACO (Hybrid), ABC, GWO, ACO, and 367 search possessing the highest delay among all. The

Figure 12. Comparison of Time Delay parameter of Proposed and Existing approach in SIPHT Workflows.

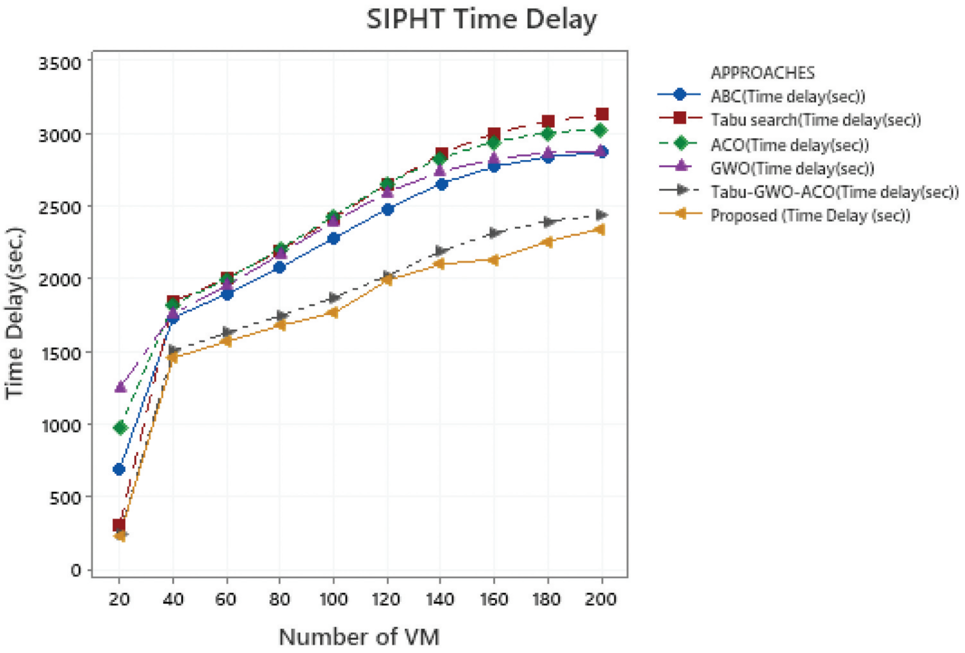
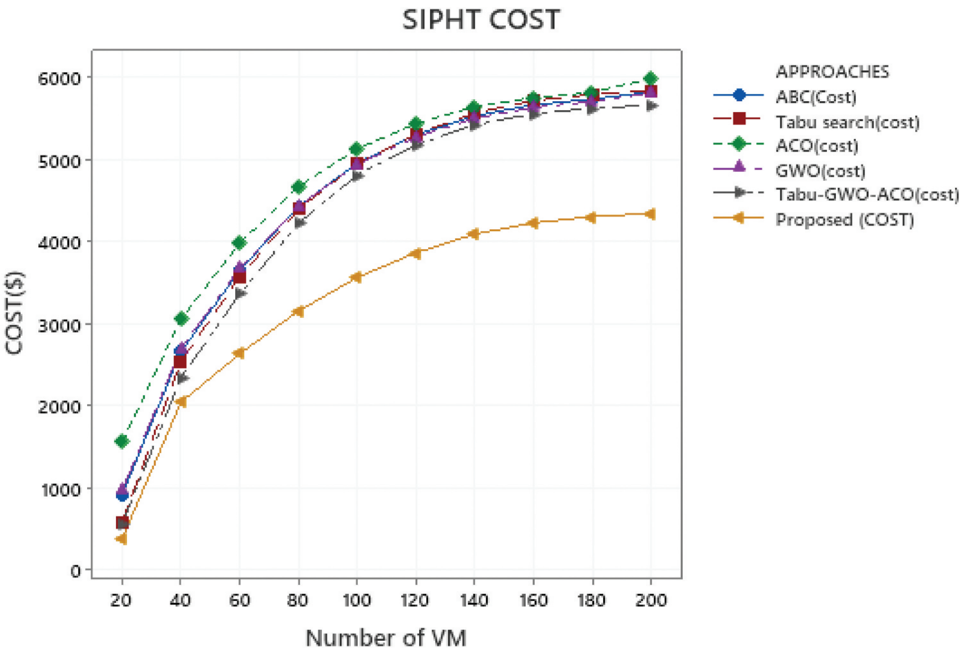


Figure 13. Comparison of Cost parameter of Proposed and Existing approach in SIPHT Workflows.



proposed approach for SIPHT has a 368 lower energy usage, followed by Tabu-GWO-ACO (Hybrid), GWO and ABC (overlapping 369 each other), with ACO having the highest energy usage.

4.6. Average performance analysis of CYBER SHAKE workflow

Figure 15 compares the average performance parameter of the proposed and existing methodologies in shake Workflows. Regarding cost analysis, the proposed solution has a significantly lower average execution cost than competing alternatives. ACO has the highest execution cost when

Figure 14. Comparison of Energy parameter of Proposed and Existing approach in SIPHT Workflows.

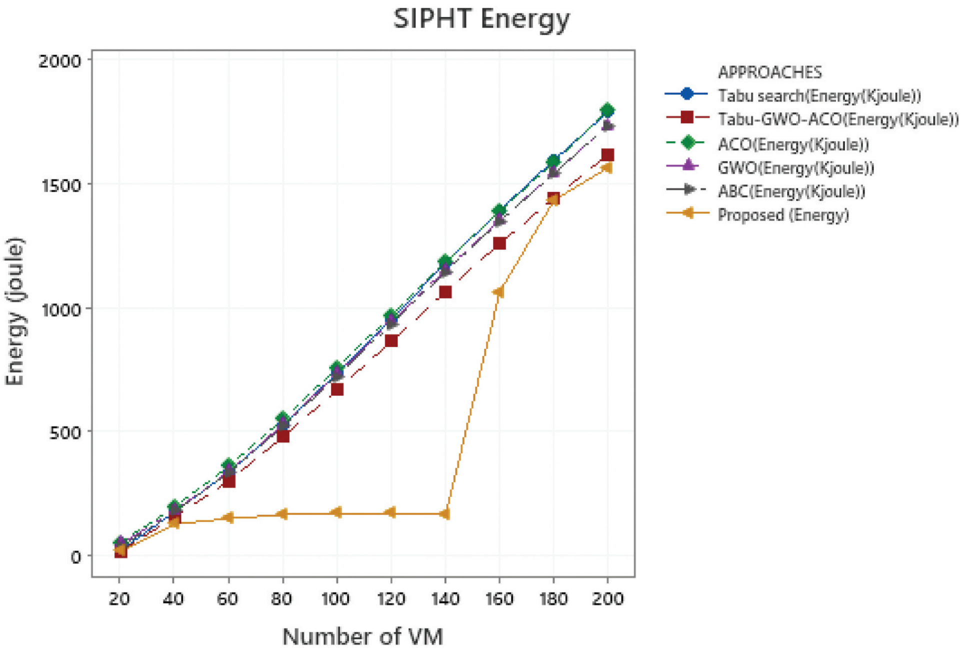
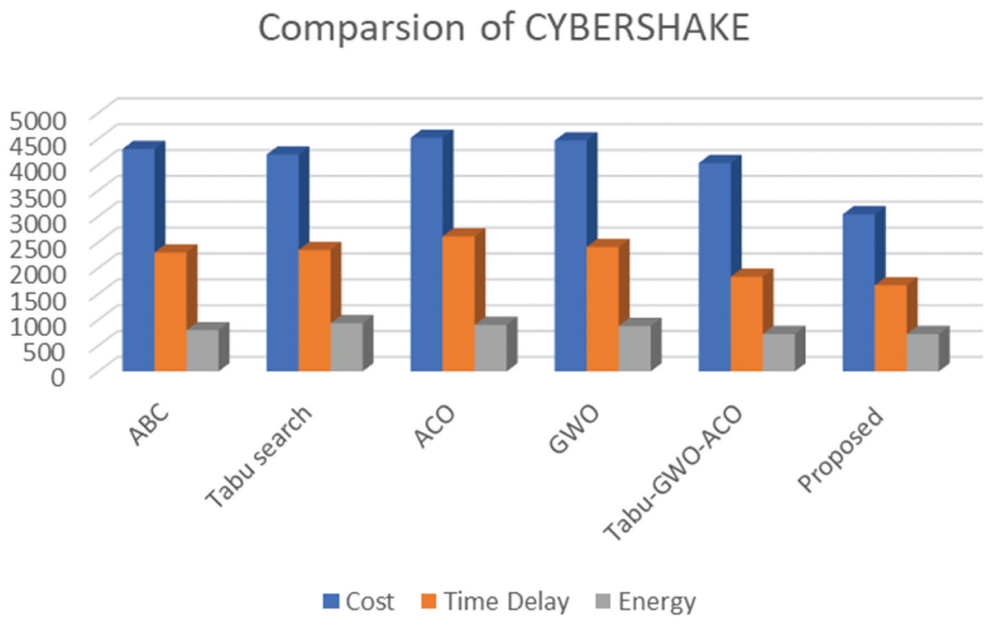


Figure 15. Comparison of average Performance parameter of Proposed and Existing approach in CYBERSHAKE Workflows.



compared to the other methods. The proposed approach performs well in time delay analysis compared to existing methodologies. In this case, ACO and GWO yielded highly similar results. Regarding energy usage, the proposed approach for shaking consumes less than others, although the other results are relatively similar. Overall, the proposed method outperforms the other cost, time, and energy alternatives.

4.7. Average performance analysis of LIGO Workflow

Figure 16 compares the average performance parameter of the proposed and existing methodologies in LIGO Workflows. Regarding cost analysis, the proposed solution has a significantly lower average execution cost than competing alternatives. ACO has the highest execution cost when compared to the other methods. The proposed approach performs well in time delay analysis compared to existing methodologies. Regarding energy consumption, the proposed approach for LIGO uses less energy than others. Overall, the proposed method outperforms the other cost, time, and energy alternatives.

4.8. Average performance analysis of GENOME Workflow

In GENOME Workflows, Figure 17 compares the average performance parameter of the proposed and existing approaches. Regarding cost analysis, the proposed solution has a significantly lower average execution cost than competing alternatives. Compared to the other approaches, GWO has the highest execution cost, which is very close to ACO. In terms of time delay analysis, the proposed strategy outperforms existing methodologies. The GWO and ACO all had pretty similar results in this case. Regarding energy consumption, the proposed approach for GENOME uses less energy than others, although the other results are relatively similar. The proposed solution outperforms the alternatives in terms of cost, time, and energy.

4.9. Average performance analysis of SIPHT Workflow

In SIPHT Workflows, Figure 18 compares the average performance parameter of the proposed and existing approaches. Regarding cost analysis, the proposed solution has a significantly lower average execution cost than competing alternatives. When compared to the other approaches, ACO has the highest execution cost. In terms of time delay analysis, the proposed strategy outperforms existing methodologies. The Tabu search, GWO, and ACO all yielded reasonably

Figure 16. Comparison of average Performance parameter of Proposed and Existing approach in LIGO Workflows.

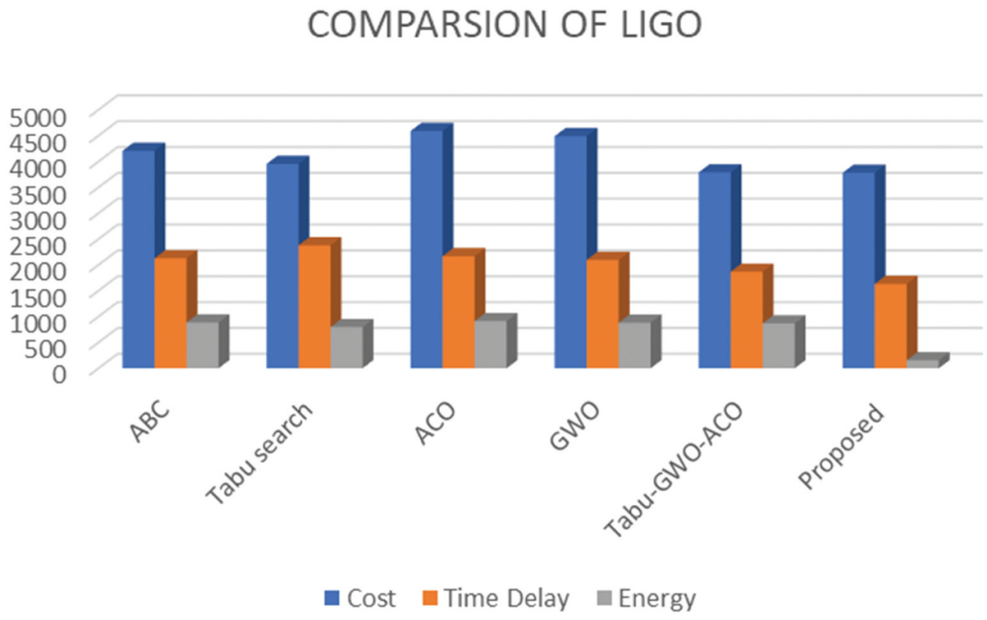
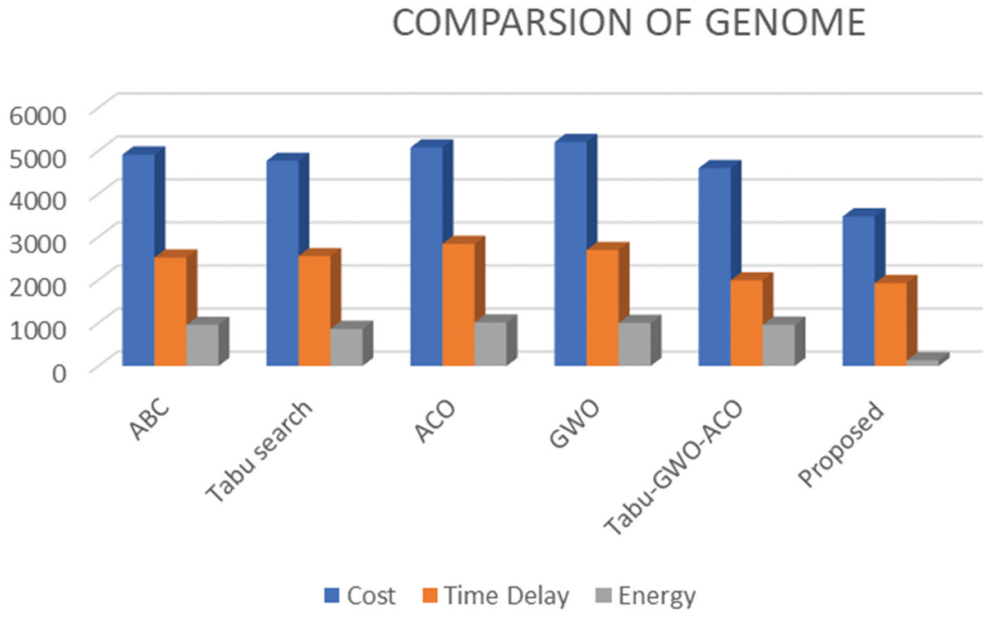


Figure 17. Comparison of average Performance parameter of Proposed and Existing approach in GENOME Workflows.

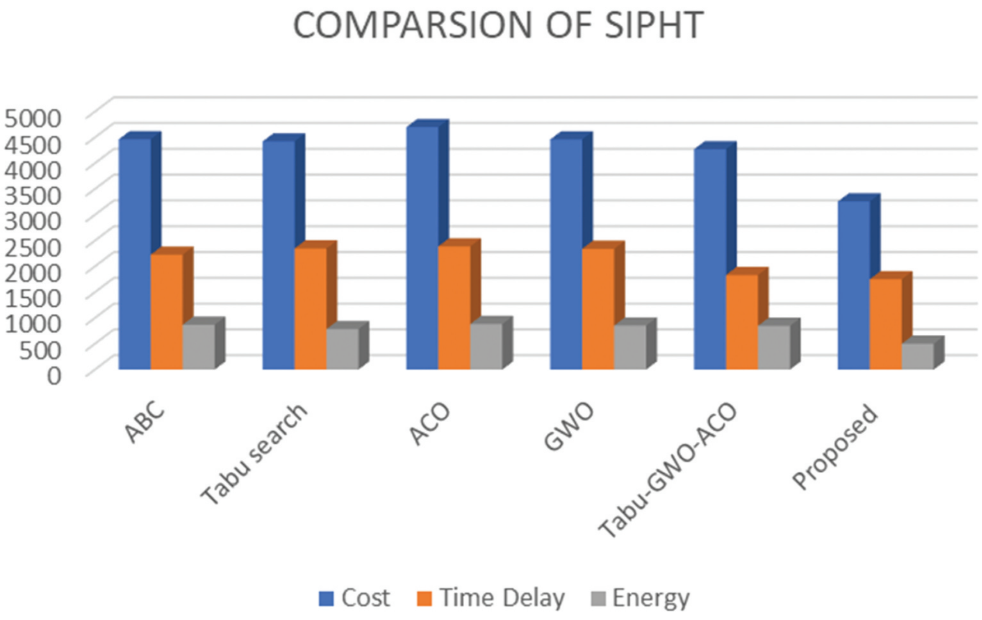


similar results in this case. The proposed approach for SIPHT requires less energy than others in terms of energy usage, although the other results are relatively similar. The proposed solution outperforms the alternatives in terms of cost, time, and energy.

4.10. Observation from experiment results

- This paper optimizes workflow scheduling to maximize resource utilization while reducing cost, time, and energy. The proposed method provides an efficient ranking and finds the optimal Pareto front in multi-objective conditions.

Figure 18. Comparison of average Performance parameter of Proposed and Existing approach in SIPHT Workflows.



- There are four distinct types of scientific workflows. Each workflow complexity is unique, providing efficient validation of the proposed approach.
- Figures 3-5 show cyber shake a comparison of existing methods (Artificial bee colony (ABC), Tabu Search, Ant colony optimization (ACO), Grey Wolves optimization (GWO), and TABU-GWO-ACO (HYBRID)). PPBNG stands for Pareto-based PEFT ranking with a multi-objective genetic algorithm.
- Figures 6-8 compare costs, delays, and energy associated with LIGO Workflows. The comparison shows that PPBNG's proposed approach improves cost, time, and energy in a significant way. It may not improve every time, but it does get better eventually, and 378, the main observation, is that it improves significantly when VM increases. Checking the average performance analysis depicted in Figure 15 reveals a significant improvement in cost (5–6%), time (7–8%), and energy (10–12%).
- In Figures 6-8, a comparison is made between cost, time delay, and energy in LIGO. The comparison demonstrates that the PPBNG-proposed strategy significantly improves cost, time and energy, but not every time; it improves the most when VM increases, which is the critical observation. Checking the average performance analysis depicted in Figure 16 reveals a 3 to 5% improvement in cost, a 7–10% improvement in time, and an 8–9% improvement in energy.
- In Figures 9-11, GENOME Workflows, a comparison is made between cost, time delay, and energy in SIPHT. The comparison demonstrates that the PPBNG-proposed strategy significantly improves cost, time, and energy, but not every time; it improves the most when 390 VM increases, which is the critical observation. Checking the average performance analysis depicted in Figure 17 reveals a considerable improvement in cost (by 2–3%), time (by 5–6%), and energy (by 8–9%).
- In Figures 12-14, SIPHT Workflows, a comparison of cost, time delay, and energy in GENOME is presented. The comparison shows that the PPBNG-proposed strategy 395 improves cost, time, and energy significantly, but not every time; it improves the most when VM increases, which is the critical observation. If you examine the average performance analysis depicted in Figure 18, one can see a considerable improvement in cost of 2 to 4%, time of 9–10%, and energy of 8–12%. In the PEFT approach, tasks are ranked efficiently based on their budgets and dead-lines, but PEFT depends on Objectives such as cost, time, and energy, shown in Equation 1 of the fitness function. Therefore, previous research ignored object-based ranking, but the proposed approach considers objective in PEFT ranking by utilizing PARETO front output.
- The ranking effect provides us with an approximate polynomial search space, and we anticipate relationships using a Bayesian technique to reduce complexity to O(n^2). This facilitates time and space management.
- Following ranking, optimize the conflict objectively and find the optimum threshold to enable efficient VM migration while minimizing cost, time, and energy.

Table 3. Comparison with existing approaches

Authors	Approaches	Time	Cost	Energy
Pillareddy and Karri (2023)	MONWS	213	230	234.23
Arora and Banyal (2022)	PSO-GWO		200	134.45
Bhasker and Murali (2023)	FMMEHO	208	240	180.34
PROPOSED	MPPNSG	100	140	110.23

4.11. Comparison with existing approaches

Table 3 analyses several approaches using the existing and proposed methods. The current work includes studies by various authors utilizing cost, time delay, and energy-based methods. Pillareddy and Karri (2023) opted for the MONWS approach, which yielded a cost value of 213, a time delay of 230, and an energy value of 234.23. Arora and Banyal (2022) selected the PSO-GWO approach, which produced findings with 110 cost values, a time delay of 200, and 204.5 energy values. Similarly, Bacanin et al. (2022) opted for the GOQRFA method, which produced results with a cost value of 278, a time delay of 190, and an energy value of 134.45. Similarly, Bhasker and Murali (2023) used the FMMEHO approach, resulting in a cost value of 208, a time delay of 240, and an energy value of 180.34. In contrast, the proposed framework adopted the MPPNSG approach, which generated outcomes with 100 cost values, a time delay of 140, and 110.23 as the energy value. As indicated by the test results performances, the proposed framework achieved the lowest cost, time delay, and energy consumption compared to all other approaches, indicating improved analysis.

5. Conclusion

This approach improves upon the conventional genetic algorithm by optimizing the Pareto front through Bayesian optimization techniques. The scheduling method that has been proposed is hybrid and is divided into two stages. In the first phase, an optimized Pareto front by Bayesian approach and PEFT ranking of tasks using to generate a list of solutions with low time complexity. In the second phase, an NSG-II was applied to overcome the premature convergence problem of a standard genetic algorithm and work on conflicting objectives. The algorithmic representations and flow charts lay forth the specifics of the activity in an organized fashion. In the methodology, CloudSim simulation tools were used to establish a cloud environment, and then simulation-based experiments were carried out to analyze the planned task. The trials have utilized various workflows and virtual machines, each with its unique combination of performance factors. According to the findings of the experiments, the suggested algorithm can produce optimal solutions while simultaneously reducing execution time, cost, and makespan. The methodology developed by PPBMG results in a 5–6% reduction in costs and an 8% reduction in time delays, respectively. To that end, our approach contributes significantly to the existing literature and we posit that this work shall inspire many works in the future.

Developing new multi objective optimization algorithms that can handle the workflow scheduling problem efficiently and effectively, considering more objectives and constraints, such as security and reliability, and we are integrating machine learning techniques to improve the performance of the algorithms. Overall, the proposed multi objective optimization approach provides a promising framework for addressing the workflow scheduling problem in the cloud and has the potential to benefit various scientific and industrial applications. However, metaheuristic approach to adequately optimize, and further enhancing load optimization in cloud environment is having a great opportunity to be explored, where this work needs further exploration and yet to be identified the best optimization in case of different scenario.

In addition to the advancement in the state-of-the-art algorithm, it would be intriguing how large language models, especially ChatGPT can be productive in optimization problems. As there are many area where researchers are incorporating ChatGPT and related AI tool to solve the problem (Farhat et al., 2023; Rajak, 2022).

Funding
This work was supported by the research.

Author details
Shabina Ghafir[1]
M. Afshar Alam[1]
Farheen Siddiqui[1]
Sameena Naaz[1]
Shahab Saquib Sohail[1]
Dag Øivind Madsen[2]
E-mail: dag.oivind.madsen@usn.no
ORCID ID: http://orcid.org/0000-0001-8735-3332
[1] Department of Computer Science and Engineering, Jamia Hamdard University, New Delhi, India.
[2] USN School of Business, University of South-Eastern Norway, Hønefoss, Norway.

Data availability statement
Data will be provided upon request.

Disclosure statement
No potential conflict of interest was reported by the author(s).

Citation information
Cite this article as: Toward optimizing scientific workflow using multi-objective optimization in a cloud environment, Shabina Ghafir, M. Afshar Alam, Farheen Siddiqui, Sameena Naaz, Shahab Saquib Sohail, Dag Øivind Madsen & Jenhui Chen*,* *Cogent Engineering* (2024), 11: 2287303.

References

Afzal, S., & Kavitha, G. (2019). Load balancing in cloud computing – A hierarchical taxonomical classification. *Journal of Cloud Computing*, *8*(1), 1–24. https://doi.org/10.1186/s13677-019-0146-7

Aktan, M. N., & Bulut, H. (2022). Metaheuristic task scheduling algorithms for cloud computing environments. *Concurrency & Computation: Practice & Experience*, *34*(9), e6513. https://doi.org/10.1002/cpe.6513

Alkhanak, E. N., Lee, S. P., & Khan, S. U. R. (2015). Cost-aware challenges for workflow scheduling approaches in cloud computing environments: Taxonomy and opportunities. *Future Generation Computer Systems*, *50*, 3–21. https://doi.org/10.1016/j.future.2015.01.007

Areeb, Q. M., Nadeem, M., Sohail, S. S., Imam, R., Doctor, F., Himeur, Y., Hussain, A., & Amira, A. (2023). Filter bubbles in recommender systems: Fact or fallacy—A systematic review. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, *13*(6), e1512. https://doi.org/10.1002/widm.1512

Arora, N., & Banyal, R. K. (2022). A particle grey wolf hybrid algorithm for workflow scheduling in cloud computing. *Wireless Personal Communications*, *122*(4), 3313–3345. https://doi.org/10.1007/s11277-021-09065-z

Bacanin, N., Zivkovic, M., Bezdan, T., Venkatachalam, K., & Abouhawwash, M. (2022). Modified firefly algorithm for workflow scheduling in cloud-edge environment. *Neural Computing & Applications*, *34*(11), 9043–9068. https://doi.org/10.1007/s00521-022-06925-y

Belgacem, A., & Beghdad Bey, K. (2022). Multi-objective workflow scheduling in cloud computing: Trade-off between makespan and cost. *Cluster Computing*, *25*(1), 579–595. https://doi.org/10.1007/s10586-021-03432-y

Bezdan, T., Zivkovic, M., Bacanin, N., Strumberger, I., Tuba, E., & Tuba, M. (2022, January 1). Multi-objective task scheduling in cloud computing environment by hybridized bat algorithm. *Journal of Intelligent & Fuzzy Systems*, 42(1), 411–423. https://doi.org/10.3233/JIFS-219200

Bhasker, B., & Murali, S. (2023). FMMEHO based workflow scheduling in virtualized cloud environment for smart irrigation System. *ACM Transactions on Sensor Networks*. https://doi.org/10.1145/3582010

Chawla, Y., & Bhonsle, M. (2012). A study on scheduling methods in cloud computing. *International Journal of Emerging Trends & Technology in Computer Science*, *1*(3), 12–17.

Farhat, F., Chaudry, B. M., Nadeem, M., Sohail, S. S., & Madsen, D. O. (2023). *Evaluating AI models for the national pre-medical exam in India: A head-to-head analysis of ChatGPT-3.5, GPT-4 and Bard*. JMIR Preprints.

Farid, M., Latip, R., Hussin, M., & Abdul Hamid, N. A. W. (2020). A survey on QoS requirements based on particle swarm optimization scheduling techniques for workflow scheduling in cloud computing. *Symmetry (Basel)*, *12*(4), 551. https://doi.org/10.3390/sym12040551

Ghasemi, S., & Hanani, A. (2019). A cuckoo-based workflow scheduling algorithm to reduce cost and increase load balance in the cloud environment. *JOIV: International Journal on Informatics Visualization*, *3*(1), 79–85. https://doi.org/10.30630/joiv.3.1.220

Gupta, I., Gupta, S., Choudhary, A., & Jana, P. K. (2019). A hybrid meta-heuristic approach for load balanced workflow scheduling in IaaS cloud. In *Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics)* (Vol. 11319, pp. 73–89). LNCS. https://doi.org/10.1007/978-3-030-05366-6_6

Gupta, G., & Mangla, N. (2019). Workflow scheduling in cloud computing. *Journal of Computational and Theoretical Nanoscience*, *16*(9), 3965–3968. https://doi.org/10.1166/jctn.2019.8278

Hariri, M., Nouri-Baygi, M., & Abrishami, S. (2022). A hybrid algorithm for scheduling scientific workflows in IaaS cloud with deadline constraint. *The Journal of Supercomputing*, *78*(15), 16975–16996. https://doi.org/10.1007/s11227-022-04563-8

Ismayilov, G., & Topcuoglu, H. R. (2020). Neural network based multi-objective evolutionary algorithm for dynamic workflow scheduling in cloud computing. *Future Generation Computer Systems*, *102*, 307–322. https://doi.org/10.1016/j.future.2019.08.012

Katyal, M., & Mishra, A. (2014). Orchestration of cloud computing virtual resources. *Proceedings of 2014 International Conference on Contemporary Computing and Informatics (IC3I)* (pp. 833–838). IEEE. https://doi.org/10.1109/IC3I.2014.7019756

Kaur, E. A. (2017). Challenges To Task and Workflow Scheduling in Cloud Environment. *International Journal of Advanced Research in Computer Science*, *8* (8), 412–415. https://doi.org/10.26483/ijarcs.v8i8.4752

Kaur, S., Bagga, P., Hans, R., & Kaur, H. (2019). Quality of service (QoS) aware workflow scheduling (WFS) in cloud computing: A systematic review. *Arabian Journal for Science & Engineering*, 44(4), 2867–2897. https://doi.org/10.1007/s13369-018-3614-3

Kaur, P., & Sachdeva, M. (2016). A grouping based scheduling algorithm on load balancing in cloud computing. *International Journal of Control Theory & Applications*, *9*(22), 293–299.

Kaur, R., & Singh Dhindsa, D. K. (2018). Efficient task scheduling using load balancing in cloud computing. *International Journal of Advanced Networking Applications*, *10*(3), 3888–3892. https://doi.org/10.35444/ijana.2018.10037

Kousalya, G., Balakrishnan, P., & Pethuru Raj, C. (2017). Workflow scheduling algorithms and approaches. *Automated Workflow Scheduling in Self-Adaptive Clouds Concepts, Algorithms Methods*, 65–83. https://doi.org/10.1007/978-3-319-56982-6_4

Krishan, R., & Kumar, V. (2020). Optimization of resource aware task-scheduling approaches in cloud computing. *Journal of Green Engineering*, *10*(3), 1077–1096.

Kumar, P., & Kumar, R. (2019). Issues and challenges of load balancing techniques in cloud computing: A survey. *ACM Computing Surveys*, *51*(6), 1–35. https://doi.org/10.1145/3281010

Li, B., Huang, Y., Liu, Z., Li, J., Tian, Z., & Yiu, S. M. (2019). HybridORAM: Practical oblivious cloud storage with constant bandwidth. *Information Sciences*, *479*, 651–663. https://doi.org/10.1016/j.ins.2018.02.019

Li, S., Li, Y., Han, W., Du, X., Guizani, M., & Tian, Z. (2021). Malicious mining code detection based on ensemble learning in cloud computing environment. *Simulation Modelling Practice and Theory*, *113*, 102391. https://doi.org/10.1016/j.simpat.2021.102391

Li, F., Tan, W. J., & Cai, W. (2022). A wholistic optimization of containerized workflow scheduling and deployment in the cloud–edge environment. *Simulation Modelling Practice and Theory*, *118*, 102521. https://doi.org/10.1016/j.simpat.2022.102521

Li, M., Tian, Z., Du, X., Yuan, X., Shan, C., & Guizani, M. (2023). Power normalized cepstral robust features of deep neural networks in a cloud computing data privacy protection scheme. *Neurocomputing*, *518*, 165–173. https://doi.org/10.1016/j.neucom.2022.11.001

Malik, B. H., Amir, M., Mazhar, B., Ali, S., Jalil, R., & Khalid, J. (2018). Comparison of task scheduling algorithms in cloud environment. *International Journal of Advanced Computer Science & Applications*, *9*(5), 384–390. https://doi.org/10.14569/IJACSA.2018.090550

Malik, N., Sardaraz, M., Tahir, M., Shah, B., Ali, G., & Moreira, F. (2021). Energy-efficient load balancing algorithm for workflow scheduling in cloud data centers using queuing and thresholds. *Applied Sciences*, *11*(13), 5849. https://doi.org/10.3390/app11135849

Mangalampalli, S., Karri, G. R., & Kumar, M. (2022). Multi objective task scheduling algorithm in cloud computing using grey wolf optimization. *Cluster Computing*, *26*(6), 1–20. https://doi.org/10.1007/s10586-022-03786-x

Mangalampalli, S., Karri, G. R., & Satish, G. N. (2023). Efficient workflow scheduling algorithm in cloud computing using whale optimization. *Procedia Computer Science*, *218*, 1936–1945. https://doi.org/10.1016/j.procs.2023.01.170

Mangalampalli, S., Swain, S. K., & Mangalampalli, V. K. (2022). Multi objective task scheduling in cloud computing using cat swarm optimization algorithm. *Arabian Journal for Science & Engineering*, *47*(2), 1821–1830. https://doi.org/10.1007/s13369-021-06076-7

Masdari, M., ValiKardan, S., Shahi, Z., & Azar, S. I. (2016). Towards workflow scheduling in cloud computing: A comprehensive analysis. *Journal of Network and Computer Applications*, *66*, 64–82. https://doi.org/10.1016/j.jnca.2016.01.018

Miftakhov, E., Mustafina, S., Akimov, A., Larin, O., & Gorlov, A. (2021). Developing methods and algorithms for cloud computing management systems in industrial polymer synthesis processes. *Emerging Science Journal*, *5*(6), 964–972. https://doi.org/10.28991/esj-2021-01324

Mirmohseni, S. M., Tang, C., & Javadpour, A. (2022, December). FPSO-GA: A fuzzy metaheuristic load balancing algorithm to reduce energy consumption in cloud networks. *Wireless Personal Communications*, *127*(4), 2799–2821. https://doi.org/10.1007/s11277-022-09897-3

Naaz, S., Alam, A., & Biswas, R. (2012). Load balancing algorithms for peer to peer and client server distributed environments. *International Journal of Computer Applications*, *47*(8), 17–21. https://doi.org/10.5120/7208-9995

Nagadevi, S., Satyapriya, K., & Malathy. (2013). A survey on economic cloud schedulers for optimized task scheduling. *International Journal of Advanced Engineering and Technology*, *5*, 58–62.

Naz, I., Naaz, S., Agarwal, P., Alankar, B., Siddiqui, F., and Ali, J. (2023). A genetic algorithm-based virtual machine allocation policy for load balancing using actual asymmetric workload traces. *Symmetry (Basel)*, *15*(5), 1025. https://doi.org/10.3390/sym15051025

Nirmala, S. J. and Bhanu, S. M. S. (2016). Catfish-PSO based scheduling of scientific workflows in IaaS cloud. *Computing*, *98*(11), 1091–1109. https://doi.org/10.1007/s00607-016-0494-9

Patel, S., & Bhoi, U. (2013). Priority based job scheduling techniques in cloud computing: A systematic review. *International Journal of Scientific & Technology Research*, *2*(11), 147–152.

Pillareddy, V. R., & Karri, G. R. (2023). MONWS: Multi-objective normalization workflow scheduling for cloud computing. *Applied Science*, *13*(2), 1101. https://doi.org/10.3390/app13021101

Qin, S., Pi, D., Shao, Z., & Xu, Y. (2022). Hybrid collaborative multi-objective fruit fly optimization algorithm for scheduling workflow in cloud environment. *Swarm and Evolutionary Computation*, *68*, 101008. https://doi.org/10.1016/j.swevo.2021.101008

Qin, S., Pi, D., Shao, Z., Xu, Y., & Chen, Y. (2023). Reliability-aware multi-objective memetic algorithm for workflow scheduling problem in multi-cloud System. *IEEE Transactions on Parallel and Distributed Systems*, *34*(4), 1343–1361. https://doi.org/10.1109/TPDS.2023.3245089

Rajak, A. A. (2022). Emerging technological methods for effective farming by cloud computing and IoT. *Emerging Science Journal*, *6*(5), 1017–1031. https://doi.org/10.28991/ESJ-2022-06-05-07

Rodriguez, M. A., & Buyya, R. (2014). Deadline based resource provisioning and scheduling algorithm for scientific workflows on clouds. *IEEE Transactions on Cloud Computing*, *2*(2), 222–235. https://doi.org/10.1109/TCC.2014.2314655

Shafiq, D. A., Jhanjhi, N. Z., & Abdullah, A. (2022). Load balancing techniques in cloud computing environment: A review. *Journal of King Saud University - Computer and Information Sciences*, *34*(7), 3910–3933. https://doi.org/10.1016/j.jksuci.2021.02.007

Shah, J. M., Kotecha, K., Pandya, S., Choksi, D. B., & Joshi, N. (2018). Load balancing in cloud computing: Methodological survey on different types of algorithm. *Proceedings of the 2017 International conference on trends in electronics and informatics* (*ICEI 2017*) (vol. 2018, pp. 100–107). IEEE. https://doi.org/10.1109/ICOEI.2017.8300865

Shameer, A. P., & Subhajini, A. C. (2017). Optimization task scheduling techniques on load balancing in cloud using intelligent bee colony algorithm. *International Journal of Pure and Applied Mathematics: IJPAM*, *116*(22), 341–352.

Shishido, H. Y., Estrella, J. C., Toledo, C. F. M., & Arantes, M. S. (2018). Genetic-based algorithms applied to a workflow scheduling algorithm with security and deadline constraints in clouds. *Computers & Electrical Engineering*, *69*, 378–394. https://doi.org/10.1016/j.compeleceng.2017.12.004

Singh, R., & Singh, S. (2013). Score based deadline constrained workflow scheduling algorithm for cloud systems. *International Journal on Cloud Computing: Services and Architecture*, *3*(6), 31–41. https://doi.org/10.5121/ijccsa.2013.3603

Sohail, S. S., Farhat, F., Himeur, Y., Nadeem, M., DØ, M., Singh, Y., Atalla, S., & Mansoor, W. (2023). Decoding ChatGPT: A taxonomy of existing research, current challenges, and possible future directions. *Journal of King Saud University-Computer & Information Sciences*, *35*(8), 101675. https://doi.org/10.1016/j.jksuci.2023.101675

Sohail, S. S., Javed, Z., Nadeem, M., Anwer, F., Farhat, F., Hussain, A., Himeur, Y., & DØ, M. (2023). Multi-criteria decision making-based waste management: A bibliometric analysis. *Heliyon*, *9*(11), e21261. https://doi.org/10.1016/j.heliyon.2023.e21261

Sreenu, K., & Sreelatha, M. (2019). W-Scheduler: Whale optimization for task scheduling in cloud computing. *Cluster Computing*, *22*(1), 1087–1098. https://doi.org/10.1007/s10586-017-1055-5

Thammarak, K., Sirisathitkul, Y., Kongkla, P., & Intakosum, S. (2022). Automated data digitization System for vehicle registration certificates using Google cloud vision API. *Civil Engineering Journal*, *8*(7), 1447–1458. https://doi.org/10.28991/CEJ-2022-08-07-09

Verma, A., & Kaushal, S. (2017). A hybrid multi-objective particle swarm optimization for scientific workflow scheduling. *Parallel Computing*, *62*, 1–19. https://doi.org/10.1016/j.parco.2017.01.002

Vijayalakshmi, R., & Vasudevan, V. (2015). Static batch mode heuristic algorithm for mapping independent tasks in computational grid. *Journal of Computational Science*, *11*(1), 224. https://doi.org/10.3844/jcssp.2015.224.229

Wang, Y., Guo, Y., Guo, Z., Baker, T., & Liu, W. (2020). CLOSURE: A cloud scientific workflow scheduling algorithm based on attack–defense game model. *Future Generation Computer Systems*, *111*, 460–474. https://doi.org/10.1016/j.future.2019.11.003

Wu, F., Wu, Q., & Tan, Y. (2015). Workflow scheduling in cloud: A survey. *The Journal of Supercomputing*, *71*(9), 3373–3418. https://doi.org/10.1007/s11227-015-1438-4

Zhao, L., Ren, Y., & Sakurai, K. (2013). Reliable workflow scheduling with less resource redundancy. *Parallel Computing*, *39*(10), 567–585. https://doi.org/10.1016/j.parco.2013.06.003

Zhou, C., Wang, T., Li, L., Sun, J., & Zhou, J. (2022). Makespan and security-aware workflow scheduling for cloud service cost minimization using firefly optimizer. *Proceedings of the International Conference on Algorithms and Architectures for Parallel Processing*, Copenhagen, Denmark (pp. 620–641).

Zivkovic, M., Bezdan, T., Strumberger, I., Bacanin, N., & Venkatachalam, K. (2021). Improved harris hawks optimization algorithm for workflow scheduling challenge in cloud–edge environment. *Computer Networks, Big Data and IoT: Proceedings of ICCBI 2020*. (pp. 87–102). Springer Singapore.