An Energy-Efficient Load Balancing Approach for Scientific Workflows in Fog Computing

Article *in* Wireless Personal Communications · August 2022

DOI: 10.1007/s11277-022-09724-9

2 authors:

Mandeep Kaur
Chitkara University
9 PUBLICATIONS **101** CITATIONS
SEE PROFILE

Rajni Aron
Narsee Monjee Institute of Management Studies
30 PUBLICATIONS **226** CITATIONS
SEE PROFILE

Wireless Personal Communications
https://doi.org/10.1007/s11277-022-09724-9

An Energy-Efficient Load Balancing Approach for Scientific Workflows in Fog Computing

Mandeep Kaur[1,2] · **Rajni Aron**[3]

Accepted: 14 April 2022

Abstract
Fog computing provides cloud services at the user end. User requests are processed on the fog nodes deployed near the end-user layer in a fog computing environment. Fog computing can play an essential role in executing parallel computational tasks, i.e. scientific workflow applications. To run workflow applications on the cloud datacenters may take more time. So, fog computing can help the cloud reduce latency issues these applications face during execution. This paper proposes a resource-utilization based workflow execution model for fog computing has been proposed. The proposed model executes workflow tasks by balancing the load among all available resources. Along with this, a PSW-Fog Clustering algorithm proposed to reduce the energy consumption, computational cost, and execution time while executing scientific workflow applications in a fog-cloud environment. The proposed approach is the hybridization of plant growth optimization, simulated annealing, and water cycle optimization approaches and is named using the first letter of each approach. iFogSim toolkit has been used to evaluate the proposed algorithm. The performance of the PSW-Fog clustering-based load balancing approach has been compared with traditional approaches, and obtained that this approach outperforms other approaches in terms of time delay, energy consumption, and cost. The article has been concluded, and some challenges for future research have been discussed.

Keywords Fog computing · Cloud computing · Resource utilization · Energy consumption · Load balancing

✉ Mandeep Kaur
saroamandeep1@gmail.com; k.mandeep@chitkara.edu.in

Rajni Aron
rajni@nmims.edu

[1] Lovely Professional University, Jalandhar, Punjab, India

[2] Chitkara University Institute of Engineering and Technology, Chitkara University, Rajpura, Punjab, India

[3] SVKM's Narsee Monjee Institute of Management Studies (NMIMS) University, Mumbai, Maharashtra, India

Published online: 12 May 2022

1 Introduction

In the Internet of Things (IoT), data is the primary concern that needs more attention. IoT has usually connected to the cloud, where they can store their data and get its processing done. In some cases, where users hurry to access data, they might have to wait for a long time due to system or network problems. So, there was a need for some platforms to reach IoT users. Hence, Cisco brought fog computing into existence, the next generation of computing technology. It resembles the cloud by providing similar services like cloud, i.e., networking, storage, and computing as well [5]. The main difference between cloud and fog computing is memory limitation in fog. Fog can provide services like a cloud, but in a limited form, it cannot store a large amount of data at a time due to its less storage capacity. The fog layer has the networking devices like routers, switches and gateways that act as computing nodes but have less storage and computing capacity. Fog computing came up with the features that can help advance intelligent society's growth. Fog computing supports different applications like smart agriculture, competent healthcare, innovative waste management, innovative water management, intelligent traffic management [35, 41].

With the deployment of fog nodes at the network end, users can experience better services than cloud computing. Fog computing contains valuable features for users like mobility support, fog node location known to the end-users, several geographically distributed fog nodes. Fog computing has to face the load balancing problem with valuable features and limited storage capacity. Many internet users who are using smart devices keep sending data simultaneously, due to which there arises a shortage of resources. Sometimes only a few resources are utilized, and the others in the fog layer remain unused, hence wastage of resources and power to keep them on. Load balancing becomes a challenging task in the fog computing layer to reduce the cost and energy usage. With the imbalance in load in the fog layer, bandwidth is also wasted, which provide less throughput and response time to the user is increased. All this happens due to a highly restricted environment, and limited resources availability [21].

Fog computing is the most trending technology in the Internet of Things (IoT) nowadays. The fog has removed the barriers of computing and storing IoT data at cloud datacentres by providing local storage and processing services. Fog computing brings computing and storage services local to the end-users and enhances the popularity of IoT. The system has to decide where the applications have to execute, i.e., in the fog layer or the cloud, to fulfil the quality of the service requirements. A cloud-fog scheduler should be installed to make the system's execution decision to avoid delays in task processing. Load balancing plays a vital role to enhance the performance of a fog computing system. Due to the distributed nature of the fog environment, load balancing becomes a very challenging task. The load distribution mechanism becomes difficult due to more users' presence, which leads to load fluctuation in the fog environment. For maximum utilization of resources in a fog environment, the load should be distributed among all available VMs to avoid overloading and underloading resources in the fog computing layer.

Scientific workflows are data-intensive applications representing distributed data sources and complex computations in various domains, i.e. astronomy, engineering sciences, and bioinformatics. In distributed environments like fog computing, various sensors and experimental processes generate a large volume of data that need collection and processing within specific time constraints. Geographically distributed fog resources can be used to collect and process this data. Although fog computing has numerous advantages over cloud computing, it also faces many challenges [10]. One challenge is load balancing

in scientific workflow task execution in the complex resources environment. Scientific workflow tasks require real-time execution, but fog computing resources can be overloaded due to the large volume of data. Hence there is a need to balance the data among existing resources in equal proportion to be processed in real-time. Even distribution of tasks among all the resources can result in proper utilization of resources, hence saving energy and execution time also [42].

Scientific workflows are considered NP-complete problem that includes a series of computational tasks of data of any scientific application. Directed Acyclic Graph(DAG) is the representation of scientific workflows, that contains set of vertices $(V_1, V_2, V_3, \ldots V_n)$, and edges$(E_1, E_2, E_3, \ldots E_n)$. In DAGs, vertices represents the different workflow tasks which are mapped to the corresponding VM_s i.e. $(VM_1, VM_2, VM_3, \ldots VM_n)$. The edges represent the communication between different tasks $T_1, T_2, T_3, \ldots T_n$. DAG is represented in a tree containing nodes and edges joining them. These edges are weighted with communication and computation time. Different types of workflows can be applied to fog computing for their execution. Figure 1 below shows the example of simple workflows: In this section we have defined different type of common workflows such as Montage [4], Sipht and CyberShake, [15], Ligo and Epigenomics [12]. These different workflows are explained as below:

A montage workflow application is used for astronomic applications, which is used to build substantial picture mosaics of the sky. Montage tasks can be identified into input and output, which do not use more CPU processing capability [4]. CyberShake workflow is used to identify natural disasters, i.e. earthquakes. It is classified as a data concentrated workflow which consumes a large amount of memory CPU capability [15]. Sipht is used in the national centre to detect replicates of all bacteria used to collect biotechnology information. These types of workflows need more CPU utilization but less input-output utilization. Laser Interferometer Gravitational-Wave Observatory (LIGO) is the workflow application used for earth's gravity detection. Ligo workflows contain those kinds of tasks that take more memory for execution in CPU [12]. Epigenomics workflows are used to detect the production of DNA. They are data incentives hence more CPU utilization. They are being used for genome sequencing operations in epigenome centre [12, 22].

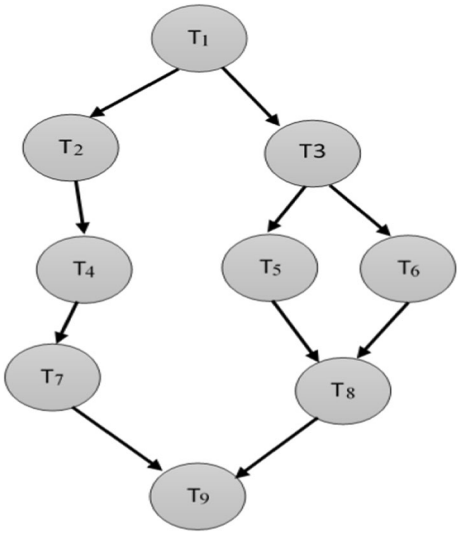


Fig. 1 Example of workflow

1.1 Motivation

Fog computing faces many challenges while executing complex workloads, which generate load imbalance in a fog environment. Load balancing means probabilistic computations that distribute workload among all Virtual Machines (VM) and generate a self-learning framework that needs to act as an algorithm capable of solving complex computational problems. The challenges faced by fog includes QoS, cost, and energy consumption related challenges that need an efficient energy-aware approach for load balancing. The proposed work has been inspired by all these facts explained before. Many existing load balancing approaches are provided for scientific workflows, but they do not consider fog computing for their implementation. This work implemented scientific workflow applications and reduced fog nodes' cost, time delay, and energy consumption.

1.2 Our Contribution

Contributions of this article are explained as given below:

- The article proposes a resource-utilization based workflow execution model for fog computing. This model helps to apply load balancing to enhance resource utilization in fog computing.
- This article proposes a hybrid load balancing approach PSW-Fog Clustering that tries to reduce energy consumption in a fog computing environment. The work considers both the availability of tasks and workflow structure. The work tries to reduce system overheads, minimize resource wastage, and enhance tasks' execution speed.
- A framework has been proposed for the implementation of workflows. The proposed work considers scientific workflows for execution in fog environments, i.e. LIGO, Genome, SIPHT, Cybershake workflows.

1.3 Paper Organisation

The article's remaining sections are as follows—A thorough literature review has been done and explained in Sect. 2. Sect. 3 proposes a resource-utilization based workflow execution model for fog computing. Furthermore, Sect. 4 PSW-Fog Clustering-based load balancing algorithm has been proposed. This section is divided into subsections that describe optimization approaches and the working of the proposed algorithm. This section also explains the considered parameters for checking the performance of the proposed PSW approach. Experimental results and performance analysis are explained in Sect. 5. The proposed approach has been evaluated, and the obtained results are compared with traditional approaches. The article has been concluded in the Sect. 6. This section also provides the future directions.

2 Related Work

Fog computing is the wide area that has been explored by many researchers. A thorough literature review has been conducted that includes managing load balancing and task scheduling in workflows that has been represented in this section. Many types of research work provided different techniques for scheduling the workflows, but load balancing still

needs more exploration. Load balancing techniques must be provided for proper resource utilization in fog computing while processing scientific workflows. This section represents various approaches proposed by different researchers for load balancing in fog computing. We have divided the related work into three parts, i.e. cost-based, resource-utilization based, and energy-aware load balancing approaches. Following is the review of some recent research works in fog computing:

2.1 Cost-Based Load Balancing

Xie et al. [42] formalized scheduling problems in business workflow applications and proposed a novel PSO based approach named DNCPSO to decrease the cost and makespan. The proposed algorithm is an improved PSO form that applies a directional search process to select data and mutation operations. In their experimental work, authors executed different workflows by considering various factors, cost, and time and compared them with the existing approaches to prove them better. Li et al. [24] have proposed an algorithm for workflow scheduling in cloud resources based upon load balancing. They offered a model for workflow scheduling in the distributed cloud environment. This system model helps to reduce the system's response time while executing workflows. The authors proposed a workflow scheduling algorithm based upon the shortest path technique to reduce the execution time of tasks and energy consumption in cloud data centres. They developed social media applications and considered live video applications of workflows to implement their proposed scenario.

Rizvi et al. [33] proposed a workflow scheduling policy to reduce the computational cost and execution time; they named it a fair budget scheduling algorithm. They implemented different scientific workflows and compared their results with their proposed technique. To prove the efficacy of their approach, obtained results have been verified through the ANOVA test. De Maio et al. [10] proposed a multi-objective workflow offloading approach called MOWO for task distribution in the fog environment. Their proposed approach's main objective is to reduce execution time, enhance reliability, and reduce financial costs. The authors considered real-world workflows for execution and obtained results, i.e. meteorological, biomedical, and astronomy workflows. The proposed approach is compared to the existing approach HEFT and reduced response time to 30%.

Tellez et al. [40] proposed tabu search based algorithm to implement load balancing in a fog-cloud environment. They have considered two cost functions, one for computational cost in fog and the other for the cloud. The proposed approach tabu search ensure online calculations in the fog layer to ensure increased processing of tasks at the fog layer itself. Beraldi et al. [3] proposed two load balancing algorithms to solve problems of resource management. For evaluating their proposed approach, the authors used fog nodes populations that are different in number, configuration, and processing power. They combined simulation and mathematical model-based approaches to verify their proposed approach's performance—the proposed approach reduced response time to 19% and a loss rate to 0.2%.

2.2 Resource-Utilization Based Load Balancing

Javadzadeh, Ghazaleh et al. [19] provided a systematic review of existing literature by studying existing approaches in fog computing. According to the authors, fog computing is the best solution for the limitations of cloud computing. Singh, Simar Preet [39] proposed a

fuzzy-based load balancer to reduce the resource wastage in fog computing. They also provided a fuzzy-based three-tier model for the load balancer based on the software-defined distribution of tasks. They have used both theoretical as well as empirical experiments. They tried to improve resource utilization and reduce costs.

Ding, Ruimiao et al. [11] proposed a scheduling approach based upon Particle Swarm Optimization (PSO) and Min–Min strategy. They define different workflow models, i.e. time and cost models, based on resource cost and execution time in a fog computing environment. The fitness function is used to calculate the execution cost of workflow implementation. For the simulation results, java JDK 1.7 has been used. The work's main focus is to reduce the implementation cost by finishing the tasks before their deadline. Elsherbiny, Shaymaa et al. [12] proposed Intelligent Water Drop (IWD) based algorithm for workflows scheduling in the cloud environment. They have compared their makespan with other approaches by executing workflows in the Workflow simulator. De Falco et al. [9] proposed extremal optimization (EO) based load balancing approach. In the proposed EO, task migrations are done during the load balancing process. The authors considered a few factors in evaluating EO's performance, i.e. fitness function and target nodes. The authors compared their proposed approach to their previously proposed approach, i.e. sequential-extremal optimization algorithms.

Liao et al. [26] developed a framework for optimizing training task distribution by considering communication cost and physical computing to reduce data transfer error at every device. They considered machine learning for their experimental evaluation and proved that their proposed network-aware approach reduces model training cost and enhance accuracy. The authors implement their synthetic and real-world data experiments to confirm network resource utilization improvement by their proposed algorithm.

Kaur et al. [22] proposed a resource utilization based model and named it as FOCALB. FOCALB is basically designed to enhance maximum utilization of resources. The proposed model implements different scientific workflow applications on iFogSim, and reduced the time delay, cost, and energy consumption in fog nodes. The proposed Tabu-GWO-ACO approach was applied to enhance all considered parameters.

2.3 Energy-Aware Load Balancing

Shahid, Muzammil Hussain et al. [37] proposed load balancing and content filtering based energy-aware mechanisms. In their proposed approach, they applied load distribution among fog nodes randomly. Then content filtration is done on active nodes. The proposed load balancing algorithm helps to increase the efficiency of the system. Kaur, Mandeep et al. [21] proposed a load balancing technique based on equal workload distribution. The provided approach is implemented using a cloud analyst tool, and results are compared with round-robin and throttled load balancing techniques. The article tries to reduce the implementation cost in the fog environment and improve resource utilization.

Saroa, Mandeep Kaur et al. [35] proposed architecture for innovative application in fog computing, i.e., intelligent waste management systems. The authors also discussed different fog computing applications. They also studied load balancing in a fog environment and compared the existing techniques. Wadhwa, Heena et al. [41] proposed additional resource provisioning and scheduling techniques in a fog environment. The authors studied fog computing-related to other models and proposed fog computing architecture for E-healthcare. It also provides IoT challenges and their solutions through fog computing.

Choudhary, Anita et al. [8] has proposed a VM placement based energy-aware load balancing algorithm. Their proposed approach uses a task clustering approach to reduce energy consumption in cloud data centres. The proposed approach is based on a min–min algorithm and combines smaller tasks into large tasks to reduce the burden of virtual machines. Kaur et al. [20] proposed an energy-aware load balancing technique for the fog computing environment. They considered scientific workflow applications to execute in fog computing using iFogSim. The proposed approach tries to enhance resource utilization by reducing latency and energy consumption in fog nodes. Naha [30] proposed an energy-aware resource allocation method based on multiple linear regression. The proposed approach tries to reduce failures that occur in fog computing because of energy constraints. Along with this, an energy-aware framework has been proposed to execute different applications in fog. The proposed approach has been compared with other approaches, reducing execution and processing time.

The following Table 1 provides a review of existing load balancing and scheduling techniques in workflows. Existing approaches have been compared based on their execution environment, objectives, performance metrics considered for evaluation, and the experiment's application.

3 Resource-Utilization Based Workflow Execution Model for Fog Computing

This section proposes a workflow execution model for a fog computing environment based on resource utilization. While executing larger computational tasks, fog computing faces specific problems like load scheduling and load balancing. Our proposed solution will help to enhance resource utilization and reduce energy consumption in fog nodes. Figure 2 shows the proposed workflow execution model for the fog computing environment. The proposed architecture has three layers as that of traditional fog architecture. As shown in the Fig. 2, five steps describe the working of this proposed model. These steps are described below in the layer-wise format:

End-user Layer The very first layer is the end-user layer in which end users generate a large number of workflow tasks. Workflow container stores these tasks for some time and then assign these tasks to the workflow scheduler. The working of these steps is as follows:

Step 1 Workflow container submit workflow tasks to the workflow scheduler. These tasks are submitted in the manner they arrived in the workflow container.

Step 2 Workflow scheduler contains a queue where tasks wait for resources. When they arrived, the workflow tasks entered into the queue from the queue's rear end and were removed from the queue from the front end.

Step 3 As the resources become available, these tasks are removed from the task queue and assigned to the fog layer's central controller layer.

Fog Layer This is the second layer of the proposed workflow execution model. This layer contains different fog clusters that contain various fog nodes. This layer also contains a central controller that controls the working of these fog clusters. The central controller checks the availability of fog codes in each cluster and assigns the workflow tasks to the available nodes. Working of this layer is explained in the following step:

Table 1 Traditional load balancing techniques

Year	Author	Environment	Objective	Performance metrics	Applications
2019	Ding, Ruimiao et al. [11]	Java with JDK 1.7	To provide cost-effective scheduling policy for multi-workflow	Execution cost and time	Heart rate monitoring
2019	Li, Chunlin et al. [24]	WorkflowSim simulator	To provide workflow scheduling based on load balancing to utilize the resources of cloud efficiently	Execution Time, System Performance, Cost	Real live video
2020	Rizvi et al. [33]	CloudSim simulator	To schedule the task fairly and minimized the makespan to satisfy finance constraints	Computation cost and makespan	Amazon's EC2
2017	Choudhary, Anita et al. [8]	WorkflowSim simulator	To provide an energy-aware load balancing technique for workflows in the cloud environment.	Energy consumption	Scientific applications
2017	Elsherbiny, Shaymaa et al. [12]	WorkflowSim simulator	To develop the workflow scheduling algorithm for meta-heuristics	System performance and scheduling cost.	Common workflows, i.e. sipht, cyber share etc.
2020	Serhani, Adel et al. [36]	Self-adapting cloud service orchestration	Proposed an architecture for end to end workflow management support	CPU utilization, storage	IoT workflows and e-Health monitoring
2020	De Maio et al. [10]	Multi-objective workflow offloading (MOWO)	To propose an efficient workflow offloading approach for fog computing	Response time, financial cost, and reliability	Real-world workflows
2020	Ying et al. [42]	Docker Swam Cluster along with PostgreSQL	To propose an architecture to support workflow management	QoS parameters i.e. cost and time, Scalability	Health monitoring
2021	Ijaz et al. [18]	MATLAB	To propose a novel energy-aware workflow scheduling model to optimize makespan and energy-consumption in fog environmenr.	Makespan, and energy-consumption	Real world workflow applications

Step 4 The central controller receives the workflow tasks from the workflow scheduler. It contains the load balancer that continuously monitors all fog nodes in all fog clusters. Here PSW-Fog clustering-based load balancing approach has been applied to distribute the tasks among all available nodes in equal proportion. In each fog cluster, various fog nodes contain VM that execute these workflow tasks. The tasks with the highest priority are executed first. The other low-priority tasks can wait for the processor and assign to the cloud layer for processing and further storage. The fog layer executes real-time tasks with limited time and requires an immediate processor. Load balancer assigns such tasks to the available VMs, and users respond after processing tasks.

Cloud Layer This is the third layer of the proposed workflow execution model. This layer contains large data centres that can store and process a large amount of data. Further, this layer contains various computing, networking and storage resources.

Step 5 After processing at fog layer, workflow tasks processing results are informed to users, and these tasks are further sent to the cloud layer for storage and more processing if required.

The proposed workflow execution model tries to reduce the execution time of resources. With the reduction of execution time, the number of resources required will also reduce these resources' energy consumption. So, our proposed solution can enhance maximum resource utilization and minimize energy consumption.

4 PSW-Fog Clustering-Based Load Balancing Algorithm

In this section, a Fog-Clustering based Load balancing algorithm has been proposed for executing workflow datasets. While executing workflow datasets, a load balancing problem is raised due to the fog computing layer's lesser storage and computing capacity. Due to this, a few VM in the fog layer becomes overloaded with the tasks, and others still wait for the tasks. Along with this, VMs consume more energy even if they are free. An efficient load balancing approach is required to reduce the overloading of VMs. This section contains the methodology and algorithm proposed in this work. This section contains three subsections, i.e. proposed methodology, proposed algorithm, and performance matrices.

4.1 Optimization Approaches Used in Our Proposed Hybrid Algorithm

There are various meta-heuristic optimization approaches available to find near-optimal solutions to complex computational problems that can not be solved using a single method. Using a single approach, we may not find the required optimum solution within the defined problem's time constraints. There are many natural phenomena available that different researchers use to solve many different optimization problems. In this paper, we tried solving the load balancing problem faced in fog computing during the execution of sizeable scientific workflow computations. So, only one single optimization approach may not find one optimum solution. Hence we combined different natural phenomenon based optimization approaches, i.e. plant growth optimization, simulated annealing algorithm, and water cycle optimization. All these approaches are explained in this section. Here Table 2 represents different notations used in this paper.

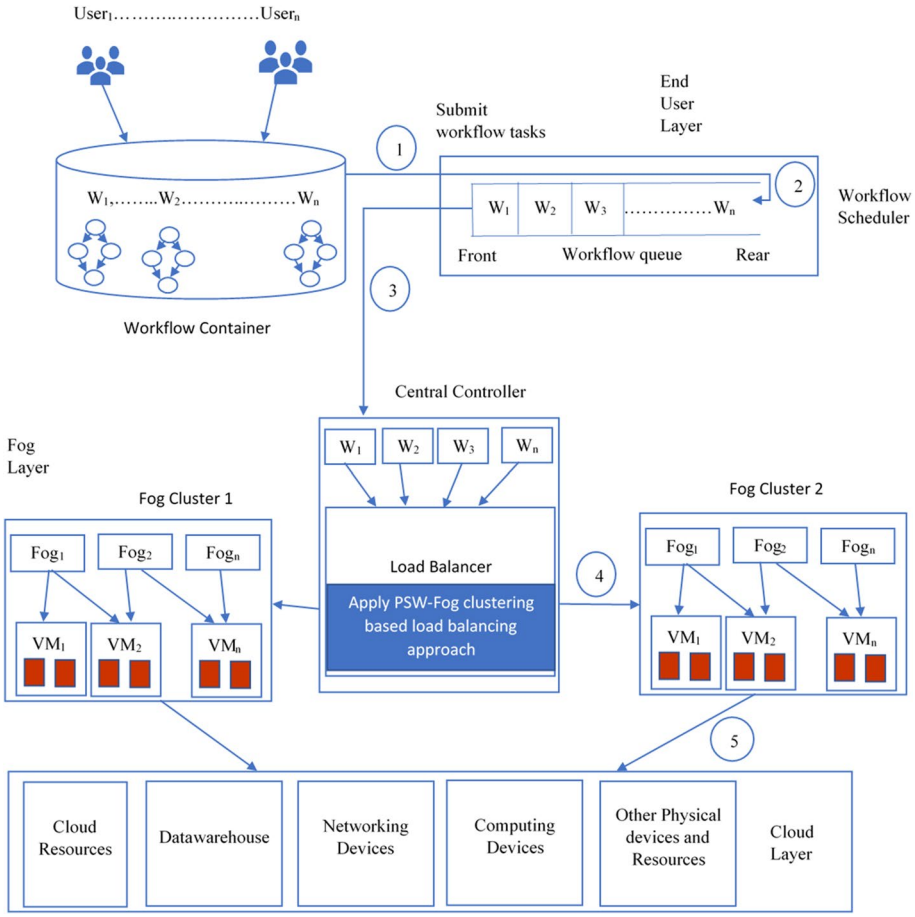


Fig. 2 Resource-utilization based workflow execution model for fog computing

4.1.1 Plant Growth Optimization(PGO)

This algorithm is proposed to simulate the real way of a plant's growth by considering its branching, leaves growth, and phototropism. PGO varies from the natural plant-growth process to consider two kinds of behaviours to find the optimal solution. Firstly it produces new branching points to find the optimal solution. Secondly, it considers the new growing leaves around branches to find accurate solution [6, 7]. We have considered the PGO approach to select cluster heads in different fog-nodes clusters in our hybrid approach. PGO is considered because of the need for less group variance because cluster size variance increases by default, causing load imbalance in nodes. The PGO is based upon a real tree's growth in which the trunk grows from roots and branches grow from the branches. This process goes on, and some new branches grow from the nodes of branches. The same process continues till the tree is formed. The plant growth process has been followed to develop an optimization approach in which optimization starts from the plant's root and keeps growing till branches until the best solution is found [1, 16]. The following equation can be used to find an optimum solution using PGO.

$$M_i = \begin{cases} 1 - \frac{f(x_i) - f_{\min}}{\sum_{j=1}^{N} [f(x_j) - f_{\min}]}, & f(x_i) > f_{\min} \\ 1, & f(x_i) = f_{\min} \end{cases} \quad i = 1, 2, 3 \ldots, N. \tag{1}$$

In Eq. (1) M_i optimize threshold at i[th]iteration its depend on fog nodes distance function $f(x_i)$ its Euclidian distance. With minimum value of threshold maximum chance on same cluster.

4.1.2 Simulated Annealing Algorithm(SAA)

The SAA was introduced by [23] in 1983 based on metal annealing's chemical process. The process of SAA starts with a random answer, and then the neighbour solution is found. SAA is generally used for single solution problems [14]. Many researchers have used SAA to solve different NP-hard problems and solved large computational tasks [2, 14, 29] In our proposed approach, we have divided different fog nodes into clusters. After selecting the cluster head using the PGO algorithm, workflow tasks are mapped onto clusters. So, we used SAA for intracluster mapping. SAA is used to analyze all cluster resources and their energy consumption. The mapping of tasks can be predicted using the current allocation of resources. SAA helps find a global solution for the NP-hard problem by having a large error margin. The margin error can be defined as the acceptance probability that can be considered to predict the task mapping on cluster resources. The following formula has been used to predict task mapping in clusters.

$$P = \frac{1}{\left(1 + e^{-\frac{\Delta(Cur_i)}{I}}\right)} \ldots \tag{2}$$

In Eq. (2) P is the mapping task prediction of task mapping on fog cluster that is calculated by $e^{-\frac{\Delta(Cur_i)}{I}}$ current allocation resources, resources utilization reduce then task mapping will increase because of prediction value increase.

4.1.3 Water Cycle Optimization(WCO)

The main idea behind WCO is taken by observation of the natural water cycle process that describes how all water from streams and rivers flow into the sea [13]. The water cycle describes how river and stream is formed by receiving water from melting of glaciers or heavy rains, and then this water goes into sea. This basic concept of natural water cycle has been used by different researchers to implement optimization to solve various computations [34]. When SAA could not find the optimized solution then WCO is used to maximize the energy and cost of resources between inter cluster resource mapping. Firstly, population in clusters is defined on which tasks to be mapped. Following equation describes the population of clusters.

$$C_i = f\left(x_1^i, x_2^i, x_3^i \ldots, x_n^i\right) \mathrm{i} = 1, 2, 3 \ldots \mathrm{N_{pop}} \tag{3}$$

In eq(3), C_i shows the population of clusters on which task is mapped. Here N is number of design variables. A matrix shows the initial population of size N_pop

Table 2 Notations

Notations	Description
Mi	Optimized threshold at ith iteration
$f(x_i)$	Distance function
P	Prediction of task mapping
e	Current resource allocation
C_i	Population of clusters
NS_n	Optimize cluster mapping by Ncluster
C_n	Fog nodes in cluster
C_i	Average value of fog nodes in cluster
F_n	Fog nodes
N_{clusters}	Random cluster of fog nodes
S_n	Simulated annealing
$N_p op$	N population
T_i	Execution time
P_i	Computation time
E_{En}^{Fog}	Energy consumption
C_{Cn}^{Fog}	Forwarding tasks
C_{Pc}^{Fog}	Computational Tasks
∂	Clustering threshold

$$NS_n = \text{round}\left\{ \left| \frac{C_n}{\sum_{i=1}^{n} C_i} \right|_{\text{cluster}} \quad n = 1, 2, 3 \ldots, N_{\text{cluster}} \right. \tag{4}$$

In Eq. (4), NS_n optimize the task mapping on fog clusters by WCA objective function which optimizes cluster mapping by number of cluster resources N_{cluster}. C_n is number of fog nodes in a cluster and $\sum_{i=1}^{n} C_i$ is average value of normalize cluster fog nodes.

4.2 Proposed Algorithm

This section proposes a PSW-fog clustering-based load balancing algorithm for scientific workflow applications. The heuristic optimization algorithms explained in Sect. 4.1 have been used in a hybrid form to create a PSW-fog clustering-based load balancing algorithm. These approaches are combined to find the best optimal solution without wasting execution time and energy of resources. In the algorithm, 1 number of fog nodes and cloud resources are considered input. The desired output of the algorithm is to optimize load balancing on fog nodes. Available fog nodes are combined in the form of fog clusters. The PGO algorithm is applied to find the cluster head for each cluster. Here population in each cluster is defined, and the fitness function takes two different parameters, i.e. energy, and cost. The convergence is applied according to both these parameters. If converged, then select cluster head. Now, after the selection of each cluster head, tasks are mapped onto resources by using SAA. If optimized, then computing parameters are analyzed. If not optimized, then task migration applied to balance the load between each resource and WCO using the same computing parameters. Now here population for each cluster is defined, and an optimized

solution for cluster mapping is found. If optimized, then computing parameters are analyzed; otherwise, again, WCO is applied to optimize. These approaches can not provide an optimal solution for load balancing in fog computing if applied alone. We have done hybridization of all the explained approaches to enhance resource utilization in a fog environment. When all the resources are utilized reasonably, it will reduce the time delay processing the tasks. Energy consumption in these nodes will automatically be reduced, which will reduce the computational cost.

Algorithm 1: PSW-Fog Clustering based load balancing algorithm

Input: Number of fog nodes and resources of cloud
Output: Optimize Load balancing on fog nodes

Fn← Fog nodes
Nclusters← Random Cluster of Fog nodes
PG = plant growth(Ncluster, Fn)
for *1 to Ncluster* **do**
Apply $M_i = \begin{cases} 1 - \frac{f(x_i) - f_{\min}}{\sum_{j=1}^{N} [f(x_j) - f_{\min}]}, f(x_i) > f_{\min} \\ 1 \qquad\qquad\qquad\quad f(x_i) = f_{\min} \end{cases}$ $i = 1, 2, 3 \ldots\ldots\ldots N$ *and Calculate* $M_i(ClusteringThreshold)$
if $Clusterth > M_i$ **then**
Begin
Cluster $\leftarrow F_n$
else
Clusteri+1$\leftarrow F_n$
EndIf
Stop
Sn =simulated annealing ($Cluster_i, F_n$)
for *1 to* $Cluster_i$ **do**
Start
Map task by $P = \frac{1}{\left(1 + e^{-\frac{\Delta(Cur_i)}{T}}\right)}$
Stop
if *(optimize)* **then**
Begin
Analysis of Task computing Parameters
else
Apply WCO (tasks, cluster$_i, F_n$)
For every cluster define population by $C_i = f\left(x_1^i, x_2^i, x_3^i \ldots\ldots\ldots\ldots, x_n^i\right)$ *i=1,2,3….*N_{pop}
Define Cost by $NS_n = round\left\{|\frac{C_n}{\sum_{i=1}^{n} C_i}|_{cluster} \quad n = 1, 2, 3 \ldots\ldots N_{cluster}\right.$
if $[NS]_n < min(NS_n)$ **then**
Begin
Computation of tasks on VM
Analysis of Task computing Parameters
else
Go to step 22
End
End

4.3 Flow of Execution of Proposed Algorithm

This section explains the proposed algorithm in the form of a flow chart that explains the proposed algorithm's working. The methodology used in this algorithm is divided into three parts, i.e. making fog nodes clusters, initial task mapping, and optimize mapping among clusters. All these steps are explained as follows:

1. *Making Fog nodes cluster* In this step, firstly, fog nodes are created into a group of fog nodes, and then a cluster head is selected. The plant growth (PGO) approach takes this decision. This approach uses two parameters for optimization, i.e. energy and

resource cost. If both parameters optimized, make a cluster and select cluster head. Plant growth optimization is used because of the need for less variance of the group because cluster size variance increases by default that causes load imbalance.

2. *Task initial mapping* After selecting optimize clusters and cluster head, parse the workflow and map task on fog groups. Here the different combination of fog nodes has been made on which different tasks are mapped. Here we initiate simulated annealing (SAA) according to fog clusters that take intra-cluster mapping decision. After simulated annealing analysis of all cluster node resources, map tasks on nodes and analyze performance parameters, i.e. energy, cost, and time delay. If performance parameters optimized, run the computation and again analyze parameters. Otherwise, go to the next step.
3. *Optimize mapping among clusters* This step comes into the picture when the previous step does not optimize. If the previous step fails to optimize all considered parameters, there is still load imbalance in the fog layer, hence needing load balancing. So task migration is applied to the clusters, and tasks from heavily loaded clusters are taken and transferred to lightly loaded clusters. In this step, we have to use water cycle optimization by considering all performance metrics. The WCO reduces the energy consumption and cost of resources between inter-cluster. If all parameters optimized, then we can analyze all the parameters to check the performance of the proposed approach; otherwise, the loop continues with task migration on clusters. Figure 3 represents the flow of execution in the PSW-Fog clustering-based load balancing algorithm. The Fig. 3 is divided into three steps explained above in this section.

4.4 Performance Metrics

The proposed algorithm tries to reduce the execution time of tasks in fog clusters to reduce time delay. Along with this computational cost, and energy consumption in fog nodes has been reduced. All the considered computational parameters have been described as follows.

4.4.1 Time Delay

Time delay in fog environment can be considered as the time interval between the submission of the task to the response after processing of that task. The time delay depends on computational time; if the computation time is less, then the time delay will be lower. The fog layer mainly tries to reduce the time delay for processing tasks near end users. The following equation describes the time delay calculated in the proposed algorithm.

$$T_i = \left\{ 1 - \frac{f(x_i) - f_{\min}}{\sum_{J=1}^{N} \left[f(x_j) - f_{\min} \right]} + \left\{ \alpha_{i(t)} * \left(P_i - \sum Q^i \right) * \beta_i^{cl} \right. \right. \tag{5}$$

In Eq. (5) First part of latency depend on objective function of plant growth optimization and second show the latency calculation by fog nodes and cloud respectively but fog nodes cluster and temporary storage improve the computation time P_i and $\sum Q_i$ show the resources and these are dependent on two hyper parameters β_i^{cl} and $\alpha_{i(t)}$.

4.4.2 Cost

The computational cost in a fog environment can be considered in the form of the maintenance cost and energy consumption in fog nodes [25]. If only a few resources are utilized, and others remain underutilized, they must be adequately maintained. Along with fog resources, cloud resources also compute tasks that also have some computational cost. The following equation is used to calculate the computational cost in a fog-cloud environment.

$$C_{Pc}^{cost} = \left\{ Cluster_i.NS_n(1-\beta) * \left(\alpha_{i(t)} - \alpha_{i(t-1)}\right) + C_{Cn}^{cloud} \right. \tag{6}$$

In Eq. (5) C_{Pc}^{cost} represents the analysis of computation task in Cloud. Computation depends on different parameters where $NS_n(1-\beta)\, N_V$ is virtual machine mapping with task migration of cloud fog nodes, and $\left(\alpha_{i(t)} - \alpha_{i(t-1)}\right)$ is number of computation resources available. If this quantity increases then computation cost also increases. C_{Cn}^{cloud} also depend on resources of cloud.

4.4.3 Energy

The fog layer contains various devices such as gateways, routers, servers that consume a large amount of energy while executing large computational tasks. While migrating tasks between clusters, energy consumption in nodes is also increased. Hence maximum load balancing in fog nodes can help to reduce energy consumption in the fog environment. The following equation represents the energy consumption in a fog environment.

$$E_{En}^{Fog} = \left\{ \partial * \sum_{i=1}^{N} C_{Cn}^{Fog} + (1-\partial) * \sum_{i=1}^{N} C_{Pc}^{Fog} + C_{Cn}^{cloud} \right. \tag{7}$$

Equation (7) shows the E_{En}^{Fog} Energy consumption which depends on two factors: first is C_{Cn}^{Fog} forwarding tasks, and second is C_{Pc}^{Fog} Computation tasks. Here ∂ is clustering threshold.

5 Experimental Results and Performance Analysis

This section evaluates the proposed PSW-Fog clustering approach by executing scientific workflow applications. For simulation purpose, iFogSim works in collaboration with CloudSim because of its huge library of resource management and cloud environment simulation. CloudSim handles all the events occuring between different fog components. In experimental requirement Table 3, it has been explained that we have considered 20 to 200 fog nodes that makes fog nodes clusters. Approximately 20 fog clusters has been considered that contains 10 to 20 VM per cluster. Fog nodes have less computing capacity, so fog layer is further connected to cloud layer having large datacenter. Fog layer works in collaboration with cloud layer. Fog layer processes the tasks having less time requirements, and other tasks with low priority are sent to cloud layer. We carried out experiment by considering benchmark workflows such as GENOME, Cybershake [15], SIPHT [12], and LIGO [12] that are available in "Pegasus" repository [https://pegasus.isi.edu/workflow_gallery/]. To evaulate our proposed PSW-Fog clustering approach we compared its results with other existing approaches i.e. Artificial Bee Colony (ABC), Tabu search [32, 38], Ant Colony

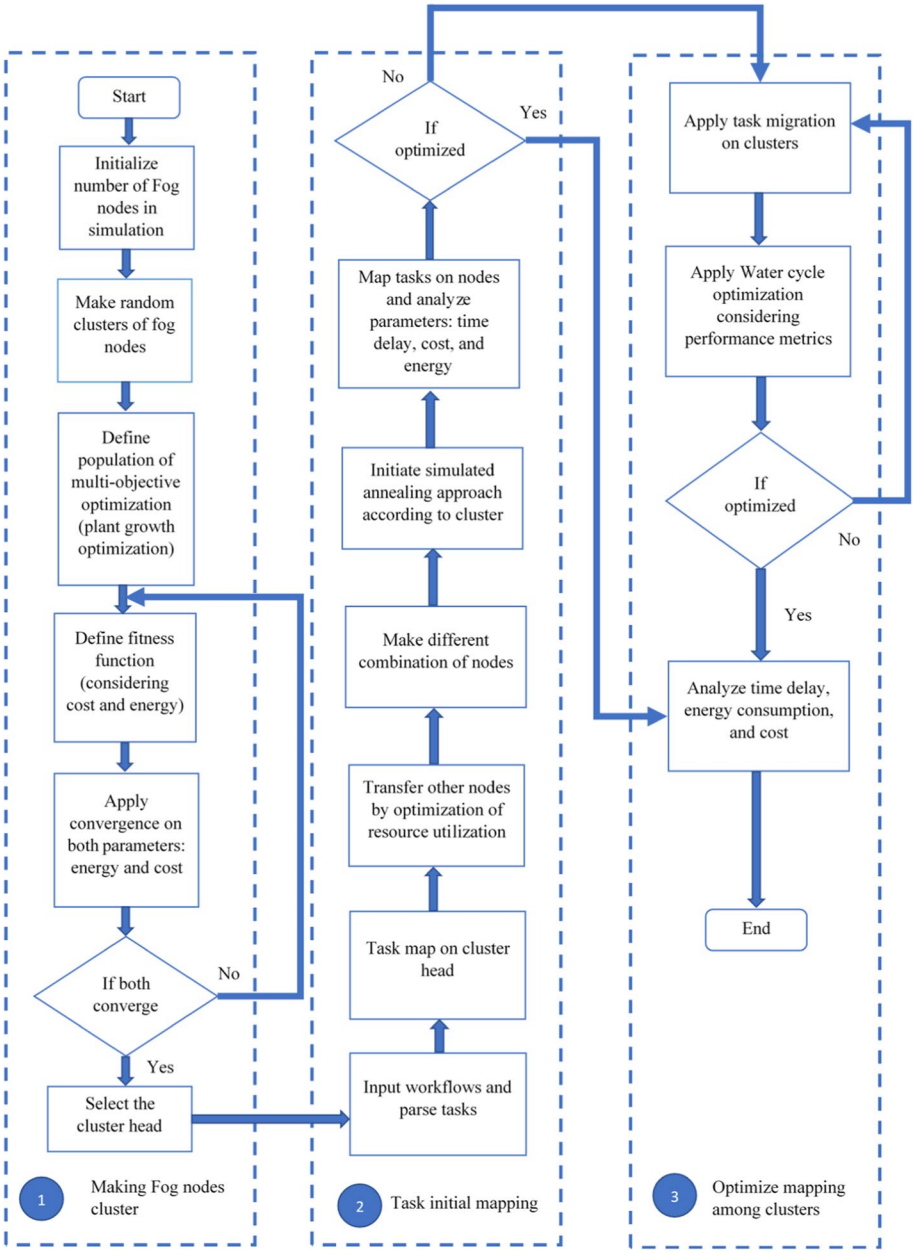


Fig. 3 Flow of execution of PSW-Fog clustering based load balancing algorithm

optimization(ACO) [17, 28], Grey Wolf Optimization(GWO) [31], and Tabu-GWO-ACO approaches and it has been represented through graphs that proposed approach outperforms than all other optimization approaches. PSW-Fog clustering approach tries to enhance performance parameters considered during experimentation.

5.1 Experimental Requirement

To obtain simulation results, we have considered few experimental requirements. Windows7 64bit operating system has been used. The fog layer has been divided into 20 fog clusters, each containing a maximum of 20 nodes. Table 3 represents the experimental requirements used for the evaluation of the proposed PSW-Fog clustering-based load balancing in fog environment.

iFogSim has been considered because of its open-source availability. iFogSim works with CloudSim in collaboration. Fog computing and IoT networks can be simulated using iFogSim. It is high-performance toolkit used for fog and edge computing environments. iFogSim contains three components, i.e. physical, logical, and management components. The physical components contains different physical fog nodes that can be deployed anywhere near to network edge. The logical component contains different application modules and application edges. The last component, i.e. management that includes fog controller as well as module mapping objects [27].

5.2 Result Analysis

This section covers the simulation results obtained from iFogSim after evaluating the proposed approach PSW-Fog clustering-based load balancing. These results have been compared based on time delay, cost, and energy consumption in a fog environment. We considered datasets of LIGO, Genome, SIPHT and cybershake. The obtained results are shown in graphs that compare the proposed approach with other existing approaches.

5.2.1 Cost Analysis

The considered workflows are executed using the proposed approach, and their computational cost has been analyzed. The workflows are taken in size of 20 to 100 tasks, and they are executed on fog nodes considering 20 to 200 fog nodes as represented in graphs in Fig. 4. The graphs represent variation in cost during execution on a varying number of fog nodes. It can be seen in graphs that with an increase in the number of fog nodes, computational cost also increased in every experiment. With our proposed multi-objective optimization approach, we have improved cost reduction compared to other existing heuristic approaches. Figure 4a shows the computational cost while evaluating LIGO workflow. It can be seen from the graph that the proposed approach significantly reduces the computational cost in fog nodes during execution on the different number of fog nodes from 20 to 200. Our proposed approach overlaps the cost values of other existing approaches in some experiments, but its average value improves the cost reduction. The improvement in cost reduction in the proposed approach is due to the clustering of fog nodes. Workflows are executed in distributed resources that reduce cost and other parameters. It has been obtained that after comparing with other approaches, PWS-fog clustering tries to reduce cost by 25% in the case of LIGO. In Fig. 4b, variation in cost can be seen during the evaluation of cybershake workflow, which clearly shows the reduction in cost by the proposed approach. The mean value of cost reduction in each experiment has been improved by PSW-Fog clustering. The reason behind this reduction in cost in the case of cybershake is its lesser complexity as compared to LIGO. In the experimental analysis, it has been obtained that PSW-Fog based approach tries to reduce 45% as compared to the average of all other considered approaches. Same like LIGO and cybershake, other workflows

Table 3 Experimental Requirement

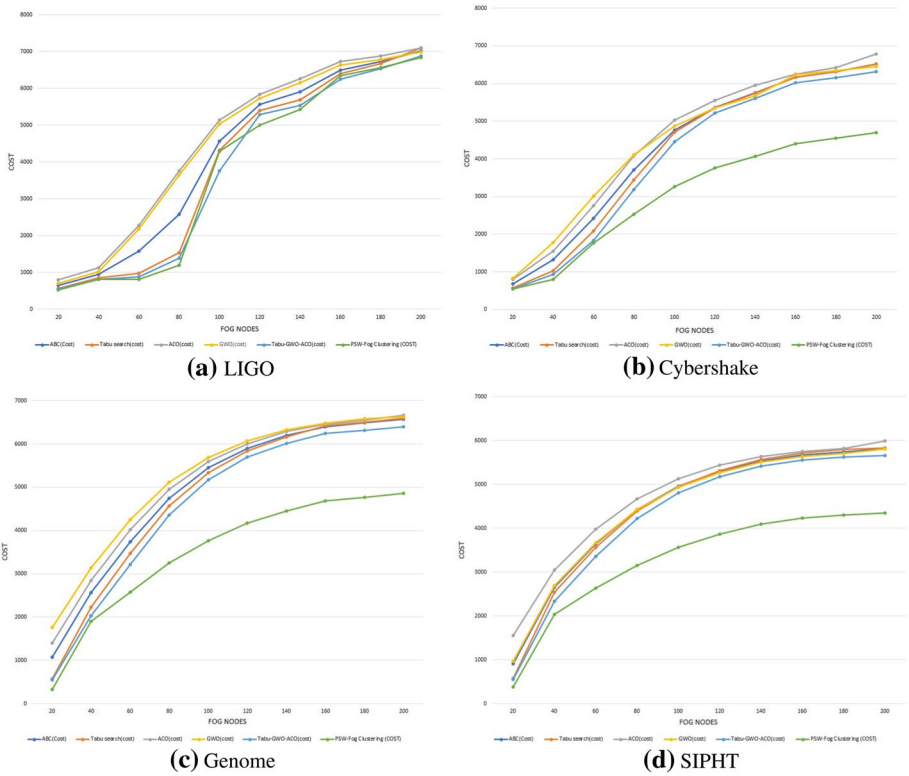
Experiments Parameters	Value/Name
Simulator	CloudSim and iFogSim
Dataset	Workflows (LIGO, SIPHT, GENOME, Cybershake)
Datacenter	One
VM	10 to 20
Fog Nodes	20 to 200
Optimization algorithm	simulated annealing, plant growth, WCA
Processer Per VM	0.5MIPS
Memory per VM	200 MB
Max Clusters	20
Max fog nodes in Cluster	20

GENOME and SIPHT are also executed. Figure 4c and d shows the improvement in cost reduction to 35%, 40% in the case of GENOME and SIPHT, respectively. Both the graphs show a significant reduction in computational cost in all experiments. This cost reduction in GENOME and SIPHT is due to optimization in the clustering of fog nodes and the migration of tasks between the optimized cluster.

5.2.2 Time Delay Analysis

Time delay has been evaluated and represented in the form of bar graphs. Figure 5 has been divided into two parts out of which Fig. 5a represents the time delay in evaluating LIGO workflows. Our proposed approach reduces time delay in executing workflow tasks on fog nodes compared to other existing approaches. As shown in the LIGO graph, when 20 fog nodes are considered, the time delay overlaps the other existing approaches. However, the average time delay has been reduced in other experiments by considering 40–200 fog nodes. Fog nodes have been grouped into clusters that help in reducing the execution time of tasks. Due to distributed nature of fog computing, all the resources are distributed in available fog nodes clusters that help in the execution of large workflow tasks, i.e. 20-100 tasks per cluster. Time delay is measured in seconds. The X-axis of graphs represents the number of fog nodes considered for experiments, and the time delay has been represented on the y-axis. Figure 5b shows time delay calculated while executing cybershake workflow that clearly represents that PSW-Fog clustering approach reduces time delay as compared to other considered approaches. Due to the low complexity of cybershake, it has been obtained that the mean value of time delay has been reduced in every experiment.

Similar improvement trends have been observed in the case of GENOME and SIPHT that are represented in Fig. 6a and b, respectively. Figures represent execution time of GENOME and SIPHT that shows significant improvement in all experiments. It has been obtained that while considering 20 to 200 fog nodes in experiments, time delay has been reduced to 30%, 50%, 40%, and 45% in case of LIGO, cybershake, Genome, and SIPHT, respectively.



(a) LIGO **(b)** Cybershake **(c)** Genome **(d)** SIPHT

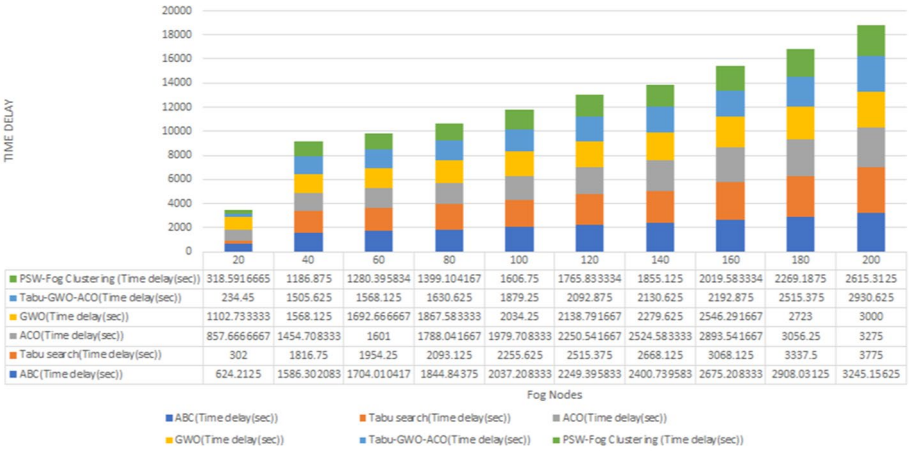
Fig. 4 Cost analysis of different workflows

5.2.3 Energy-Consumption Analysis

A similar kind of experiments has been performed to analyze energy consumption in fog nodes. We have considered the same number of fog nodes like cost and time analysis, i.e. 20 to 200, for conducting experiments. We have conducted approximately ten experiments considering 20,40,60,⋯,200 fog nodes in every experiment. Figure 7a shows the energy consumption in the case of LIGO. It can be seen from Fig. 7a that PSW-Fog based load balancing technique diminishes energy consumption to 60% compared to other existing approaches. Clustering of fog nodes reduces the intra-cluster task migration, reducing energy consumption during the migration of tasks between nodes. Figure 7b represents energy consumption in cybershake, which shows a 70% reduction in energy consumption with the proposed approach. Similarly Fig. 7c and d shows energy consumption in GENOME and SIPHT respectively. Both the Fig. 7c and d represents the reduction in energy consumption to 50%, and 45% in case of PSW-fog clustering-based load balancing approach in case of all the experiments. This improvement is due to optimization in the clustering of fog nodes.

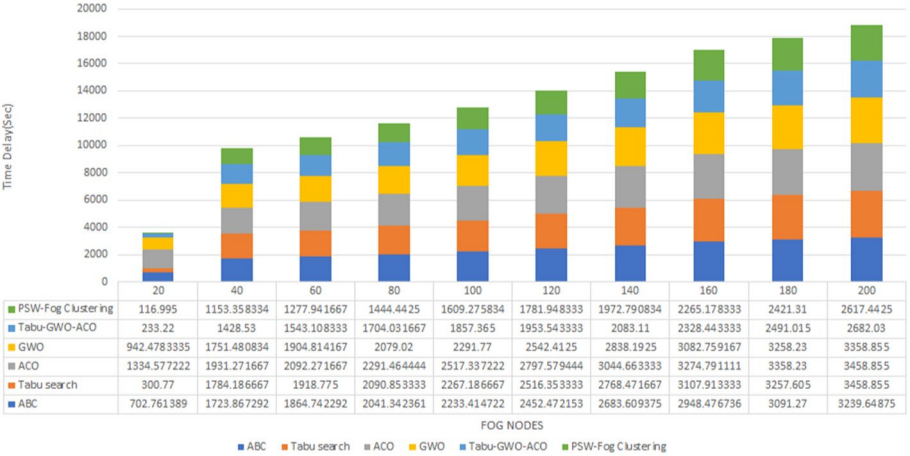
6 Conclusion and Future Directions

Load balancing in scientific workflows is necessary to utilize the resources at the fog layer entirely. This article come up with the architecture for fog computing, implementing load balancing in a scientific workflow. Furthermore, this article reviews the existing load balancing and scheduling techniques in workflows and briefly reviews them. Furthermore, the PSW-fog clustering-based load balancing algorithm has been proposed by amalgamating different existing heuristic approaches, i.e. plant growth, simulated annealing, and water cycle algorithms. Different types of existing scientific workflow examples have been described that are implemented in a fog environment using the proposed approach. We have considered three different computing parameters used to check the performance of the PSW-clustering, and these parameters are time delay, cost, and energy consumption. In order to evaluate the proposed approach and find simulation results, iFogSim has been used. The results obtained by executing PSW-fog clustering are collated with traditional



	20	40	60	80	100	120	140	160	180	200
PSW-Fog Clustering (Time delay(sec))	318.5916665	1186.875	1280.395834	1399.104167	1606.75	1765.833334	1855.125	2019.583334	2269.1875	2615.3125
Tabu-GWO-ACO(Time delay(sec))	234.45	1505.625	1568.125	1630.625	1879.25	2092.875	2130.625	2192.875	2515.375	2930.625
GWO(Time delay(sec))	1102.733333	1568.125	1692.666667	1867.583333	2034.25	2138.791667	2279.625	2546.291667	2723	3000
ACO(Time delay(sec))	857.6666667	1454.708333	1601	1788.041667	1979.708333	2250.541667	2524.583333	2893.541667	3056.25	3275
Tabu search(Time delay(sec))	302	1816.75	1954.25	2093.125	2255.625	2515.375	2668.125	3068.125	3337.5	3775
ABC(Time delay(sec))	624.2125	1586.302083	1704.010417	1844.84375	2037.208333	2249.395833	2400.739583	2675.208333	2908.03125	3245.15625

(a) LIGO



	20	40	60	80	100	120	140	160	180	200
PSW-Fog Clustering	116.995	1153.358334	1277.941667	1444.4425	1609.275834	1781.948333	1972.790834	2265.178333	2421.31	2617.4425
Tabu-GWO-ACO	233.22	1428.53	1543.108333	1704.031667	1857.365	1953.543333	2083.11	2328.443333	2491.015	2682.03
GWO	942.4783335	1751.480834	1904.814167	2079.02	2291.77	2542.4125	2838.1925	3082.759167	3258.23	3358.855
ACO	1334.577222	1931.271667	2092.271667	2291.464444	2517.337222	2797.579444	3044.663333	3274.791111	3358.23	3458.855
Tabu search	300.77	1784.186667	1918.775	2090.853333	2267.186667	2516.353333	2768.471667	3107.913333	3257.605	3458.855
ABC	702.761389	1723.867292	1864.742292	2041.342361	2233.414722	2452.472153	2683.609375	2948.476736	3091.27	3239.64875

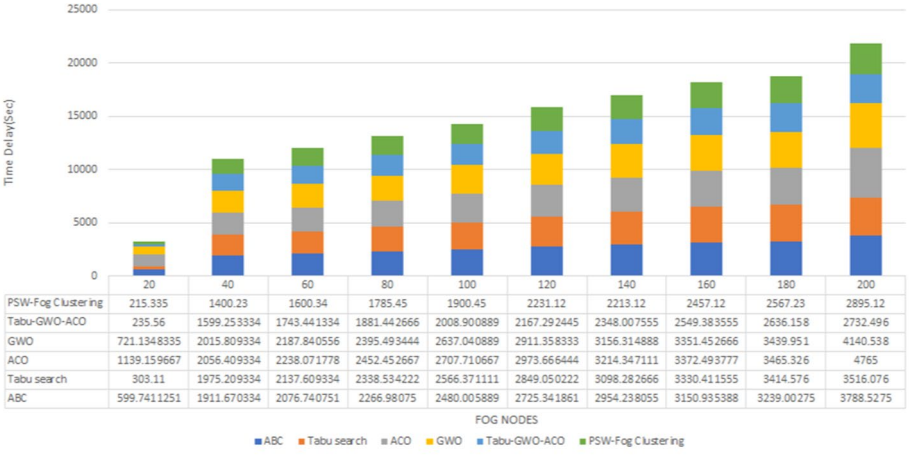
(b) Cybershake

Fig. 5 Time delay analysis of LIGO and cybershake workflows

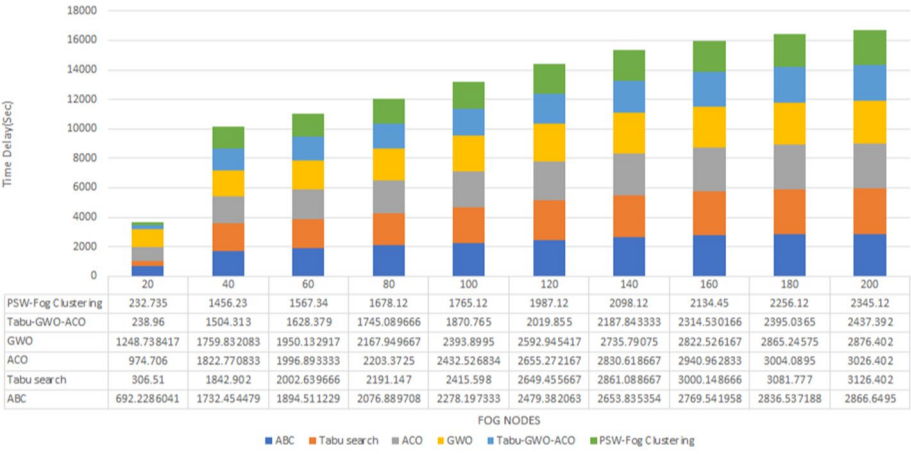
load balancing approaches. The graphs represent that our proposed solution outperforms other approaches. PSW-fog clustering reduces fog nodes' time delay, computational cost, and energy consumption.

There are some open issues in fog computing that can show future directions to the researchers to explore this area more. These issues are as below:

- Security of nodes in fog environment can be considered for future research.
- All existing load balancing approaches has been executed in the simulation environment, so there is a need for experimentation in the real world.
- There is a need to improve the multi-objective load scheduling problem in a fog-cloud environment.

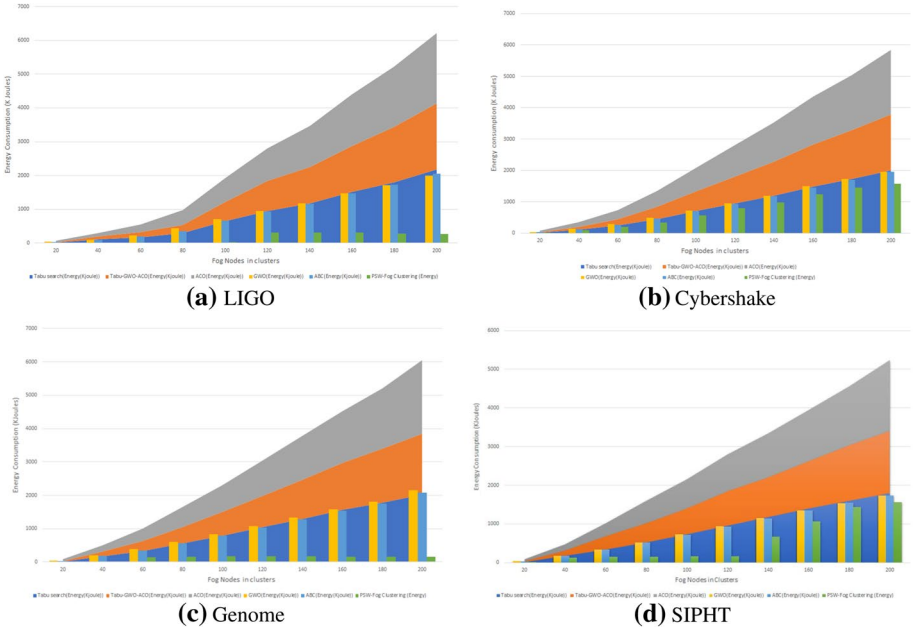


(a) Genome



(b) SIPHT

Fig. 6 Time delay analysis of genome and SIPHT workflows



(a) LIGO **(b)** Cybershake **(c)** Genome **(d)** SIPHT

Fig. 7 Energy analysis of different workflows

All these issues can be considered in future to explore the fog computing area. In the future, we will explore more load balancing approaches in fog computing and try to enhance the fog-cloud environment's performance.

Funding NA

Data Availability NA

Code availability NA

Declarations

Conflict of interest Authors does not have any conflicts of interests.

References

1. Akyol, S., & Alatas, B. (2017). Plant intelligence based metaheuristic optimization algorithms. *Artificial Intelligence Review, 47*(4), 417–462.
2. Bahadori-Chinibelagh, S., Fathollahi-Fard, A. M., & Hajiaghaei-Keshteli, M. (2019). Two constructive algorithms to address a multi-depot home healthcare routing problem. *IETE Journal of Research*. https://doi.org/10.1080/03772063.2019.1642802.
3. Beraldi, R., Canali, C., Lancellotti, R., & Mattia, G. P. (2020). Distributed load balancing for heterogeneous fog computing infrastructures in smart cities. *Pervasive and Mobile Computing*, p. 101221.

4. Berriman, G. B., Deelman, E., Good, J. C., Jacob, J. C., Katz, D. S., Kesselman, C., Laity, A. C., Prince, T. A., Singh, G., & Su, M. H. (2004). Montage: A grid-enabled engine for delivering custom science-grade mosaics on demand. In *Optimizing Scientific Return for Astronomy through Information Technologies* (Vol. 5493, pp. 221–232). International Society for Optics and Photonics.
5. Bonomi, F., Milito, R., Zhu, J., & Computing, S. A. Its role in the internet of things‖. In *Proceedings First Ed. MCC Workshop Mob. Cloud Comput.* (pp. 13–16). New York, NY: ACM.
6. Cai, W., Yang, W., & Chen, X. (2008). A global optimization algorithm based on plant growth theory: plant growth optimization. In *2008 International conference on intelligent computation technology and automation (ICICTA)* (Vol. 1, pp. 1194–1199). IEEE.
7. Cai, X., Li, P., & Wu, X. (2014). Artificial plant optimization algorithm with double selection strategies for dv-hop. *Sensor Letters, 12*(9), 1383–1387.
8. Choudhary, A., Govil, M. C., Singh, G., Awasthi, L. K., & Pilli, E. S. (2018). Task clustering-based energy-aware workflow scheduling in cloud environment. In *2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)* (pp. 968–973). IEEE.
9. De Falco, I., Laskowski, E., Olejnik, R., Scafuri, U., Tarantino, E., & Tudruj, M. (2015). Extremal optimization applied to load balancing in execution of distributed programs. *Applied Soft Computing, 30*, 501–513.
10. De Maio, V., & Kimovski, D. (2020). Multi-objective scheduling of extreme data scientific workflows in fog. *Future Generation Computer Systems, 106*, 171–184.
11. Ding, R., Li, X., Liu, X., & Xu, J. (2018). A cost-effective time-constrained multi-workflow scheduling strategy in fog computing. In *International Conference on Service-Oriented Computing* (pp. 194–207). New York: Springer.
12. Elsherbiny, S., Eldaydamony, E., Alrahmawy, M., & Reyad, A. E. (2018). An extended intelligent water drops algorithm for workflow scheduling in cloud computing environment. *Egyptian Informatics Journal, 19*(1), 33–55.
13. Eskandar, H., Sadollah, A., Bahreininejad, A., & Hamdi, M. (2012). Water cycle algorithm-a novel metaheuristic optimization method for solving constrained engineering optimization problems. *Computers & Structures, 110*, 151–166.
14. Fathollahi-Fard, A. M., Govindan, K., Hajiaghaei-Keshteli, M., & Ahmadi, A. (2019). A green home health care supply chain: New modified simulated annealing algorithms. *Journal of Cleaner Production, 240*, 118200.
15. Graves, R., Jordan, T. H., Callaghan, S., Deelman, E., Field, E., Juve, G., et al. (2011). Cybershake: A physics-based seismic hazard model for Southern California. *Pure and Applied Geophysics, 168*(3), 367–381.
16. Guney, K., Durmus, A., & Basbug, S. (2009). A plant growth simulation algorithm for pattern nulling of linear antenna arrays by amplitude control. *Progress In Electromagnetics Research, 17*, 69–84.
17. Hussein, M. K., & Mousa, M. H. (2020). Efficient task offloading for Iot-based applications in fog computing using ant colony optimization. *IEEE Access, 8,* 37191–37201.
18. Ijaz, S., Munir, E. U., Ahmad, S. G., Rafique, M. M., & Rana, O. F. (2021). Energy-Makespan optimization of workflow scheduling in fog-cloud computing. *Computing, 103,* 2033–2059.
19. Javadzadeh, G., & Rahmani, A. M. (2020). Fog computing applications in smart cities: A systematic survey. *Wireless Networks, 26*(2), 1433–1457.
20. Kaur, M., & Aron, R. (2020). Energy-aware load balancing in fog cloud computing. In: *Materials Today: Proceedings*.
21. Kaur, M., & Aron, R. (2020). Equal distribution based load balancing technique for fog-based cloud computing. In *International Conference on Artificial Intelligence: Advances and Applications, 2019* (pp. 189–198). New York: Springer.
22. Kaur, M., & Aron, R. (2021). Focalb: Fog computing architecture of load balancing for scientific workflow applications. *Journal of Grid Computing, 19*(4), 1–22.
23. Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science, 220*(4598), 671–680.
24. Li, C., Tang, J., Ma, T., Yang, X., & Luo, Y. (2020). Load balance based workflow job scheduling algorithm in distributed cloud. *Journal of Network and Computer Applications, 152*, 102518.
25. Li, C., Zhuang, H., Wang, Q., & Zhou, X. (2018). Sslb: Self-similarity-based load balancing for large-scale fog computing. *Arabian Journal for Science and Engineering, 43*(12), 7487–7498.

26. Liao, S., Wu, J., Mumtaz, S., Li, J., Morello, R., & Guizani, M. (2020). Cognitive balance for fog computing resource in internet of things: An edge learning approach. *IEEE Transactions on Mobile Computing, 21,* 1596–1608.
27. Mahmud, R., & Buyya, R. (2019). Modelling and simulation of fog and edge computing environments using ifogsim toolkit. In *Fog and edge computing: Principles and paradigms* (pp. 1–35).
28. Mirtaheri, S. L., & Shirzad, H. R. (2019). Optimized distributed resource management in fog computing by using ant-colony optimization. *Future Trends of HPC in a Disruptive Scenario, 34,* 206.
29. Mousavi, S. M., & Tavakkoli-Moghaddam, R. (2013). A hybrid simulated annealing algorithm for location and routing scheduling problems with cross-docking in the supply chain. *Journal of Manufacturing Systems, 32*(2), 335–347.
30. Naha, R. K., Garg, S., Battula, S. K., Amin, M. B., & Georgakopoulos, D. (2021). Multiple linear regression-based energy-aware resource allocation in the fog computing environment. arXiv:2103.06385.
31. Patel, D., Patra, M. K., & Sahoo, B. (2020). Gwo based task allocation for load balancing in containerized cloud. In *2020 International Conference on Inventive Computation Technologies (ICICT)* (pp. 655–659). IEEE.
32. Rehman, A., Hussain, S. S., ur Rehman, Z., Zia, S., & Shamshirband, S. (2019). Multi-objective approach of energy efficient workflow scheduling in cloud environments. *Concurrency and Computation: Practice and Experience, 31*(8), e4949.
33. Rizvi, N., & Ramesh, D. (2020). Fair budget constrained workflow scheduling approach for heterogeneous clouds. *Cluster Computing, 23*(4), 3185–3201.
34. Sadollah, A., Eskandar, H., Bahreininejad, A., & Kim, J. H. (2015). Water cycle algorithm with evaporation rate for solving constrained and unconstrained optimization problems. *Applied Soft Computing, 30*, 58–71.
35. Saroa, M. K., & Aron, R. (2018). Fog computing and its role in development of smart applications. In *2018 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Ubiquitous Computing & Communications, Big Data & Cloud Computing, Social Computing & Networking, Sustainable Computing & Communications (ISPA/IUCC/BDCloud/SocialCom/SustainCom)* (pp. 1120–1127). IEEE.
36. Serhani, M. A., El-Kassabi, H. T., Shuaib, K., Navaz, A. N., Benatallah, B., & Beheshti, A. (2020). Self-adapting cloud services orchestration for fulfilling intensive sensory data-driven Iot workflows. *Future Generation Computer Systems, 108,* 583–597.
37. Shahid, M. H., Hameed, A. R., ul Islam, S., Khattak, H. A., Din, I. U., & Rodrigues, J. J. (2020). Energy and delay efficient fog computing using caching mechanism. *Computer Communications, 154*, 534–541.
38. Siasi, N., Jaesim, A., & Ghani, N. (2019). Tabu search for efficient service function chain provisioning in fog networks. In *2019 IEEE 5th International Conference on Collaboration and Internet Computing (CIC)* (pp. 145–150). IEEE.
39. Singh, S. P., Sharma, A., & Kumar, R. (2020). Design and exploration of load balancers for fog computing using fuzzy logic. *Simulation Modelling Practice and Theory, 101*, 102017.
40. Téllez, N., Jimeno, M., Salazar, A., & Nino-Ruiz, E. (2018). A tabu search method for load balancing in fog computing. *International Journal of Artificial Intelligence, 16*(2), 78–105.
41. Wadhwa, H., & Aron, R. (2018). Fog computing with the integration of internet of things: architecture, applications and future directions. In *2018 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Ubiquitous Computing & Communications, Big Data & Cloud Computing, Social Computing & Networking, Sustainable Computing & Communications (ISPA/IUCC/BDCloud/SocialCom/SustainCom)* (pp. 987–994). IEEE.
42. Xie, Y., Zhu, Y., Wang, Y., Cheng, Y., Xu, R., Sani, A. S., Yuan, D., & Yang, Y. (2019). A novel directional and non-local-convergent particle swarm optimization based workflow scheduling in cloud-edge environment. *Future Generation Computer Systems, 97*, 361–378.

Mandeep Kaur is presently working in Chitkara University Institute of Engineering and Technology, Chitkara University, Rajpura, Punjab, India. She has completed her PhD (Part-Time) in 2021 from lovely professional university, Jalandhar. She has completed her M.Tech in CSE in 2015 from DIET, Kharar under PTU, Jalandhar. She has completed B.Tech in CSE in 2011 from G.G.S.C.M.T. Kharar under PTU, Jalandhar. Her area of interest is Fog computing and cloud computing.

Rajni Aron received her PhD in computer science from Thapar University, Patiala in 2013. She obtained her Master's degree in mathematics and computing from Thapar University, Patiala in 2009. At present she is working as Assistant Professor in SVKM's Narsee Monjee Institute of Management Studies (NMIMS) University,Mumbai (Maharashtra), India. Previously, she worked as Associate Professor in Computer Science & Engineering Department, Lovely Professional University, Punjab. She has also worked as Assistant Professor, Computer Science & Engineering Department, LNMIIT, Jaipur. Before joining LNMIIT, she was Post-Doctorate Fellow in INRIA, France and Concordia University, Montreal, Canada. She has published her research work in highly reputed scientific citation index journals. She won "Microsoft Azure" educator grand award and got many scholarships to attend international conferences like Grace hopper celebration of women in computing sponsored by Google.