International Journal of Networked and Distributed Computing
Vol. **9**(**1**); January (2021), *pp.* 59–74
DOI: https://doi.org/10.2991/ijndc.k.210111.001; ISSN 2211-7938; eISSN 2211-7946
https://www.atlantis-press.com/journals/ijndc

Research Article

Scheduling Algorithms in Fog Computing: A Survey

Khaled Matrouk*, Kholoud Alatoun

Department of Computer Engineering, Faculty of Engineering, Al-Hussein Bin Talal University, Ma'an, 71111, Jordan

ARTICLE INFO

Article History

Received 30 April 2020
Accepted 14 July 2020

Keywords

Fog computing
scheduling algorithms
task scheduling
job scheduling
workflow scheduling
resource allocation
IoT applications

ABSTRACT

Over the last recent years, applications related to the internet of things have become the most important techniques in the world that facilitate interactions among humans and things to enhance the quality of life. So, the number of devices used in these applications will increase, leading to the creation of huge amounts of data. Cisco proposed fog computing in 2012, which located between the end-users (Internet of Things devices) and cloud computing. Fog computing is not a replacement for cloud computing, but it reduces the drawbacks of cloud computing, makes it efficient and provides storage and computing services at the edge of the internet. Resource management is the key factor that decides the performance of fog computing. Whereas scheduling plays an important role in managing resources in fog computing, task scheduling is the ability to map tasks to the appropriate resources in fog computing. The task is a small part of a work that must be performed within a specific time. Because fog computing contains heterogeneous and distributed resources, task scheduling becomes complex. Task scheduling is an NP-hard problem that needs to apply effective task scheduling strategies to reach an ideal solution. There were many proposed algorithms about scheduling in the previous years; most of them were applied in cloud computing, while the minority were applied in fog computing. This paper aims to comprehensively review and analyze the most important up-to-date scheduling algorithms in fog computing.

1. INTRODUCTION

Nowadays, with the development of technology, the Internet of Things (IoT) applications have become parts of our daily life. Consequently, the number of devices used in these applications will increase, leading to the creation of huge amounts of data. This data will be transferred to cloud computing for processing, and because the cloud is far from these devices, there will be a delay in the response. From here, it was necessary to find a new technology closer to the Internet of Things devices and overcome the problems in the cloud. So, Cisco proposed fog computing in 2012 [1], which placed between cloud computing and the IoT devices (end users). Fog computing is an emerging computing paradigm that extends cloud computing from the core of the network to the edge of the network. It aims to bring computation, storage and networking services close to users [2].

Internet of Things applications is connecting every physical object like cameras, vehicles, sensors, wearables, and home appliances [3]. Many applications require low latency, mobility, location awareness, high response time. Although many researchers have developed algorithms to improve the performance of cloud computing about the IoT applications, there are still many challenges regarding the requirements of the IoT applications and mobile services such as low latency, high response time, cost, support for mobility and geo-distributions [4]. They can be addressed these challenges by fog computing [5]. Kazem [6] presents a comparison between fog and cloud computing according to their latency and response time. As a result of the comparison, fog computing always performs better than cloud computing to meet the demands of time-sensitive applications by reducing the delay and response time. Through the prediction of Cisco, there are more than 50 billion devices that will be connected to the internet by 2020. Also, the data produced by users, devices and their interactions will reach 500 zettabytes [7].

The general fog computing architecture can be divided into three layers, as shown in Figure 1. The first is IoT devices layer that includes different types of devices, such as smartphones, smart vehicles, tablet computers, and various smart home devices. This layer can sense the surrounding environment and collect data through sensor devices, and communicate with the fog computing layer through 3G, 4G, 5G, WiFi, and Bluetooth technologies. The second is the middle layer is the fog computing layer that includes routers, gateways, workstations, switches, access points. This layer has the capability of computing, networking, and storage. Finally, the upper layer is cloud computing that includes cloud servers with high computation power [2,8–10].

Changes like user devices in terms of bandwidth, storage, latency and computation make resource management in the fog computing environment a major issue [11].

Scheduling is the main challenge in fog computing. In a fog environment, tasks are divided into two groups: tasks requiring the computing intensity and tasks requiring the data intensity. While scheduling the tasks requiring computing intensity, the scheduler migrates the data to the high productivity resource, and hence, the task execution time is reduced. On the other hand, while scheduling

*Corresponding author. Email: khaled.matrouk@ahu.edu.jo

Figure 1 | Fog computing architecture.

the tasks requiring data intensity, it is attempted to reduce the number of data migration. As a result, the time of data transfer is reduced [12].

Nowadays, the number of IoT devices is increasing, which - in turn- increases the load on fog and cloud nodes for processing. Hence, it requires an efficient technique to schedule tasks and to manage resources of fog and cloud environments [11]. Many authors have improved heuristic algorithms to obtain a better scheduling performance [13].

There are some studies related to fog computing, and its architecture, its characteristics and the challenges it faces. Liu et al. [10] present a definition and an architecture of fog computing. Also, they discussed the framework of resource allocation for latency reduction combined with reliability, fault tolerance, privacy, and underlying optimization problems. Hu et al. [8] present an overview and summarize fog computing architecture, application, key technologies, challenges, and open issues. Mahmud et al. [14] analyze the challenges in the fog environment and present a taxonomy of fog computing according to the identified challenges and its key features. Atlam et al. [15] present a review of fog computing including a discussion of fog characteristics, architecture, and benefits. Also, it focuses on different IoT applications that will be improved through the fog. Kraemer et al. [16] provide a review on fog computing within healthcare informatics, and explore, classify, and discussed different application use cases presented in the literature. Also, they discuss where the fog computing tasks can be executed, on which level of the network. They provide tradeoffs concerning requirements relevant to healthcare. Hao et al. [2] present a description of fog computing and discussed its research challenges and problems. Also, they propose a flexible software architecture that can incorporate different design choices and user-specified policies. Moreover, they discuss the design of WM-FOG. Wadhwa and Aron [17] discuss the concept, architecture of fog computing and implemented application. They also highlight resource provisioning techniques to identify the overutilization of fog nodes. Yousefpour et al. [18] present a taxonomy on fog computing and its related computing paradigms, including their similarities and differences. Also, they provide a survey of research topics in fog computing and provide challenges and future directions for research in fog computing.

Despite the importance of scheduling approach in fog computing, there are no surveys on the scheduling algorithms in fog environment that help the researcher necessities on scheduling tasks and resource allocation field. Therefore, this paper aims to review and analyze the most important recent scheduling algorithms in fog computing comprehensively. After reading most of the current papers in the scheduling algorithms, only the most relevant scheduling algorithms papers have been taken into consideration. The scheduling problems have also been classified into five main categories: task scheduling, resource scheduling, resource allocation, job scheduling, and workflow scheduling.

The rest of this paper is organized as follows: Section 2 mentions related survey papers in the scheduling approaches in fog computing. Section 3 provides Scheduling objectives in fog computing. In Section 4, we mention the major five scheduling problems in fog computing and literature reviews of them. Section 5 shows a discussion and comparison of existing scheduling algorithms in fog computing. Finally, we conclude the survey in Section 6.

2. RELATED SURVEYS

This section illustrates some related review and survey papers in the scheduling approaches in the fog environment.

Ghobaei-Arani et al. [19] present a systematic literature review on the resource management techniques in fog environment. In this paper, they compare the resource management techniques with each other according to the important factors such as case study, performance metrics, utilized techniques, and evaluation tools as well as discussing their advantages and disadvantages.

Sharma and Rani [20] provide a review on fog computing including advantages, limitations, features, threats, and comparison of the different scheduling algorithms.

Rahbari and Nickray [21] present a review of the task scheduling and resource allocation in cloud, edge and fog computing. They also propose machine learning methods for intelligent task scheduling and offloading in distributed computing.

Naha et al. [9] provide a survey including an overview of fog computing and the differences between fog and cloud. Also, they investigate fog computing architectures and describe the components of these architectures in detail. Moreover, they mention some resource allocation and scheduling algorithms in the fog environment.

Hosseinioun et al. [22] present a survey to analyze the research works in task scheduling techniques in fog computing from 2015 to 2018. Also, they classify the task scheduling approaches in two fields: dynamic and static.

Mon et al. [23] provide a survey of different resource scheduling and load balancing algorithms in cloud and fog computing. They mention some limitations of scheduling and load balancing processes.

Briefly, the previous review and survey papers suffer from some weaknesses as follows:

- All papers do not contain the new current scheduling algorithms.
- Some papers focus on a part of scheduling approaches such as resource management [19], task scheduling [22] and resource scheduling [20,23].
- The papers do not study the topic of scheduling problems in fog computing.

The mentioned reasons above motivate us to prepare a survey paper on the scheduling algorithms in fog computing to overcome all these weaknesses.

3. SCHEDULING OBJECTIVES

Scheduling aims to find an optimal solution for scheduling a set of tasks or workflows on a set of machines. Scheduling parameters are effective in the success of the scheduling problem. Scheduling parameters are classified into two groups based on the service approach: consumer services and service providers [24].

3.1. Consumer Services

- Make-span is defined as the overall time required to execute and complete all tasks. In other words, it is the duration from which the user sends the job to the time he/she completes and gives results [25].
- The cost of an application includes computation cost, data transfer cost, and storage cost. When a resource is assigned to some tasks, data is transferred between the tasks and this transferring cost is calculated as Transfer cost = the size of the output data given by task/the average bandwidth between the Virtual Machines (VMs). When parent task and child tasks are scheduled on the same VM, the transfer cost becomes zero. Usually, the resources are charged per hour of computation and storage following the Amazon cloud services prices: (i) $0.1 per CPU instance hour for the computation resources. (ii) $0.15 per Gigabyte per month for the storage resource [24].
- Reliability is the probability of the run successfully and completes the execution of the job. Also, it means that all the resources work well during the execution of an application [24,26].
- Latency is the time duration to execute the whole task assigned to the fog node [3].
- Response time is the time taken by a user request until the arrival of the response at the requesting interface [3].

3.2. Service Provider

- Resource utilization: Increasing resource utilization is beneficial to the service provider to get maximum profit by renting the limited resources to the user in such a way that the resources are fully utilized [24].
- Energy consumption: It means the total energy consumed by the whole system. It is measured by any of the components of the system such as sensors, fog nodes, etc [3]. On the other hand, CPU utilization and resource utilization directly affect the energy consumed by a task. Energy consumption will be high when CPUs are not correctly utilized because idle power is not effectively used. Sometimes it gives high energy consumption due to the heavy demand of the resources, and this may lower down the performance [24].
- Stability is the probability that the resource node can complete the task smoothly [27].
- Allocated memory is the total amount of memory of a fog node, dedicated to the execution of a given task [4].
- The failure rate is the ratio of the number of tasks not completed within the deadline to the total number of scheduled tasks [28].

Table 1 shows the metrics considered for the current scheduling algorithms in the fog environment. From this table, we can decide the most important metrics used in the current algorithms. Considering all the metrics in a single scheduling algorithm is not a feasible solution, hence it increases the complexity of the algorithm.

4. SCHEDULING PROBLEMS

We consider five major scheduling problems in fog computing, which include task scheduling, resource scheduling, resource allocation, job scheduling, and workflow scheduling. Currently, there are a few existing works on the scheduling in fog computing. After reading in-depth the current research works in scheduling, we select the most relevant works, compare them, and divide them according to the type of scheduling problem as follows.

4.1. Resource Scheduling

Resource scheduling is aimed to find the best matching resources for the client to achieve the optimal scheduling goal such as improving resource utilization and reducing the processing delay and Quality of Service (QoS) [29].

Li et al. [29] propose an algorithm that combines the Fuzzy C-Means (FCM) clustering algorithm [30] with the Particle Swarm Optimization (PSO) [31], which is called Fuzzy Clustering Algorithm with the PSO (FCAP) algorithm. This algorithm proposes to cluster fog resources into three categories: computing, bandwidth, and storage resources. They also propose a Resource Scheduling Algorithm based on the FCAP algorithm (RSAF) to accomplish resource scheduling. Using MATLAB experiment results shows that the objective function of the FCAP algorithm has a faster convergence speed than the FCM algorithm [30]. Moreover, the RSAF algorithm can quickly match user requests with the appropriate resource categories and improve user satisfaction when compared to the Min–Min algorithm [32]. After all, this work does not consider the dynamic changes in resources and suffers from low resource utilization.

Sun et al. [27] propose a two-level resource scheduling technique in fog computing. The first level is the resource scheduling among fog cluster, which specifies which fog cluster will perform the task when it arrives. The second level is the resource scheduling among fog nodes in the same fog cluster, which specifies which fog resource will perform the task when it arrives. They also design a resource scheduling mechanism based on an improved Non-dominated Sorting Genetic Algorithm II (NSGA-II) [27] to improve the stability of the task execution and reduce service latency. The experimental results using MATLAB show that the proposed scheme gives better performance compared with a Fog-based IoT Resource Management Model (FIRMM) [33] and Random scheduling algorithms in terms of service latency and stability. However, the proposed work is not suitable for complex topology and suffers from the high cost of resource requesters.

Table 1 | Metrics considered of current scheduling algorithms in fog environment

Authors and references	Make span	Cost	Resource utilization	Reliability	Stability	Energy consumption	Allocated memory	Latency	Failure rate	Response time
Li et al. [29]	✓	×	×	✓	×	×	×	×	×	×
Xu et al. [28]	×	×	×	×	×	✓	×	×	✓	×
Wang and Li [13]	✓	×	×	✓	×	✓	×	×	×	×
Nguyen et al. [37]	✓	✓	×	×	×	×	×	×	×	×
Sun et al. [27]	×	×	×	×	✓	×	×	✓	×	×
Boveiri et al. [39]	×	×	✓	×	×	×	×	×	×	×
Yin et al. [45]	×	×	✓	×	×	×	×	✓	×	×
Pham et al. [46]	✓	✓	×	×	×	×	×	×	×	×
Hoang and Dang [48]	✓	×	✓	×	×	×	×	✓	×	×
Bitam et al. [4]	✓	×	×	×	×	×	✓	×	×	×
Stavrinides and Karatza [49]	✓	×	×	×	×	×	×	×	×	×
Ghobaei-Arani et al. [12]	✓	×	×	×	×	×	×	×	×	×
Ghaffari [51]	✓	×	×	×	×	✓	×	×	×	✓
Sharma and Saini [3]	×	×	×	×	×	✓	×	✓	×	✓
Rafique et al. [11]	✓	×	×	×	×	✓	×	×	×	✓
Wu and Lee [73]	×	×	×	×	×	✓	×	×	×	×
Li et al. [60]	×	×	×	×	×	✓	×	✓	×	×
Xu et al. [72]	✓	✓	×	×	×	×	×	×	×	×
Wu et al. [34]	✓	×	×	×	×	✓	×	×	×	×
Liu et al. [61]	✓	✓	×	×	×	×	×	×	×	×
Rahbari and Nickray [68]	×	✓	×	×	×	✓	×	×	×	×
Ghenai et al. [70]	×	×	×	×	×	×	×	✓	×	×
Pham et al. [62]	✓	✓	×	×	×	×	×	×	×	×
Agarwal et al. [67]	×	✓	✓	×	×	×	×	×	×	✓
Rahbari and Nickray [63]	×	✓	✓	×	×	✓	×	×	×	×
Aburukba et al. [66]	×	×	×	×	×	×	×	✓	×	×
Jamil et al. [79]	×	×	✓	×	×	×	×	✓	×	×
Kabirzadeh et al. [75]	✓	✓	✓	×	×	✓	×	×	×	×

Wu et al. [34] propose an energy-efficient evolutionary algorithm called Estimation of Distribution (EDA-P), to address the resource scheduling of three tiers of IoT systems. The EDA-P algorithm is used for producing task permutation. The experiments performed on both single and multi-application cases show that the proposed algorithm is effective in reducing make-span, the energy consumption of devices and prolonging the lifetime of IoT devices. On the other hand, the proposed algorithm does not address the application with soft or hard deadlines, and it does not consider the execution cost.

4.2. Task Scheduling

Task scheduling is aimed to assign a set of tasks to fog nodes to meet the satisfaction of QoS requirements in such a way the execution and transmission time of tasks is minimized [12].

Xu et al. [28] propose an associated task scheduling strategy based on Laxity-Based Priority and Ant Colony System (LBP-ACS) in the cloud-fog environment. The proposed strategy is divided into two parts: the LBP Algorithm (LBPA) and Constrained Optimization Algorithm based on the ACS (COA-ACS). The first part (LBPA) is used to obtain the task priority order, and the second part (COA-ACS) is used to obtain the task scheduling scheme. By using CloudSim simulation and Compared with Greedy for Energy (GfE), Heterogeneous Earliest Finish Time (HEFT) [35], and Hybrid ant colony optimization with differential evolution algorithms [36], the experimental results show that the proposed algorithm can reduce the energy consumption of processing all tasks and reduce the failure rate of associated tasks scheduling with mixed deadlines. Anyway, the authors consider the scheduling of tasks that include only associated tasks with no independent tasks. They do not also propose putting the task to a nearby edge server which would be more benefit those delay-sensitive tasks.

Wang and Li [13] propose a task scheduling strategy based on a Hybrid Heuristic (HH) algorithm for different fog nodes to solve the problem of terminal devices with limited computing resources and high energy consumption. The HH algorithm combines the Improved Particle Swarm Optimization (IPSO) algorithm and the Improved Ant Colony Optimization (IACO) algorithm. The authors evaluate this work by using MATLAB. The experiment results show that the HH algorithm gives the best performance compared with IPSO, IACO, and Round Robin (RR) with three performance metrics (Make-span, energy consumption, and reliability). After all, this algorithm does not apply to task clustering and fog node clustering.

Nguyen et al. [37] propose an algorithm to optimize task scheduling problems for Bag-of-Tasks applications in the cloud-fog environment called Time-Cost aware Scheduling (TCaS) algorithm. By using iFogSim simulation, the results show that the TCaS algorithm gives an improvement of 15.11%, 11.04%, and 44.17% compared with Bee Life Algorithm (BLA) [4], Modified PSO (MPSO) [38], and RR algorithms, respectively, while achieving a balance between make-span and operating costs. However, the proposed algorithm does not concern transmission cost, computing resources, and energy consumption.

Boveiri et al. [39] propose a high-performance way based on Max–Min Ant System (MMAS) to address the static task graph scheduling inhomogeneous multiprocessor environments. Because the MMAS is very flexible which allows it to be able to exploit the full potentials of an increase in the number of processors, it gives the best performance compared with Highest Level First with Estimated Times (HLFET) [40], Insertion Scheduling Heuristic (ISH) [41], Modified Critical Path (MCP) [42], Earliest Time First (ETF) [43], and Dynamic Level Scheduling (DLS) [44] algorithms. Consequently, the proposed approach does not consider the heterogeneous multiprocessor system, many-core system, grid and cluster computing. Moreover, it does not address energy consumption.

Yin et al. [45] propose a task scheduling model by considering the role of containers. Also, they designed a task scheduling algorithm to guarantee that the number of concurrent tasks for the fog node is enhanced and tasks are finished on time. Besides, by the characteristics of the containers, they proposed a reallocation technique to reduce task delay. The task scheduler distributes the task when its request is accepted. if the task is only completed in either the cloud or the fog node, it is directly distributed. Provided that the cloud and the fog node both complete the task, the task scheduler will need to decide where to place it. The task that has low computing is executed in fog node, but the task with high computing is forwarded to the cloud. The proposed algorithm and reallocation mechanism reduce task delay and improve the resource utilization of fog nodes. Anyhow, the authors ignore the computation time in the cloud, but in a real situation, this factor should be taken into account. Additionally, the image placement of containers is a significant problem, and this must be solved to reduce task execution time.

Pham et al. [46] propose a Cost-Makespan aware Scheduling (CMaS) heuristic algorithm for achieving the balance between the performance of application execution and the mandatory cost for the use of cloud resources. Also, they propose an efficient task reassignment mechanism based on the critical path of the directed acyclic graph to satisfy the user-defined deadline constraints of the system. The experiment results, using CloudSim simulation, show that the proposed work achieves a better balance between cost and make-span compared with HEFT [35], Greedy for Cost (GfC), and Cost-Conscious Scheduling Heuristic (CCSH) [47] algorithms. However, this study does not address energy consumption, and it is not suitable for large scale application, also it takes high scheduling time for task execution.

Hoang and Dang [48] designed a fog-based region architecture to provide close computing resources for latency-sensitive applications. Furthermore, they investigated an efficient scheduling algorithm to distribute tasks among regions and remote clouds, which called the Fog-Based Region and Cloud (FBRC) algorithm. The proposed work gives an efficient result of minimizing latency, reducing make-span, and increasing resource utilization compared with region-based and cloud-based resource management. On the other hand, the proposed work suffers from high time complexity, high processing time, low efficiency in service processing rate, and it does not consider the energy consumption.

Stavrinides and Karatza [49] propose a hybrid fog and cloud-aware heuristic [Hybrid-Earliest Dead-line First (Hybrid-EDF)] for dynamic scheduling of real-time IoT workflows in a three-tiered architecture. This method is attempted to schedule communication-intensive tasks with low computational demands in the fog, and computationally demanding tasks with low communication requirements in the cloud. The proposed scheduling technique consists of two phases: the task selection phase and the Virtual Machine (VM) selection phase. Tasks are given a priority based on the earliest deadline first. If the two tasks or more have the same priority, the task with the highest average computational cost is selected first. The task is selected by the scheduler, it is allocated to the VM that can provide it with the earliest estimated finished time. All VMs in the cloud and fog layers are considered. This hybrid approach providing on average 76.69% lower deadline miss ratio compared with Fog-EDF. However, this comes at a significant monetary cost, due to the usage of cloud resources, and it does not address energy consumption.

Ghobaei-Arani et al. [12] propose a Task Scheduling algorithm based on a Moth-Flame Optimization (TS-MFO) algorithm to meet the satisfaction of QoS requirements of the Cyber-Physical System (CPS) applications in fog environment. TS-MFO algorithm was spread like a moth to a flame. The proposed algorithm contains two matrices: The first matrix contains a set of fog nodes selected for the first task. And each row represents a solution. The best solution found for each individual of a population is stored in the second matrix as a flame. Both matrices include solutions for finding positions on fog nodes. The proposed algorithm is compared to PSO [50], NSGA-II [27], and BLA [4], algorithms by using iFogSim. The result shows that the proposed algorithm can minimize the total execution time of tasks. It is found that this work does not address energy consumption and communication cost.

Ghaffari [51] propose a scheduling method using an ant colony algorithm in fog computing. The proposed method is done in three steps as follows: Firstly, the tasks are divided into two groups based on end time and cost. Secondly, the tasks are prioritized, the tasks with the lowest end time have to be scheduled first and the tasks with the lowest cost have to be executed first. Finally, the ant colony algorithm is used to select an optimal virtual machine for executing the tasks. Using iFogSim, the simulation results show that the proposed method gives an acceptable performance in (the make-span, response time, and energy consumption) when compared to Ant Colony Based Meta-Heuristic approach for load balancing in cloud computing [52]. It is stated this method does not concern with transmission cost, computing resources and it only tests on small datasets.

Sharma and Saini [3] propose a four-tier architecture for delay aware scheduling and load balancing in the fog environment. The proposed algorithm is simulated by iFogSim and gives better results in terms of response time, scheduling time, load balancing rate, latency, and energy consumption compared with nine approaches CMaS [46], Delay Energy Balanced Task Scheduling (DEBTS) [53], BLA [4], NSGA-II [27], HEFT [35], Multi-Population Genetic Algorithm (MPGA) [54], Graph partitioning [55], Simple scheduling [56], and Dynamic Resource Allocation Method (DRAM) [57]. However, the proposed algorithm does not address the execution cost and the authors do not address the data replication techniques for managing data in a fog computing network since it can further reduce delay and overall dependency.

Rafique et al. [11] propose a Novel Bio-Inspired Hybrid Algorithm (NBIHA) which is a hybrid of MPSO and Modified Cat Swarm Optimization (MCSO) [58] for task scheduling and efficient resources management. MPSO is used to schedule the tasks

through the fog devices, and the NBIHA algorithm is used to manage resources efficiently. In the proposed NBIHA algorithm, the scheduler finds the best match of fog devices for an incoming task based on its requested for memory and CPU time. If it does not find any match from fog devices, it sends the task to the cloud. The result shows that the proposed algorithm is better compared to three algorithms [First Come First Serve (FCFS) [59], Shortest Job First (SJF), and MPSO] by minimizing the execution time, energy consumption, and average response time. However, it does not address the communication cost and it is intended to utilize the reinforcement learning techniques for managing resources in the fog-IoT environment.

Li et al. [60] designed a cloud-fog cooperation scheduling algorithm to reduce energy consumption when considering new tasks generated by IoT applications. Also, they proposed a task offloading algorithm to complete tasks when their nodes leave. They proposed a task execution flag. When the flag value is 1, the task is executed on the mobile device layer. If the flag value is 2, the task is executed on the fog node layer. Moreover, if the flag value is 3, this indicates that the task is executed on the cloud server layer. If the task is not allocated, the value of the flag is 0. The IoT network is dynamic, and one node could leave the system when it runs out of power. If there is an unfinished task in the node's queue, the task will be offloaded to other nodes for execution. The experiment results show that energy consumption is reduced by approximately 22%, while the delay is 12.5% less than the FCFS algorithm. This work only optimizes the energy consumption and delay in cloud fog computing with homogeneous fog nodes.

Liu et al. [61] present an Adaptive Double Fitness Genetic Task Scheduling (ADGTS) algorithm to optimize the make-span of the tasks and communication cost of the fog resources for smart cities. The simulation results show that the ADGTS algorithm can balance the performance of communication cost and task make-span at the same time, and it has better performance than the Min–Min algorithm [32] in task make-span. After all, this work does not address energy consumption.

Pham et al. [62] propose a heuristic-based Algorithm to achieve the balance between the make-span and the monetary cost of cloud resources. It determines the task priority and chooses a suitable node to execute each task. The results show that the proposed algorithm achieves a better cost-makespan tradeoff value than GfC, HEFT [35], and DLS [44] algorithms. On the other hand, the scheduling algorithm should be made more strong by considering additional constraints, such as deadline constraints of workflow execution and Fog provider's budget. It suffers from high workload execution time and low scalability as well.

Rahbari and Nickray [63] propose a Scheduling Algorithm [Knapsack Symbiotic Organisms Search (KnapSOS)] using SOS [64] based on the knapsack algorithm [65] to reduce the delay and energy consumption in the fog networks. KnapSOS algorithm was simulated by two-tracker case studies based on camera sensors with actuators. The simulation results show that the improvement in energy consumption is 18%, total network usage is 1.17%, execution cost is 15% and the lifetime of sensors is 5% compared with the original FCFS and knapsack algorithms. It is shown that the proposed algorithm suffers from high execution time.

Aburukba et al. [66] introduce a customized implementation of the Genetic Algorithm (GA) as a heuristic approach to scheduling the IoT requests to achieve the objective of minimizing the overall latency. The performance of the genetic algorithm is evaluated and compared to the performance of Waited Fair Queuing (WFQ), Priority-Strict Queuing (PSQ), and RR techniques. The simulation results show that the overall latency for the proposed approach is 21.9–46.6% better than the other algorithms. Also, the proposed approach shows significant improvement in meeting the requests deadlines by up to 31%. Still, this work can extend the proposed model to include critical request scheduling and to allow preemption. Moreover, multiple objective functions can be included to maximize resource utilization and minimize latency.

4.3. Resource Allocation

Resource allocation is the systematic approach of allocating available resources to the clients need over the internet [67].

Rahbari and Nickray [68] propose the Greedy Knapsack-based Scheduling (GKS) algorithm for allocating resources appropriately to modules in fog computing. The proposed method was simulated in iFogSim. The simulation results show that execution cost, energy consumption, and sensor lifetime in GKS are better than those of the Concurrent, FCFS, and Delay-Priority Algorithms [69]. However, the proposed algorithm ignores network usage that is an important parameter.

Ghenai et al. [70] propose a resource management approach based on the dynamic scheduling of multi-user devices using classification methods for cloud servers and heterogeneous devices. This approach is applied to manage hospitals that use cloud servers to save and treat patient and employee data to improve the employee's satisfaction and patient care quality over time. The simulation results show that the proposed approach guarantees a decrease in the network usage and hence substantiates its efficiency in streamlining patient information flow and its accessibility for health care providers. Nonetheless, this work does not consider the monetary cost of cloud resources and energy consumption, also does not apply in large scale applications.

Agarwal et al. [67] propose an efficient architecture and algorithm for resource provisioning in fog computing by using the virtualization technique. The Efficient Resource Allocation (ERA) algorithm is proposed which based on the fog layer. As the request will be performed by the client, this request will be accepted by the fog layer and not interrupt the clouds. If the request is not processed within its specified time, it will be sent to clouds by the fog layer. So the proposed algorithm is efficient to allocate the resources, maximize the throughput, and minimize the response time. The simulation results show that the proposed strategy can be allocated resources in an optimized way and better than Reconfigure Dynamically with Load Balancing (RDLB) and Optimize Response Time (ORT) [71] algorithms in terms of overall response time, bandwidth utilization, and the cost of data transfer in fog computing. However, the proposed algorithm is not satisfying the requirement of resources during the execution of the request. They have only allocated those resources to clients who are requesting before processing. Furthermore, the proposed algorithm designs for reserved instances only, and it suffers from low availability and high complexity.

4.4. Workflow Scheduling

Workflow scheduling aims to assign computing resources with different processing abilities to the tasks of workflow application, which can minimize the make-span and cost [72].

Wu and Lee [73] propose an Energy Minimization Scheduling (EMS) algorithm to minimize the energy consumption for IoT workflow on heterogeneous fog computing architectures. They consider only two types of edge nodes, fast nodes with more computationally powerful but consume more power and slow nodes have a lower power consumption. The experiment results show that the EMS algorithm can give the best energy consumption in comparison with Longest Time First (LTF) method, Integer Linear Programming (ILP) [74] and Random Algorithm. Although they do not consider the periodic tasks, they should be finished within their period. In addition to this, they only consider two types of fog nodes. The task migration between the edge nodes is not taken into consideration as well.

Xu et al. [72] propose a workflow scheduling algorithm based on IPSO in the cloud-fog environment. In this algorithm, the novel update method of inertia weight is designed as a nonlinear decreasing function, which facilitates to balance and adjust the global and local abilities of particles. According to the number of workflow tasks, each particle in PSO is encoded as a scheduling plan. The result shows that the proposed algorithm reduces the completion time of workflow compared to the original PSO [31] while the economic costs of the two algorithms are close with each other. However, they do not address energy consumption.

Kabirzadeh et al. [75] propose a Hyper-Heuristic algorithm to find better solutions for the workflow scheduling problem in the fog computing environment. The results show that the proposed algorithm improved the average energy consumption of 69.99% and cost 59.62% relative to the PSO [31], ACO [76], Simulated Annealing Algorithm (SA) [77], and GA [78] algorithms. This method reduces the simulation time and energy consumption by the size of a heuristic algorithm and increases the decision-making power for assigning resources with specific constraints to users according to the type of workflows. Eventually, this algorithm has low scalability. Also, the authors' algorithm does not address the virtual machine migration, resource provisioning and scheduling of applications on the IoT based on fog networks.

4.5. Job Scheduling

Job scheduling aims to assign a group of jobs to the lowest number of fog resources (e.g. less memory) in the shortest CPU execution time [4].

Bitam et al. [4] propose a BLA to optimize the distribution of a set of tasks among all the fog computing nodes. The proposed algorithm gives better performance in the execution time and the allocated memory compared to PSO and GA algorithms. Despite this, the proposed algorithm suffers from Low scalability. Besides, it does not address the dynamic job scheduling. Furthermore, the authors of BLA have tested the algorithm only on small datasets, and the response time is high for task execution.

Jamil et al. [79] propose an optimized job scheduling algorithm called SJF to minimize the delays for latency-critical applications. The simulation results show that the delay and network usage of the proposed algorithm improves by 32% and 16% respectively, compared to the FCFS algorithm. Although the proposed algorithm reduces the average waiting time. It can starve tuples with larger lengths.

Table 2 summarizes the main idea of the scheduling algorithms, advantages, limitations, and the simulation tool used in them. This table helps researchers to compare algorithms in terms of the method used to solve a certain type of scheduling problem, what are the advantages of using this proposed algorithm and what are the challenges it faces. Thus, researches can choose and develop the algorithms closest to their studies.

A wide range of proposed scheduling algorithms has been compared in Table 3 in terms of the following issues: the problem statement which is categorized by (task/job/workflow scheduling, resource scheduling, and resource allocation), metrics used in the algorithm, environment (fog or both fog-cloud), a case study in terms of the application they use, and finally, the algorithms that are compared to the proposed algorithm.

5. DISCUSSION AND COMPARISON

This section shows a discussion and comparison of current scheduling algorithms in fog computing. Figure 2 shows a statistical comparison of the scheduling problems. We consider five major scheduling problems that include task scheduling, resource scheduling, resource allocation, job scheduling, and workflow scheduling. The task scheduling problem has the highest percentage of the scheduling algorithms by 57% usage in the literature. Both resource scheduling and resource allocation have the same percentage by 13%, workflow scheduling problem has 10%, and job scheduling has 7% usage in fog computing.

The applied case studies of scheduling algorithms in fog computing are shown in Figure 3. These case studies include many applications such as smart production lines, homogeneous multiprocessors, large scale applications, and so forth. We notice that most studies have a general case study.

According to Figure 4, 36% of the research papers have implemented the proposed approach using the iFogSim simulation tool. Besides, 18% of the research papers used the MATLAB simulator. Also, 11% of the studies have presented an evaluation of their case study using the CloudSim tool. Other studies used different simulation tools as shown in Figure 4. Moreover, some research studies have not specified a measurement environment and simulation tool for evaluating their methods. Figure 5 shows an analytical report of the performance metrics for evaluating scheduling algorithms. It is noticed that the make-span has the most usage in scheduling algorithms by 25%, followed by the energy consumption which has 19%, the cost is next which has 15%. Then, the latency has 13%, and resource utilization has 12%. The response time has 7%, and the reliability has 3%. Finally, stability, allocated memory, and the failure rate has the same percentage of 2%.

To conclude, fog computing has a major role in implementing IoT scheduling algorithms. Based on this survey, the task scheduling problem in fog computing is a research hotspot, and more research needs to be carried out on it. As a future scope new scheduling strategies to optimize multiple metrics in a way to address and solve other scheduling problems can be found. Make-span and energy

Table 2 | A comparison of the current scheduling algorithms in fog environment

Authors and references	Main ideas	Simulation	Advantages	Limitations
Li et al. [29]	Fuzzy Clustering Algorithm with the Particle Swarm Optimization (FCAP) algorithm is used to cluster fog resources.	MATLAB	RSAF algorithm can match user requests with the appropriate resource categories faster and improve user satisfaction.	Not consider the dynamic changes in resources.
	Resource Scheduling Algorithm based on FCAP (RSAF) algorithm is used to accomplish resource scheduling.		FCAP has faster convergence speed than the Fuzzy C-Means (FCM) algorithm.	Low resources utilization.
Xu et al. [28]	Associated task scheduling strategy based on Laxity-based Priority and Ant Colony System (LBP-ACS) algorithm in the cloud-fog environment.	CloudSim	Reduces the energy consumption of processing all tasks.	Consider the scheduling of tasks that include only associated tasks with no independent tasks.
			Reduce the failure rate of associated tasks scheduling with mixed deadlines.	They do not propose to put the task to a nearby edge server which would be more benefit those delay-sensitive tasks.
Wang and Li [13]	A task scheduling algorithm called Hybrid Heuristic (HH) algorithm in fog computing is proposed to solve the problem of terminal devices with limited computing resources and high energy consumption.	MATLAB	HH algorithm gives the best performance comparing with three algorithms Improved Particle Swarm Optimization (IPSO), Improved Ant Colony Optimization (IACO), and RR with three performance metrics (completion time, power consumption, and reliability).	Not apply on task clustering and fog nod clustering.
Nguyen et al. [37]	Time-Cost aware Scheduling (TCaS) algorithm based on the Genetic Algorithm is proposed to optimize task scheduling problems for Bag-of-Tasks applications in the cloud-fog environment in terms of execution time and operating costs.	iFogSim	TCaS algorithm gives an improvement of 15.11% compared with BLA, 11.04% compared to MPSO, and 44.17% compared to RR.	Not concerned with transmission cost, computing resources, and energy consumption.
	Each chromosome represents a job assignment to a node. Mutation and crossover were used to generate a new population.		This algorithm achieves a balance between completing time and operating costs.	
Sun et al. [27]	Multi-objective optimization of resource scheduling in fog computing using an improved Non-dominated Sorting Genetic Algorithm II (NSGA-II).	MATLAB	Reduce the service latency.	Not suitable for complex topology.
	Two-level resource scheduling model in fog computing. The first level is the resource scheduling among fog cluster, which specifies which fog cluster will perform the task when it arrives. The second level is the resource scheduling among fog nodes in the same fog cluster, which specifies which fog resource will perform the task when it arrives.		Improve the stability of the task execution.	Latency is high due to simple architecture. High cost of resource requesters.
Boveiri et al. [39]	This approach is based on the Max–Min Ant System (MMAS) to treat the static task graph scheduling in homogeneous multiprocessor environments.	Pentium IV using Visual basic 6.0	The Max–Min Ant System (MMAS) is very flexible which allows it to be able to exploit the full potentials of an increase in the number of processors, it gives the best performance compared with Highest Level First with Estimated Times (HLFET), Insertion Scheduling Heuristic (ISH), Modified Critical Path (MCP), Earliest Time First (ETF), Dynamic Level Scheduling (DLS) algorithms.	Not consider the heterogeneous multiprocessor system, many-core system, and grid and cluster computing. Not address energy consumption.

Table 2 | A comparison of the current scheduling algorithms in fog environment—*Continued*

Authors and references	Main ideas	Simulation	Advantages	Limitations
Yin et al. [45]	Task scheduling model by considering the role of containers. Construct a task scheduling algorithm to ensure that tasks are completed on time and the number of concurrent tasks for the fog node is optimized. Reallocation mechanism to reduce task delay by the characteristics of the containers.	Linpack with java	Reduce task delay. Improve the resource utilization of fog nodes.	Ignored the computation time in the cloud. The image placement of containers is a significant problem, this problem must be solved to reduce task execution time.
Pham et al. [46]	Cost-Makespan aware Scheduling (CMaS) for achieving the balance between the performance of application execution and the mandatory cost for the use of cloud resources. Efficient task reassignment strategy based on the critical path of the directed acyclic graph modeling the applications to satisfy the user-defined deadline constraints of the system.	CloudSim	More cost-effective and achieves better performance compared with other existing methods.	Not address power consumption. Not suitable for large scale application. It takes high scheduling time for task execution. It is static. Considers a single workflow for scheduling.
Hoang and Dang. [48]	Fog-based Region and Cloud (FBRC) architecture to provide nearby computing resources. Efficient scheduling algorithm (FBRC algorithm) to distribute tasks among regions and remote clouds.	Unknown simulators	Gives an efficiency result of latency response and resource utilization compared with region-based and cloud-based resource management.	High time complexity. High processing time. Low efficiency in service processing rate. Not consider energy consumption.
Bitam et al. [4]	Bee Life Algorithm (BLA) is proposed to optimize the distribution of a set of tasks among all the fog computing nodes.	C++	Give better performance in the execution time and the allocated memory compared with PSO and GA algorithms.	Not address the dynamic job scheduling approach. Response time is high for task execution. Only tested on small datasets. Low scalability.
Stavrinides and Karatza [49]	Hybrid approach for dynamic scheduling of real-time Internet of Things (IoT) workflows in a three-tiered architecture in fog and cloud environments. This approach attempted to schedule computationally demanding tasks with low communication requirements in the cloud and communication-intensive tasks with low computational demands in the fog.	C++	Hybrid approach providing on average 76.69% lower deadline miss ratio.	Increase monetary cost, due to the usage of cloud resources. Not address energy consumption.
Ghobaei-Arani et al. [12]	Task scheduling algorithm based on a Moth-flame Optimization (MFO) algorithm is proposed to minimize the total execution time of tasks to meet the satisfaction of Quality of Service (QoS) requirements of Cyber-Physical System (CPS) applications.	iFogSim	Better compared with three algorithms (PSO, NSGA-II, BLA) by minimizing the total execution time of tasks.	This work does not address energy consumption and communication cost.
Ghaffari [51]	The proposed algorithm using the ant colony algorithm is done in three steps as follows: The first step: the tasks are divided into two groups based on end time and cost. Second step: the tasks are prioritized. The tasks with the lowest end time have to be scheduled first and the tasks with the lowest cost have to be executed first. Third step: the ant colony algorithm is used to select an optimal virtual machine for executing the tasks.	iFogSim	Give acceptable performance in (the end time, delay, energy consumption) when compared with (Ant Colony Based Meta-Heuristic approach for load balancing in cloud computing).	Not concerned with transmission cost, computing resources. Only tested on small datasets.

(*Continued*)

Table 2 | A comparison of the current scheduling algorithms in fog environment—*Continued*

Authors and references	Main ideas	Simulation	Advantages	Limitations
Sharma and Saini [3]	Four tires architecture for delay aware scheduling and load balancing in the fog environment.	iFogSim	It gives better results in terms of response time, scheduling time, load balancing rate, delay, and energy consumption compared with nine approaches. (CMaS, DEBTS, BLA, NSGA-II, etc.)	The proposed algorithm does not address the execution cost. The authors did not address the data replication techniques for managing data in a fog computing network since it can further reduce delay and overall dependency.
Rafique et al. [11]	NBIHA algorithm which is a hybrid of MPSO and MCSO for task scheduling and efficient resources management. MPSO is used to schedule the tasks among the fog devices. NBIHA is used to manage resources efficiently.	iFogSim	Proposed work is better compared with three algorithms [First Come First Serve (FCFS), Shortest Job First (SJF), Modified PSO (MPSO)] through minimizing the execution time, energy consumption, and average response time.	Not address the communication cost. It is intended to utilize the reinforcement learning techniques for managing resources in the fog-IoT environment.
Wu and Lee [73]	Energy Minimization Scheduling (EMS) algorithm is used to minimize the energy consumption for IoT workflow on heterogeneous fog computing architectures. They consider only two types of edge nodes, fast nodes with more computationally powerful but consume more power and slow nodes have a lower power consumption.	SPEC CPU2006	EMS algorithm can give the best energy consumption compared with Longest Time First (LTF), Integer Linear Programming (ILP), Random algorithms.	Not consider the periodic tasks that should be finished within their period. Only consider two types of fog nodes, where more types could be considered. Not consider the task migration between the edge nodes.
Li et al. [60]	Cloud-fog cooperation scheduling algorithm to reduce energy consumption when considering new tasks generated by IoT applications. Designed a task offloading algorithm to complete tasks when their nodes leave.	MATLAB	Energy consumption is reduced by approximately 22%, while the delay is 12.5% less than the FCFS algorithm.	Only optimize the energy consumption and delay in cloud fog computing with homogeneous fog nodes.
Xu et al. [72]	Workflow scheduling algorithm based on IPSO in the cloud-fog environment. In this algorithm, the novel update method of inertia weight is designed as a nonlinear decreasing function, which facilitates to balance and adjust the global and local abilities of particles. Each particle in PSO is encoded as a scheduling plan according to the number of workflow tasks.	MATLAB	Reduces the completion time of workflow compared with the original PSO while the economic costs of the two algorithms are close with each other.	Not address energy consumption.
Wu et al. [34]	Energy-efficient evolutionary algorithm called Estimation of Distribution (EDA) is proposed to address the resource scheduling of three tiers of IoT systems. The EDA algorithm is used for producing task permutation.	C++	Proposed algorithm is effective in reducing makespan and the energy consumption of devices, as well as prolonging the lifetime of IoT devices.	Not address the application with soft or hard deadlines. Not concerned with execution cost.
Liu et al. [61]	Adaptive Double Fitness Genetic Task Scheduling (ADGTS) algorithm is proposed to optimize the makespan of the tasks and communication cost of the fog resources for smart cities.	Unknown simulators	The Adaptive Double Fitness Genetic Task Scheduling (ADGTS) algorithm has better performance than the traditional Min-Min algorithm in task makespan and can balance the performance of task makespan and communication cost at the same time.	Not consider energy consumption

Table 2 | A comparison of the current scheduling algorithms in fog environment—*Continued*

Authors and references	Main ideas	Simulation	Advantages	Limitations
Rahbari and Nickray [68]	Greedy Knapsack-based Scheduling (GKS) algorithm for allocating resources appropriately is proposed to modules in fog computing.	iFogSim	Energy consumption, execution cost, and sensor lifetime in GKS are better than those of the FCFS, concurrent, and delay-priority algorithms.	Ignored network usage that is an important parameter.
Ghenai et al. [70]	Resource management approach based on the dynamic scheduling of multi-user devices using classification methods for cloud servers and heterogeneous devices to minimize latency for IoT applications. This approach was applied to manage a hospital that uses cloud servers to save and treat patient and employee data to improve the employee's satisfaction and the patient care quality over time.	iFogSim	The proposed approach guarantees a decrease in the network usage and hence substantiates its efficiency in streamlining the patient information flow and its accessibility for health care providers.	Not consider the monetary cost of cloud resources and power consumption. Not apply in large scale applications.
Pham et al. [62]	Heuristic-based algorithm to achieve the balance between the make-span and the monetary cost of cloud resources. Determine the task priority and choose a suitable node to execute each task.	CloudSim	The proposed algorithm achieves better cost make-span trade-off value than Greedy, HEFT, and DLS algorithms.	High workload execution time. Low scalability. Not consider the important parameters like deadline constraints of workflow execution and Fog provider's budget.
Agarwal et al. [67]	Efficient Resource Allocation (ERA) for resource provisioning in fog computing environments by using the virtualization technique.	Cloud Analyst (part of CloudSim)	Proposed strategy can be allocated resources in an optimized way and better than existing algorithms [Reconfigure Dynamically with Load Balancing (RDLB) and Optimize Response Time (ORT)] in terms of overall response time, data transfer cost and bandwidth utilization in the fog computing environment.	Not satisfying the requirement of resources during the execution of the request. They have only allocated those resources to the clients who are requesting before processing. Applicable for only reserved instances. Low availability. High complexity.
Rahbari and Nickray et al. [63]	Novel scheduling algorithm using symbiotic organisms search based on the knapsack algorithm to reduce the delay and energy consumption in the fog networks.	iFogSim	The improvements in the energy consumption by 18%, total network usage by 1.17%, execution cost by 15% and the lifetime of sensors by 5%. The proposed algorithm is better than the original FCFS and knapsack algorithms.	High execution time.
Aburukba et al. [66]	A genetic algorithm as a heuristic approach is proposed to scheduling the IoT requests to achieve the objective of minimizing the overall latency.	Discrete event simulator	The overall latency for the proposed approach is 21.9% to 46.6% better than the Waited Fair Queuing (WFQ), Priority-Strict Queuing (PSQ), and RR techniques. The proposed approach showed significant improvement in meeting the requests deadlines by up to 31%.	This work can extend the model to include critical request scheduling and to allow preemption. Multiple objective functions can be included to maximize resource utilization and minimize latency.
Jamil et al. [79]	Optimized job scheduling algorithm (SJF) is proposed to minimize the delays for latency-critical applications.	iFogSim	The delay and network usage of the proposed algorithm improves by 32% and 16% respectively, compared with the FCFS algorithm.	The proposed algorithm can starve tuples with larger lengths.
Kabirzadeh et al. [75]	A hyper-heuristic algorithm is proposed to find better solutions for the workflow scheduling problem in the fog computing environment.	iFogSim	The proposed algorithm improved the average energy consumption of 69.99% and cost 59.62% relative to the PSO, ACO, SA algorithms.	Low scalability. The authors' algorithm did not address the virtual machine migration, resource provisioning and scheduling of applications on the IoT based on fog networks.

Table 3 | A detailed comparison of the current scheduling algorithms in fog computing

References and year	Algorithm	Problem	Metrics	Environment	Case study	Results compared
[29], 2019	FCAP RSAF	Resource scheduling	User satisfaction Make-span Reliability	Fog	General	FCM and MIN-MIN
[28], 2019	LBP-ACS	Task scheduling	Energy consumption Failure rate	Cloud-fog	General	GfE, HEFT, and DEACO
[13], 2019	HH (Hybrid heuristic)	Task scheduling	Make-span Energy consumption Reliability	Fog	Smart production line	IPSO, IACO, and RR
[37], 2019	TCaS	Task scheduling	Make-span Operating costs	Cloud-fog	Bag of tasks applications	BLA, MPSO, and RR
[27], 2018	NSGA-II	Resource scheduling	Latency Stability	Fog	General	FIRMM and Random scheduling algorithm
[39], 2018	MMAS	Task scheduling	Resource utilization	Fog	Homogeneous multiprocessor	HLFET, ISH, MCP, ETF, and DLS
[45], 2018		Task scheduling Resource allocation	Latency Resource utilization	Fog	Containers for smart manufacturing	FT-FQ, FT-RE, DT-FQ, and DT-RE
[46], 2017	CMaS	Task scheduling	Mandatory cost for using the cloud resource Makespan	Cloud-fog	Large scale applications	GfC, HEFT, and CCSH
[48], 2017	FBRC	Task scheduling	Latency Makespan Resource utilization	Cloud-fog	Latency sensitive applications	Region-based and cloud-based resource management
[4], 2018	BLA	Job scheduling	Makespan allocated memory	Fog	General	PSO and GA
[49], 2018	Hybrid-EDF	Task scheduling	Makespan Deadline miss ratio	Cloud-fog	General	Fog-EDF
[12], 2019	TS-MFO	Task scheduling	Makespan	Fog	CPS applications	PSO, NSGA-II, and BLA
[51], 2019	Using-ACO	Task scheduling	Makespan Response time Energy consumption	Fog	General	Gill
[3], 2019	EDF	Task scheduling	Response time Scheduling time Load balancing rate Latency Energy consumption	Fog	Video Surveillance / Object Tracking (VSOT) application	CMaS, DEBTS, BLA, NSGA-II, HEFT, MPGA, Graph partitioning, Simple scheduling, and DRAM
[11], 2019	NBIHA	Task scheduling Resource scheduling	Make-span Energy consumption Response time	Fog	General	FCFS, SJF, MPSO
[73], 2018	EMS	Workflow scheduling	Energy consumption	Fog	General	LFT, ILP, and Random algorithms
[60], 2019	STML	Task scheduling	Energy consumption latency	Cloud-fog	General	FCFS
[72], 2018	Workflow scheduling algorithm based on IPSO	Workflow scheduling	Make-span Economic costs	Cloud-fog	Workflow applications	PSO
[34], 2018	EDA-P	Resource scheduling	Make-span Energy consumption	Fog	General	sEDA
[61], 2017	ADGTS	Task scheduling	Make-span Communication cost	Fog	Smart cities	Min-Min
[68], 2019	GKS	Resource allocation	Energy consumption Execution cost Sensor lifetime	Fog	Sensor application	FCFS, concurrent, and delay-priority algorithms
[70], 2018	Based on the dynamic scheduling of multi-user requests	Resource allocation	Latency	Fog	Manage a hospital that uses cloud servers to save and treat patient and employee data to improve the employee's satisfaction and patient care quality over time	Cloud-based approach

Table 3 | A detailed comparison of the current scheduling algorithms in fog computing—*Continued*

References and year	Algorithm	Problem	Metrics	Environment	Case study	Results compared
[62], 2016	Heuristic-based algorithm	Task scheduling	Make-span Monetary cost of cloud server	Cloud-fog	General	Greedy, HEFT, and DLS
[67], 2016	ERA	Resource allocation	Response time Data transfer cost Bandwidth utilization	Fog	General	RDLB and ORT
[63], 2017	KnapSOS	Task scheduling	Energy consumption Network usage Execution cost Lifetime of sensors	Fog	camera sensor with actuator	FCFS and knapsack algorithms
[66], 2019	GA	Task scheduling	Latency	Cloud-fog	General	WFQ, PSQ, and RR
[79], 2019	SJF	Job scheduling	Latency Network usage	Fog	Latency-critical applications	FCFS
[75], 2017	HH (Hyper-heuristic)	Workflow scheduling	Energy consumption Cost Make-span Network usage	Fog	General	PSO, ACO, SA, and GA

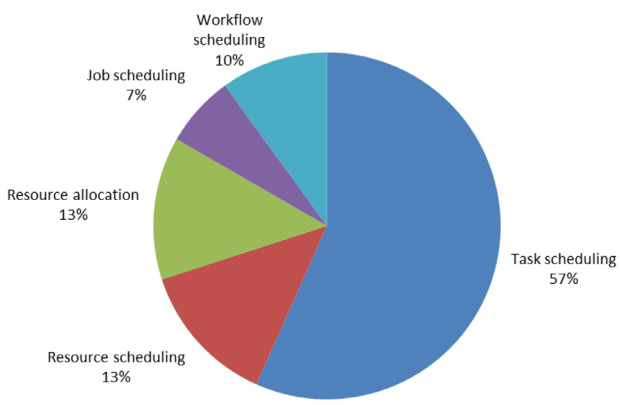

Figure 2 | Percentage of the scheduling problems solved by scheduling algorithms in fog computing.

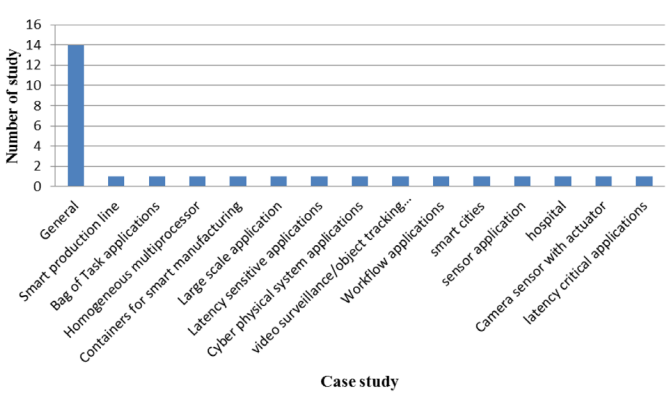

Figure 3 | Percentage of the applied case studies of scheduling algorithms in fog computing.

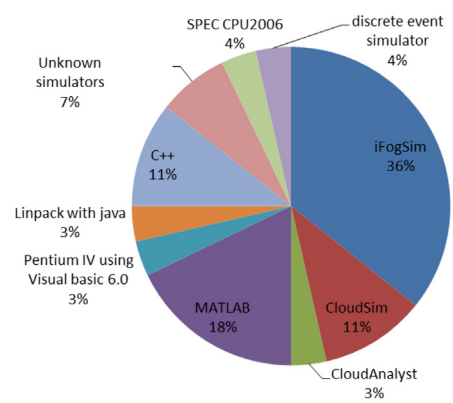

Figure 4 | Percentage of the presented evaluation tools in the literature.

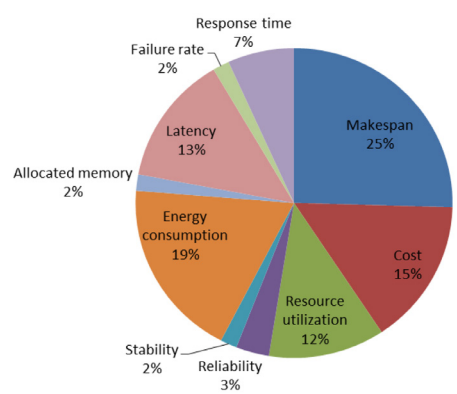

Figure 5 | Percentage of performance metrics for evaluating scheduling algorithms.

consumption are very important metrics in scheduling algorithms of fog computing. Fog computing uses in a very sensitive latency IoT applications, so the make-span is a very important metric. Energy wastage and potential battery drain in fog computing are new challenges. The energy of each fog node can be saved by using an efficient method in scheduling data. There are some other metrics and applications the researches need to take into consideration to make their new proposed scheduling algorithm good enough such as dynamic task scheduling, periodic tasks, task migration between the edge nodes, heterogeneous fog nodes, applications with soft or hard deadlines, virtual machine's migration, and energy consumption

in fog nodes. It is thought that the iFogSim tool is good to use because it is an open-source simulator and can use more than three metrics to evaluate the proposed algorithm.

6. CONCLUSION

This paper provides a comprehensive review and analysis of the most important recent scheduling algorithms in fog computing. After we read and analyze most of the current papers in the scheduling algorithms, the top scheduling algorithms have been chosen. This survey classifies the scheduling problems into five main categories: task scheduling, resource scheduling, resource allocation, job scheduling, and workflow scheduling. From the compared result, task scheduling has the highest percentage of the scheduling algorithms by 57% usage in the literature. And 36% of the research papers have implemented the proposed approach using the iFogSim tool. The makespan has the most usage in scheduling algorithms by 25%. Hence, most scheduling problems in fog computing are considered a research hotspot and more research need to be carried out on it. Moreover, make-span and energy consumption are very important metrics in the scheduling algorithms of fog computing.

CONFLICTS OF INTEREST

The authors declare they have no conflicts of interest.

REFERENCES

[1] F. Bonomi, R. Milito, J. Zhu, S. Addepalli, Fog computing and its role in the internet of things, Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing (MCC), 2012, pp. 13–16.

[2] Z. Hao, E. Novak, S. Yi, Q. Li, Challenges and software architecture for fog computing, IEEE Internet Comput. 21 (2017), 44–53.

[3] S. Sharma, H. Saini, A novel four-tier architecture for delay aware scheduling and load balancing in fog environment, Sustain. Comput. Inform. Syst. 24 (2019), 100355.

[4] S. Bitam, S. Zeadally, A. Mellouk, Fog computing job scheduling optimization based on bees swarm, Enterpr. Inform. Syst. 12 (2018), 373–397.

[5] D. Tychalas, H. Karatza, A scheduling algorithm for a fog computing system with bag-of-tasks jobs: simulation and performance evaluation, Simul. Modell. Pract. Theory 98 (2020), 101982.

[6] L.H. Kazem, Efficient resource allocation for time-sensitive IoT applications in cloud and fog environments, Int. J. Recent Technol. Eng. 8 (2019), 2356–2363.

[7] Cisco Knowledge Networking, Cisco Global Cloud Index: Forecast and Methodology, 2015-2020, white paper, Cisco Public, San Jose, 2016.

[8] P. Hu, S. Dhelim, H. Ning, T. Qiu, Survey on fog computing: architecture, key technologies, applications and open issues, J. Netw. Comput. Appl. 98 (2017), 27–42.

[9] R.K. Naha, S. Garg, D. Georgakopoulos, P.P. Jayaraman, L. Gao, Y. Xiang, et al., Fog computing: survey of trends, architectures, requirements, and research directions, IEEE Access 6 (2018), 47980–48009.

[10] Y. Liu, J.E. Fieldsend, G. Min, A framework of fog computing: architecture, challenges, and optimization, IEEE Access 5 (2017), 25445–25454.

[11] H. Rafique, M.A. Shah, S.U. Islam, T. Maqsood, S. Khan, C. Maple, A novel bio-inspired hybrid algorithm (NBIHA) for efficient resource management in fog computing, IEEE Access 7 (2019), 115760–115773.

[12] M. Ghobaei-Arani, A. Souri, F. Safara, M. Norouzi, An efficient task scheduling approach using moth-flame optimization algorithm for cyber-physical system applications in fog computing, Trans. Emerg. Telecommun. Technol. 31 (2020), e3770.

[13] J. Wang, D. Li, Task scheduling based on a hybrid heuristic algorithm for smart production line with fog computing, Sensors (Basel) 19 (2019), 1023.

[14] R. Mahmud, R. Kotagiri, R. Buyya, Fog computing: a taxonomy, survey and future directions, in: B. Di Martino, K.C. Li, L. Yang, A. Esposito (Eds.), Internet of Everything, Springer, Singapore, 2018, pp. 103–130.

[15] H.F. Atlam, R.J. Walters, G.B. Wills, Fog computing and the internet of things: a review, Big Data Cogn. Comput. 2 (2018), 10.

[16] F.A. Kraemer, A.E. Braten, N. Tamkittikhun, D. Palma, Fog computing in healthcare–a review and discussion, IEEE Access 5 (2017), 9206–9222.

[17] H. Wadhwa, R. Aron, Fog computing with the integration of internet of things: architecture, applications and future directions, 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), IEEE, Melbourne, Australia, 2018, pp. 987–994.

[18] A. Yousefpour, C. Fung, T. Nguyen, K. Kadiyala, F. Jalali, A. Niakanlahiji, et al., All one needs to know about fog computing and related edge computing paradigms: a complete survey, J. Syst. Architect. 98 (2019), 289–330.

[19] M. Ghobaei-Arani, A. Souri, A.A. Rahmanian, Resource management approaches in fog computing: a comprehensive review, J. Grid Comput. 18 (2012), 1–42.

[20] R. Sharma, S. Rani, Resource scheduling in fog computing: a review, Int. J. Adv. Stud. Sci. Res. 4 (2019), 883–886.

[21] D. Rahbari, M. Nickray, Computation offloading and scheduling in edge-fog cloud computing, J. Electron. Inform. Syst. 1 (2019), 26–36.

[22] P. Hosseinioun, M. Kheirabadi, S.R. Kamel Tabbakh, R. Ghaemi, aTask scheduling approaches in fog computing: a survey, Trans. Emerg. Telecommun. Technol. (2020), e3792.

[23] M.M. Mon, M.A. Khine, Scheduling and load balancing in cloud-fog computing using swarm optimization techniques: a survey, 2019. Available from: https://onlineresource.ucsy.edu.mm/bitstream/handle/123456789/1119/ICCA%202019%20Proceedings%20Book-pages-19-25.pdf?sequence=1&isAllowed=y.

[24] P. Singh, M. Dutta, N. Aggarwal, A review of task scheduling based on meta-heuristics approach in cloud computing, Knowl. Inform. Syst. 52 (2017), 1–51.

[25] S. Abrishami, M. Naghibzadeh, D.H.J. Epema, Cost-driven scheduling of grid workflows using partial critical paths, IEEE Trans. Parallel Distrib. Syst. 23 (2011), 1400–1414.

[26] L. Chunlin, L. Layuan, QoS based resource scheduling by computational economy in computational grid, Inform. Process. Lett. 98 (2006), 119–126.

[27] Y. Sun, F. Lin, H. Xu, Multi-objective optimization of resource scheduling in Fog computing using an improved NSGA-II, Wireless Personal Commun. 102 (2018), 1369–1385.

[28] J. Xu, Z. Hao, R. Zhang, X. Sun, A method based on the combination of laxity and ant colony system for cloud-fog task scheduling, IEEE Access 7 (2019), 116218–116226.
[29] G. Li, Y. Liu, J. Wu, D. Lin, S. Zhao, Methods of resource scheduling based on optimized fuzzy clustering in fog computing, Sensors (Basel) 19 (2019), 2122.
[30] J.C. Bezdek, R. Ehrlich, W. Full, FCM: the fuzzy c-means clustering algorithm, Comput. Geosci. 10 (1984), 191–203.
[31] J. Kennedy, R. Eberhart, Particle swarm optimization, Proceedings of ICNN'95 - International Conference on Neural Networks, IEEE, Perth, WA, Australia, 1995, pp. 1942–1948.
[32] K. Etminani, M. Naghibzadeh, A min-min max-min selective algorihtm for grid task scheduling, Proceedings of the 2007 3rd IEEE/IFIP International Conference in Central Asia on Internet, IEEE, Tashkent, Uzbekistan, 2007, pp. 1–7.
[33] M. Aazam, E.N. Huh, Fog computing micro datacenter based dynamic resource estimation and pricing model for IoT, Proceedings of the 2015 IEEE 29th International Conference on Advanced Information Networking and Applications, IEEE, Gwangiu, South Korea, 2015, pp. 687–694.
[34] C. Wu, W. Li, L. Wang, A. Zomaya, Hybrid evolutionary scheduling for energy-efficient fog-enhanced internet of things, IEEE Trans. Cloud Comput. Early Access (2018), 1.
[35] H. Topcuoglu, S. Hariri, M.y. Wu, Performance-effective and low-complexity task scheduling for heterogeneous computing, IEEE Trans. Parallel Distrib. Syst. 13 (2002), 260–274.
[36] X. Zhang, H. Duan, J. Jin, DEACO: hybrid ant colony optimization with differential evolution, Proceedings of the 2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence), IEEE, Hong Kong, China, 2008, pp. 921–927.
[37] B.M. Nguyen, H.T.T. Binh, T.T. Anh, D.B. Son, Evolutionary algorithms to optimize task scheduling problem for the IoT based bag-of-tasks application in cloud–fog computing environment, Appl. Sci. 9 (2019), 1730.
[38] L. Zhang, Q. Fu, J. Chen, H. Bai, X. Zhou, A modified particle swarm optimization algorithm—CPSODE, Proceedings of the 2017 29th Chinese Control And Decision Conference (CCDC), IEEE, Chongqing, China, 2017, pp. 6659–6663.
[39] H.R. Boveiri, R. Khayami, M. Elhoseny, M. Gunasekaran, An efficient swarm-intelligence approach for task scheduling in cloud-based internet of things applications, J. Ambient Intell. Human. Comput. 10 (2019), 3469–3479.
[40] T.L. Adam, K.M. Chandy, J.R. Dickson, A comparison of list schedules for parallel processing systems, Commun. ACM 17 (1974), 685–690.
[41] B. Kruatrachue, T. Lewis, Duplication scheduling heuristics (DSH): a new precedence task scheduler for parallel processor systems, Oregon State University, Corvallis, OR, 1987.
[42] M.Y. Wu, D.D. Gajski, Hypertool: a programming aid for message-passing systems, IEEE Trans. Parallel Distrib. Syst. 1 (1990), 330–343.
[43] J.J. Hwang, Y.C. Chow, F.D. Anger, C.Y. Lee, Scheduling precedence graphs in systems with interprocessor communication times, SIAM J. Comput. 18 (1989), 244–257.
[44] G.C. Sih, E.A. Lee, A compile-time scheduling heuristic for interconnection-constrained heterogeneous processor architectures, IEEE Trans. Parallel Distrib. Syst. 4 (1993), 175–187.
[45] L. Yin, J. Luo, H. Luo, Tasks scheduling and resource allocation in fog computing based on containers for smart manufacturing, IEEE Trans. Ind. Inform. 14 (2018), 4712–4721.
[46] X.Q. Pham, N.D. Man, N.D.T. Tri, N.Q. Thai, E.N. Huh, A cost- and performance-effective approach for task scheduling based on collaboration between cloud and fog computing, Int. J. Distrib. Sens. Netw. 13 (2017), 1550147717742073.
[47] J. Li, S. Su, X. Cheng, Q. Huang, Z. Zhang, Cost-conscious scheduling for large graph processing in the cloud, Proceedings of the 2011 IEEE International Conference on High Performance Computing and Communications, IEEE, Banff, AB, Canada, 2011, pp. 808–813.
[48] D. Hoang, T.D. Dang, FBRC: optimization of task scheduling in fog-based region and cloud, Proceedings of the 2017 IEEE Trustcom/BigDataSE/ICESS, IEEE, Sydney, NSW, Australia, 2017, pp. 1109–1114.
[49] G.L. Stavrinides, H.D. Karatza, A hybrid approach to scheduling real-time IoT workflows in fog and cloud environments, Multimed. Tools Appl. 78 (2019), 24639–24655.
[50] E.S. Alkayal, N.R. Jennings, M.F. Abulkhair, Efficient task scheduling multi-objective particle swarm optimization in cloud computing, Proceedings of the 2016 IEEE 41st Conference on Local Computer Networks Workshops (LCN Workshops), IEEE, Dubai, United Arab Emirates, 2016, pp. 17–24.
[51] E. Ghaffari, Providing a new scheduling method in fog network using the ant colony algorithm, Collect. Articles Comput. Sci. (2019). Available from: https://www.scipedia.com/public/Ghaffari_2019a.
[52] S. Dam, G. Mandal, K. Dasgupta, P. Dutta, An ant-colony-based meta-heuristic approach for load balancing in cloud computing, in: S. Khalid (Ed.), Applied Computational Intelligence and Soft Computing in Engineering, IGI Global, Hershey, PA, 2018, pp. 204–232.
[53] Y. Yang, S. Zhao, W. Zhang, Y. Chen, X. Luo, J. Wang, DEBTS: delay energy balanced task scheduling in homogeneous fog networks, IEEE Internet Things J. 5 (2018), 2094–2106.
[54] T. Wang, Z. Liu, Y. Chen, Y. Xu, X. Dai, Load balancing task scheduling based on genetic algorithm in cloud computing, Proceedings of the 2014 IEEE 12th International Conference on Dependable, Autonomic and Secure Computing, IEEE, Dalian, China, 2014, pp. 146–152.
[55] S. Ningning, G. Chao, A. Xingshuo, Z. Qiang, Fog computing dynamic load balancing mechanism based on graph repartitioning, China Commun. 13 (2016), 156–164.
[56] C.K. Chen, Y.H. Chang, Y.T. Chen, C.C. Yang, J.K. Lee, Switching supports for stateful object remoting on network processors, J. Supercomput. 40 (2007), 281–298.
[57] X. Xu, S. Fu, Q. Cai, W. Tian, W. Liu, W. Dou, et al., Dynamic resource allocation for load balancing in fog environment, Wireless Commun. Mobile Comput. 2018 (2018), 6421607.
[58] K.C. Lin, Y.H. Huang, J.C. Hung, Y.T. Lin, Modified cat swarm optimization algorithm for feature selection of support vector machines, in: J. Park, A. Zomaya, H.Y. Jeong, M. Obaidat (Eds.), Frontier and Innovation in Future Computing and Communications, Springer, Dordrecht, 2014, pp. 329–336.
[59] U. Schwiegelshohn, R. Yahyapour, Analysis of first-come-first-serve parallel job scheduling, Proceedings of the Ninth Annual ACM-SIAM Symposium On Discrete Algorithms (SODA), Association for Computation Machinery, San Francisco, California, USA, 1998, pp. 629–638.
[60] G. Li, J. Yan, L. Chen, J. Wu, Q. Lin, Y. Zhang, Energy consumption optimization with a delay threshold in cloud-fog cooperation computing, IEEE Access 7 (2019), 159688–159697.
[61] Q. Liu, Y. Wei, S. Leng, Y. Chen, Task scheduling in fog enabled internet of things for smart cities, Proceedings of the 2017 IEEE

17th International Conference on Communication Technology (ICCT), IEEE, Chengdu, China, 2017, pp. 975–980.

[62] X.Q. Pham, E.N. Huh, Towards task scheduling in a cloud-fog computing system, Proceedings of the 2016 18th Asia-Pacific Network Operations and Management Symposium (APNOMS), IEEE, Kanazawa, Japan, 2016, pp. 1–4.

[63] D. Rahbari, M. Nickray, Scheduling of fog networks with optimized knapsack by symbiotic organisms search, Proceedings of the 2017 21st Conference of Open Innovations Association (FRUCT), IEEE, Helsinki, Finland, 2017, pp. 278–283.

[64] M.Y. Cheng, D. Prayogo, Symbiotic organisms search: a new metaheuristic optimization algorithm, Comput. Struct. 139 (2014), 98–112.

[65] M.A. Rodriguez, R. Buyya, A responsive knapsack-based algorithm for resource provisioning and scheduling of scientific workflows in clouds, Proceedings of the 2015 44th International Conference on Parallel Processing, IEEE, Beijing, China, 2015, pp. 839–848.

[66] R.O. Aburukba, M. AliKarrar, T. Landolsi, K. El-Fakih, Scheduling Internet of Things requests to minimize latency in hybrid Fog–Cloud computing, Future Gen. Comput. Syst. 111 (2020), 539–551.

[67] S. Agarwal, S. Yadav, A.K. Yadav, An efficient architecture and algorithm for resource provisioning in fog computing, Int. J. Inform. Eng. Electron. Bus. 8 (2016), 48–61.

[68] D. Rahbari, M. Nickray, Low-latency and energy-efficient scheduling in fog-based IoT applications, Turk. J. Electr. Eng. Comput. Sci. 27 (2019), 1406–1427.

[69] L.F. Bittencourt, J. Diaz-Montes, R. Buyya, O.F. Rana, M. Parashar, Mobility-aware application scheduling in fog computing, IEEE Cloud Comput. 4 (2017), 26–35.

[70] A. Ghenai, Y. Kabouche, W. Dahmani, Multi-user dynamic scheduling-based resource management for Internet of Things applications, Proceedings of the 2018 International Conference on Internet of Things, Embedded Systems and Communications (IINTEC), IEEE, Hamammet, Tunisia, 2018, pp. 126–131.

[71] K. Kishor, V. Thapar, An efficient service broker policy for cloud computing environment, Int. J. Comput. Sci. Trends Technol. 2 (2014), 104–109.

[72] R. Xu, Y. Wang, Y. Cheng, Y. Zhu, Y. Xie, A.S. Sani, et al., Improved particle swarm optimization based workflow scheduling in cloud-fog environment, International Conference on Business Process Management, Springer, Cham, 2018, pp. 337–347.

[73] H.Y. Wu, C.R. Lee, Energy efficient scheduling for heterogeneous fog computing architectures, Proceedings of the 2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC), IEEE, Tokyo, Japan, 2018, pp. 555–560.

[74] A.F.T. Martins, N.A. Smith, E.P. Xing, Concise integer linear programming formulations for dependency parsing, Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP, Association for Computational Linguistics, Stroudsburg, PA, US, 2009, pp. 342–350.

[75] S. Kabirzadeh, D. Rahbari, M. Nickray, A hyper heuristic algorithm for scheduling of fog networks, Proceedings of the 2017 21st Conference of Open Innovations Association (FRUCT), IEEE, Helsinki, Finland, 2017, pp. 148–155.

[76] M. Dorigo, M. Birattari, T. Stützle, Ant colony optimization, IEEE Comput. Intell. Mag. 1 (2006), 28–39.

[77] X. Liu, J. Liu, A task scheduling based on simulated annealing algorithm in cloud computing, Int. J. Hybrid Inform. Technol. 9 (2016), 403–412.

[78] D. Whitley, A genetic algorithm tutorial, Stat. Comput. 4 (1994), 65–85.

[79] B. Jamil, M. Shojafar, I. Ahmed, A. Ullah, K. Munir, H. Ijaz, A job scheduling algorithm for delay and performance optimization in fog computing, Concurr. Comput. Pract. Exp. 32 (2020), e5581.