Article

A Cost-Aware Framework for QoS-Based and Energy-Efficient Scheduling in Cloud–Fog Computing

Husam Suleiman

Department of Computer Engineering, College of Computer and Information Technology, Jordan University of Science and Technology, Irbid 22110, Jordan; hasuleiman@just.edu.jo

Abstract: Cloud–fog computing is a large-scale service environment developed to deliver fast, scalable services to clients. The fog nodes of such environments are distributed in diverse places and operate independently by deciding on which data to process locally and which data to send remotely to the cloud for further analysis, in which a Service-Level Agreement (SLA) is employed to govern Quality of Service (QoS) requirements of the cloud provider to such nodes. The provider experiences varying incoming workloads that come from heterogeneous fog and Internet of Things (IoT) devices, each of which submits jobs that entail various service characteristics and QoS requirements. To execute fog workloads and meet their SLA obligations, the provider allocates appropriate resources and utilizes load scheduling strategies that effectively manage the executions of fog jobs on cloud resources. Failing to fulfill such demands causes extra network bottlenecks, service delays, and energy constraints that are difficult to maintain at run-time. This paper proposes a joint energy- and QoS-optimized performance framework that tolerates delay and energy risks on the cost performance of the cloud provider. The framework employs scheduling mechanisms that consider the SLA penalty and energy impacts of data communication, service, and waiting performance metrics on cost reduction. The findings prove the framework's effectiveness in mitigating energy consumption due to QoS penalties and therefore reducing the gross scheduling cost.

Keywords: SLA-based scheduling; cloud–fog computing; resource allocation; QoS optimization; energy-efficient scheduling; genetic algorithms

Citation: Suleiman, H. A Cost-Aware Framework for QoS-Based and Energy-Efficient Scheduling in Cloud–Fog Computing. *Future Internet* **2022**, *14*, 333. https://doi.org/10.3390/fi14110333

Academic Editors: Konstantinos Oikonomou, Georgios Tsoumanis and Athanasios Tsipis

Received: 9 October 2022
Accepted: 7 November 2022
Published: 14 November 2022

1. Motivation

The continued progression in service and the demand for short response delays play a vital role in shaping the performance of large-scale, service-based environments. Cloud–fog computing is an example of such an environment developed to deliver fast executions and scalable services [1–4]. Fog computing in particular presents a decentralized computing infrastructure, in which services and resources could be connected to a cloud [5,6]. It is based on bringing the intelligence and processing power of the cloud to places close to where data are generated and acted upon, which are called edge devices or fog gateways. The goal is to process as much data as possible locally on fog computing nodes co-located with fog devices, so as to mitigate latency and bandwidth requirements of processing such data entirely on a remote cloud [7–9].

Fog nodes are typically connected to distributed smart sensors and IoT devices that collect data from an operating environment [10–12]. Each fog node by itself operates independently, as it decides on which data to process locally and which data to send remotely to the cloud for further analysis [13–15]. Short-term jobs delivered from such sensor-based and IoT devices are processed locally in their corresponding fog nodes, whereas resource-intensive jobs are sent by fog nodes (in the form of fog jobs) to the cloud computing environment. Nevertheless, a huge volume of fog jobs is still transmitted by fog nodes to the cloud service provider for further processing and analysis [16,17]. Such jobs have various operational characteristics [18,19], time-sensitivities [20,21], energy constraints [22–24], and service costs [25,26]. They tend to arrive in a random manner to the cloud service provider, as well as entail different QoS requirements and service demands that are to be fulfilled. Thus, reliability and network bandwidth are challenged in satisfying the obligations of such data in the cloud.

A cloud provider in turn experiences heterogeneous fog workloads of several timing variations and service difficulties [27–29]. Such workloads increase not only in volume but also in complexity, which strictly forces execution latencies and energy complications on the cloud service provider [30,31]. The latter utilizes a pool of cloud resources in its data center to accommodate incoming fog workloads and thus allocates sufficient resources to achieve reliability and economies of scale. To execute such workloads, the service provider employs job scheduling and balancing schemes so that cloud resources are effectively utilized.

However, a client of a fog node typically tends to request a service that is both cost-efficient and high in performance [32,33]. To effectively meet such fog requests, a cloud service provider strives to maintain an efficient cost of service and energy consumption. Existing scheduling strategies often account for optimizing system performance based on response time and resource utilization metrics. Recent scheduling strategies start to incorporate factors of energy consumption when scheduling decisions are triggered.

A major limitation in such service schedulers is that they are not developed to manage mutual performance impacts between the fog service environment and the cloud computing environment. Such schedulers do not predict resource workloads, which is due to the lack of the load-management frameworks required to constantly measure system bottlenecks in fog environments along with cloud-resource queues.

Furthermore, such schedulers adopt allocation methods that focus on improving performance by only deciding on an optimal allocation of each individual fog job based on available cloud resources, so that particular performance metrics and QoS penalties are enhanced, in which, however, the energy constraints of fog jobs due to such backlog bottlenecks and SLA violations are not employed. As such, existing strategies do not optimize the energy efficiency performance based on the QoS obligations of fog jobs, resulting in the metric of measuring client satisfaction formulating schedules that lack a joint energy- and QoS-optimized performance.

2. Objectives

A cloud service provider must employ a management framework that utilizes various SLA requirements, energy complications, service demands, and cost obligations of fog jobs of different characteristics to effectively complete workload executions. The framework must formalize cost-aware schedules that employ:

- The communication bandwidth allocated by fog environments;
- The waiting delays incurred due to backlog bottlenecks delivered by fog nodes and congestion in the resource queues of the cloud;
- The cost of services provided by the cloud;
- The energy constraints developed due to serving such fog jobs;
- The SLA penalties of fog jobs incurred due to service violations.

The scheduling framework is intended to provide services to fog jobs with heavy computations, for which it utilizes the power and speed of a remote cloud computing data center to serve heavy fog demands that cannot be processed locally in the fog environment due to the performance limitations of fog nodes.

3. Problem Statement

Recent advances in critical cloud–fog computing infrastructures as service-based dynamic environments have accelerated the deployment of intelligent devices to accomplish specific tasks and achieve market goals. Consider the example of applying IoT and sensory-based fog systems in vehicular networks. Some time-critical data, such as accident data, are of high importance to ambulance/police departments and must be sent to control systems equipped with powerful computing resources to take countermeasures. Delays in the processing of such data result in monetary and catastrophic effects, in which QoS requirements and agreements produce penalties reflective of such effects. Hence, the integration of such devices into cloud–fog computing environments constructs reliable networks that can further monitor, collect, analyze, and process data efficiently.

Constraints on energy consumption, response time, and bandwidth requirements are inevitable challenges that have further attracted the attention of researchers. A fog computing environment increasingly produces a large volume of jobs waiting to receive cloud services, each of which consumes resources and energy to strictly meet SLA obligations. The growing volume of fog data potentially produces backlog bottlenecks that cause execution difficulties on cloud computing resources, which are structured with computational processing power to tackle complex fog workloads.

Operation costs on fog clients and cloud service providers increase by increasing not only the cost of servicing fog jobs and SLA violation penalties but also the cost of the energy required to communicate and execute such fog demands. For instance, consider a fog node that requests a task to be serviced on a remote cloud within a specific tardiness limit. The longer the waiting and service times of the fog job in the cloud–fog environment, the higher the service cost and energy consumption required to meet the job's demand. Similarly, the performance metrics of the response and execution times incur high performance costs when their time values increase in the service environment.

The research question arises when a scheduler formulates fog jobs for execution using the computing power of cloud resources so that QoS requirements are met while energy is concurrently saved. In this paper, the performance enhancement is focused on the side of the cloud computing environment, and thus the problem addressed is stated as follows:

Consider the case of fog nodes that deliver job workloads of various QoS expectations and energy demands to a cloud computing environment that comprises identical computing resources to service fog jobs. Each fog job is subject to SLA obligations that define constraints of service cost and execution energy. It is required to deploy and service fog jobs in the cloud computing environment such that energy is preserved and the cost of service is mitigated.

4. Background and Related Work

Cloud computing, as a distributed paradigm, hosts heterogeneous resources pooled in data centers to serve the demands of applications and IoT jobs [1,34,35]. Such demands present challenges to the cloud computing environment in meeting the QoS obligations of clients, in which a cloud service provider strives to provision sufficient resources so as to meet the jobs' execution demands and satisfy their SLA requirements [36,37]. However, the huge number of cloud resources allocated brings energy challenges to cloud data centers, in which energy consumption greatly increases and so does the cost of operation [38–40]. Together with QoS, energy consumption in cloud computing has consequently attracted the attention of researchers from academia and industry [41–43]. It becomes of paramount importance for a cloud provider to satisfy the QoS obligations of clients while simultaneously achieving energy efficiency based on QoS during the scheduling process.

The existing work in the literature presents various execution models and techniques to tackle such service challenges in the cloud–fog environment. A model for assigning tasks to servers is formulated in Dong et al. [44] with the goal of minimizing the energy consumption of servers in the cloud data center. They propose a scheduling scheme that allocates tasks to a minimum number of servers while the response time of jobs is kept within an acceptable range, in which the scheme has proven its effectiveness against the random-based scheduling. Li et al. [25] propose a load balancing model that ensures user satisfaction by efficiently executing tasks at a reduced cost. However, energy efficiency is not effectively incorporated with respect to the QoS penalties of schedules in the execution procedure.

Tadakamalla et al. [45] present a model that controls the fraction of data processed remotely on cloud servers against the fraction of data processed locally on fog servers, where a utility function is proposed to optimize the performance metrics of average response time and cost. The task scheduling process has been optimized on a cloud–fog computing environment by Dang et al. [46], in which efficient scheduling algorithms assign jobs among fog regions and clouds. Tsai et al. [47] adopt an optimal task scheduling procedure that considers the operating cost and execution time of a task in a cloud–fog computing environment. The procedure particularly formulates globally optimal schedules that are computed based on task requirements and usage costs of resources. Nevertheless, jobs are modeled without considering energy factors when service decisions are triggered in the algorithm designs.

Furthermore, Guo et al. [48] decided on the optimal scheduling of virtual machines on queues of the cloud system with heterogeneous workloads, but with considerations of energy in the scheduling process. In contrast, Anjos et al. [49] present an algorithm that selects a suitable cloud or mobile-edge computing server to schedule IoT workloads, with the goal of achieving a better service time with a low cost. The energy required to perform such tasks are employed in the scheduling process, however, without correlating energy consumption with a QoS penalty of schedules formulated on resources.

In addition, an optimization framework to meet the deadlines of cloud applications is proposed in Alamro et al. [50], in which they utilize a Probability of Completion before Deadlines (PoCD) metric to quantify the probability of a job to meet its deadline. Perret et al. [51] present a deadline-based scheduler that orders jobs for execution in the cloud according to their laxity and locality, in which the algorithm demonstrates its efficacy against time-shared and space-shared scheduling algorithms. In both deadline schedulers, penalties for the energy consumption incurred due to executing a job and for violating deadlines of jobs are missing factors.

A delay-aware Earliest Deadline First (EDF) algorithm is proposed by Sharma et al. [52] that allocates tasks for execution on a four-tier architecture. The algorithm demonstrates its effectiveness in improving the performance of energy consumption during the execution and scheduling processes of tasks. Wu et al. [53] also present an energy-efficient scheduling algorithm that minimizes the energy consumption for IoT workflows. Xue et al. [54] propose a scheduling algorithm to minimize the energy consumption in the cloud computing environment. They present a QoS model with respect to response time and throughput of jobs, as well as present an energy model for physical machines of the cloud environment. However, proposed algorithms do not measure mutual performance impacts between the energy consumption of machines and the QoS requirements of jobs.

In addition, the genetic algorithm, as a metaheuristic approach, has been extensively applied to mitigate the complexity of scheduling problems. Nguyen et al. [55] tackle the scheduling process in cloud–fog computing systems by formulating a model that accounts for different performance constraints and applying metaheuristic approaches to solve a multi-objective optimization scheduling problem. Similarly, such approaches are applied by Ben-Alla et al. [56] to propose a job scheduling method for cloud environments based on dynamic dispatch queues.

Arora et al. [57] analyze popular first-come first-served, shortest job first, round robin, Min-Min, Max-Min, genetic, and ant colony optimization scheduling algorithms by comparing them in terms of response time and makespan. Thus, the genetic approach has proven its effectiveness in achieving the best performance on the metric of response time, whereas the ant colony optimization algorithm outperforms other scheduling algorithms in terms of makespan. In addition, the genetic approach has been utilized in Salido et al. [58] to solve the job-shop scheduling problem and produce a good-quality, energy-efficient scheduling solution in a reasonable time. Their approach adopts machine resources that are modeled to consume different energy rates for processed jobs. Zhang et al. [59] also minimize the energy consumption in a job-shop scheduling problem by utilizing a multi-objective, genetic-based approach.

In contrast, Lin et al. [60] propose a framework in which they employ modern artificial intelligence techniques to overcome limitations of traditional heuristic-based scheduling algorithms and cope with dynamic changes in cloud environments. The framework utilizes the power of the deep learning approach to propose a model for scheduling jobs in cloud data centers. In addition, the framework adopts a deep Q-network model for resource allocation to deploy virtual machines to physical servers to execute jobs. Moreover, a scheduling scheme is proposed by Cui et al. [61] to minimize average waiting times and makespans under different deadline constraints, in which a reinforcement-learning-based approach is utilized in a grid and in Infrastructure-as-a-Service (IaaS) cloud computing environments.

Furthermore, Zhang et al. [62] propose a resource management framework for virtualized two-tier cloud data center environments, in which the framework demonstrates better performance in improving resource utilization and obtains an energy saving of 13.8%. The enhancement in energy consumption is also achieved in Zhao et al. [63] by a multi-objective scheduling algorithm, as well as in Paul et al. [64], in which a commonly used approach in control theory called model predictive control is utilized to address the scheduling problem for deferrable jobs in a tiered architecture data center.

Overall, the proposed scheduling methods in existing frameworks and models do not optimize the performance of energy efficiency based on the QoS obligations of fog jobs. It is required that client satisfaction is to be measured based on a metric that accounts for joint energy andQoS performance optimization, and hence, a pragmatic satisfaction between the fog jobs of the cloud environment and service providers is met. Such satisfactions are to be assessed by deriving a performance metric that measures the QoS of fog jobs so as to penalize the amounts of energy consumption and violations of service and subsequently formulare schedules across the cloud–fog computing environment.

5. Contributions

A service management framework is designed to incorporate the QoS penalty and energy consumption of fog jobs waiting for execution in resource queues of the cloud environment such that cost of service and energy is mitigated. The framework employs an analytical model for communication and computational performance metrics derived to calculate the service cost. In this perspective, the communication bandwidth is the performance metric that affects the QoS delivered to fog clients. A high bandwidth allocated to fog nodes can, for instance, mitigate the latency incurred from the transmission time of data jobs. Moreover, the resource demands of fog jobs proportionally influence transmission and computational energy, and as a result, energy constraints affect job latencies in that fog jobs with high resource demands demand high communication and computational energy consumption.

The management framework employs the following: (i) the allocation cost of the communication bandwidth assigned for a job delivered from a fog node; (ii) the waiting cost of a fog job in the resource queues of the cloud computing environment; (iii) the execution cost of a fog job in the cloud resource allocated for it; and (iv) the SLA violation cost if the service of a fog job does not meet its QoS deadline and tardiness constraints. The contributions of this paper are summarized as follows:

- Designing a cost model based on a performance metric derived by utilizing QoS obligations and energy demands of fog jobs transmitted for execution in the cloud computing environment, in which the performance of energy efficiency is optimized based on the QoS of fog jobs;
- Employing information of resource usage required by fog workloads to decide on their optimal allocation to cloud resources, so as to serve the demands of fog nodes such that the cost of service is mitigated;
- Considering mutual performance impacts between quality metrics of fog jobs allocated for execution and factors of energy consumption required to service such jobs, so as to achieve pragmatic client satisfaction and mitigate the gross energy cost of job workloads queued for execution across the cloud–fog environment;

- Mitigating the management complexity of the scheduling model so that schedules of minimum cost of QoS and energy penalties are evolved in a reasonable time.

Overall, schedules are formulated based on a joint energy- and QoS-optimized performance wherein the performance is predicted and evaluated using the cost of service and energy. Optimal schedules are thus formulated on queues of cloud resources such that the cost of service and energy are minimized, so as to maximize the probability of satisfied clients. The system performance is assessed on modeled fog jobs generated with heterogeneous service characteristics, energy demands, and SLA penalties.

6. Service Management Framework

The scheduling and allocation framework is developed to manage the execution of fog jobs in the cloud–fog computing environment. The framework is analyzed on a service-based environment modeled by employing a queuing system that represents a fog layer with IoT devices and a cloud layer with a job dispatcher associated with cloud resources. Table 1 shows an alphabetical summary of the notations and concepts used in the paper.

Table 1. Summary of notations.

Notation	Definition	Notation	Definition
a_i	Arrival time of a fog job J_i to the cloud layer	μ	Service rate of a server
β	Schedule ordering for a set of fog jobs	m	Index of a cloud resource
$c\omega^{\beta}_{i\|cd}$	The time spent by a fog job J_i in the dispatcher's queue	μ_{Γ}	Bandwidth penalty mean
$c\omega^{\beta}_{i}$	The waiting time of a fog job J_i governed by ordering β in resource queues $\boldsymbol{Q}$ of the cloud environment	μ_{ω}	Waiting penalty mean
$c^{(t)}_{i}$	Target completion time of a fog job J_i	$\mu_{\mathcal{E}}$	Penalty execution mean
$\mathcal{C}$	Penalty cost	μ_{α}	SLA penalty mean
d_i	The departure time of a fog job J_i from the cloud layer	n	Maximum number of cloud resources
$\mathcal{E}_i$	Prescribed service time of fog job J_i in the cloud layer	ρ^{Γ}_{i}	Penalty cost of bandwidth usage per time unit of data
ϵ	Energy cost	ρ^{ω}_{i}	Penalty cost of waiting $t\omega^{\beta}_{i}$ for a fog job J_i
e_{γ,Γ_i}	Energy cost per time unit of bandwidth allocation	$\rho^{\mathcal{E}}_{i}$	Penalty cost of servicing a fog job J_i in a cloud resource R_m
$E_{i,\Gamma}$	Total bandwidth energy cost of a fog job J_i	ρ^{α}_{i}	Penalty cost of SLA violation of a fog job J_i in a cloud resource R_m
e_{m,ω_i}	Energy cost per time unit of waiting $t\omega^{\beta}_{i}$ in a resource queue Q_m	$\boldsymbol{Q}$	A set of cloud queues
$E_{i,\omega}$	Total cost of waiting energy of a fog job J_i	Q_m	The mth cloud queue
$e_{m,\mathcal{E}_i}$	Energy cost per time unit of service $\mathcal{E}_i$ in the cloud resource R_m	q	Communication bit-rate
$E_{i,\mathcal{E}}$	Total cost of service energy of a fog job J_i	$\boldsymbol{R}$	A set of cloud computing resources
e_{m,α_i}	Energy cost per time unit of SLA violation α^{β}_{i} with the cloud service provider	R_m	The mth cloud resource
$E_{i,\alpha}$	Total cost of SLA-violation energy of a fog job J_i	rt^{β}_{i}	The response time of a fog job J_i governed by ordering β across the cloud–fog environment
ξ_i	Service cost per time unit of execution $\mathcal{E}_i$	$t\omega^{\beta}_{i}$	The total waiting time of a fog job J_i governed by ordering β across the cloud–fog environment
$\digamma$	Maximum number of allocations exist	u	Energy consumption per bit in the fog layer
$\mathbb{F}$	A set of fog nodes	ν	Arbitrary scaling factor
F_g	gth fog node	ψ_i	Waiting cost per time unit of waiting $t\omega^{\beta}_{i}$
$f\omega^{\beta}_{i}$	The waiting time of a fog job J_i governed by ordering β in the fog environment	χ^{Γ}_{i}	Scaling factor on penalty cost of bandwidth usage
g	Index of a fog node	χ^{ω}_{i}	Scaling factor on penalty cost of waiting
G	Maximum number of fog nodes	$\chi^{\mathcal{E}}_{i}$	Scaling factor on penalty cost of service
i	Index of a fog job	χ^{α}_{i}	Scaling factor of penalty cost SLA violation
$\boldsymbol{J}$	A set of fog jobs	$z_{i,m}$	Allocation of a fog job J_i on queue Q_m of cloud resource R_m
J_i	The ith fog job	ζ_i	SLA cost incurred per time unit of SLA violation α^{β}_{i}
κ_{Γ}	Monetary cost factor for bandwidth allocation penalty	α^{β}_{i}	SLA violation for a fog job J_i
κ_{ω}	Monetary cost factor for waiting penalty	Γ_i	Bandwidth allocated for a fog job J_i in the fog layer
$\kappa_{\mathcal{E}}$	Monetary cost factor for execution penalty	Λ_i	The cost of bandwidth usage incurred per data unit of a fog job J_i
κ_{α}	Monetary cost factor for SLA violation penalty	λ_{Γ}	Rate of energy consumption g
ℓ	Maximum number of fog jobs in the stream	$\lambda_{\mathcal{E}}$	Rate of energy cost $e_{m,\mathcal{E}_i}$
$\mathcal{L}_i$	Service deadline of a fog job J_i	λ_{ω}	Rate of energy cost e_{m,ω_i}
$l\omega_i$	Tardiness allowance for a fog job J_i	λ_{α}	Rate of energy cost e_{m,α_i}

6.1. System Architecture and Queuing System Model

The architecture of the cloud–fog computing environment consists of a fog tier and a cloud tier, as shown in Figure 1. The fog tier comprises a set of fog devices interconnected together that deliver jobs of different service characteristics to fog nodes $\mathbb{F}$ as follows:

$$\mathbb{F} = F_1, F_2, \ldots, F_G, \quad \forall g \in [1, G]. \tag{1}$$

Jobs received by a fog node F_g are atomic and independent of each other; they hold no information to be exchanged with other jobs from other fog nodes. Data that cannot be processed locally on fog nodes $\mathbb{F}$ are transmitted to a remote cloud for further analysis and processing. In this paper, the performance enhancement is modeled to tackle the execution of those jobs sent to the cloud, in that it models the portion of the data analyzed and processed in the cloud. The cloud tier is structured with a number of homogeneous servers, each of which entails a queue with infinite capacity to buffer incoming fog jobs for execution in its computing resource. Factors incurred due to server failures and server-to-server communications are not considered.

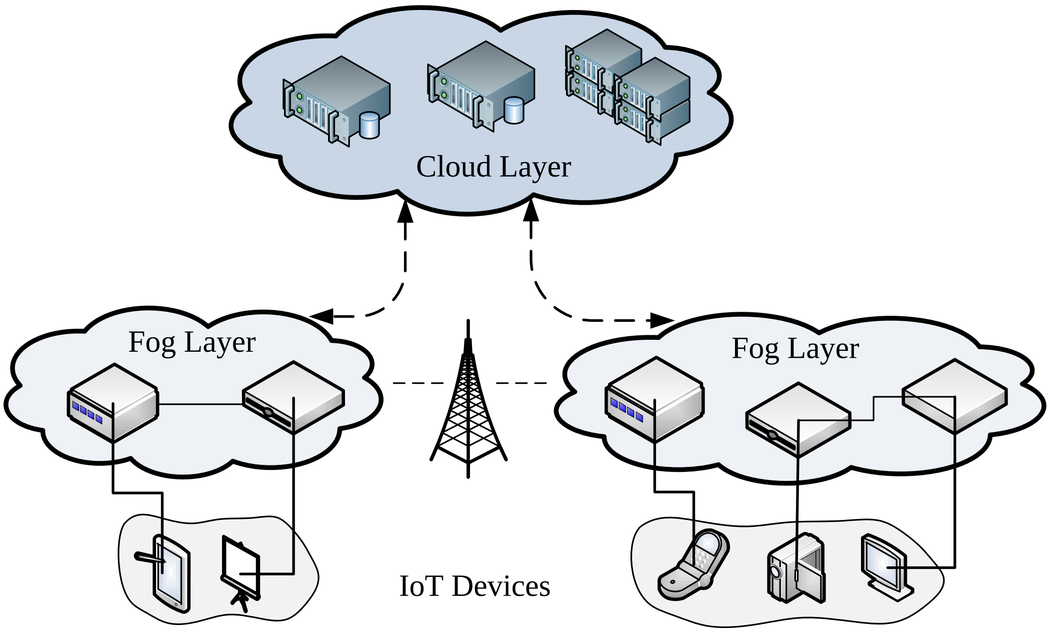

Figure 1. Cloud–fog architecture.

The *M*/*M*/*N* queuing model is adopted to design the cloud computing system, where *N* represents the number of cloud resources that exist in the system, as shown in Figure 2. A system-queue, called a cloud dispatcher *cd*, receives fog environment jobs and allocates them for execution on cloud resources for further processing. The arrival behavior of fog jobs to the dispatcher *cd* of the cloud tier is modeled as a Poisson process. The time between consecutive arrivals of such fog jobs follows an exponential distribution with a particular arrival rate. The service demand of each fog job in a cloud resource is assumed to be known in advance based on prediction methods applied on incoming workload history to estimate a job's execution time, and thus it is modeled from an exponential distribution with a service rate μ [65,66].

Fog jobs allocated by the cloud dispatcher *cd* to resource queues are allowed to be both reordered in the same queue and migrated from one queue to another. A fog job can be executed by only one cloud resource at a time. A cloud resource can execute only one fog job at a time. Cloud resources are available to provide services at any time. The service discipline of such fog jobs in cloud resources is non-preemptive; a fog job cannot be interrupted once it starts data execution in a cloud resource.

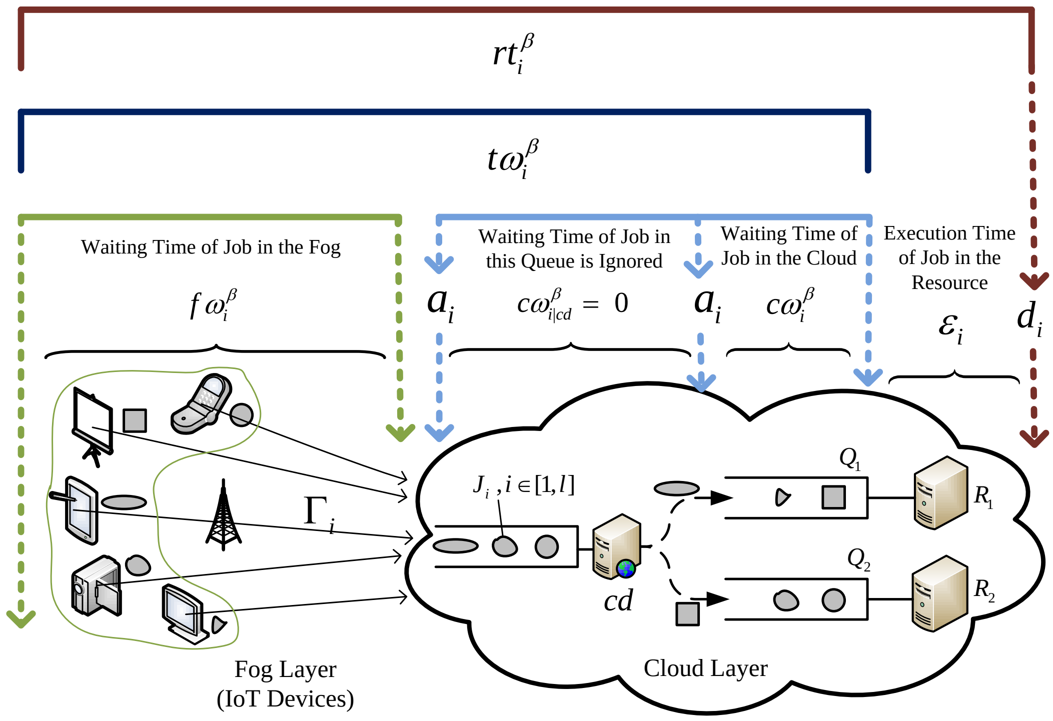

Figure 2. System architecture.

6.2. Cost Analytical Model

The scheduling optimality problem is formalized for a given bag of fog jobs waiting to receive services from the cloud computing environment. The design of the cloud system employs a set of identical computing resources $\boldsymbol{R}$, namely, virtual machines, available to service fog demands:

$$\boldsymbol{R} = \{R_1, R_2, R_3, \ldots, R_n\}, \quad \forall m \in [1, n] \tag{2}$$

Each resource R_m in the cloud environment entails a queue Q_m that holds incoming fog jobs waiting to receive service, which formulates a queuing system $\boldsymbol{Q}$ that reflects resources $\boldsymbol{R}$ in the cloud environment as follows:

$$\boldsymbol{Q} = \{Q_1, Q_2, Q_3, \ldots, Q_n\}, \quad \forall m \in [1, n] \tag{3}$$

A set of ℓ atomic, independent fog jobs $\boldsymbol{J}$ are delivered from fog nodes and received by the dispatcher *cd* of the cloud environment as follows:

$$\boldsymbol{J} = \{J_1, J_2, J_3, \ldots, J_\ell\}, \quad \forall i \in [1, \ell] \tag{4}$$

Fog jobs $\boldsymbol{J}$ arrive in a random manner to the cloud dispatcher *cd*. The index i of each fog job J_i indicates and signifies its arrival ordering to the dispatcher *cd*. For instance, J_1 is the first job to arrive, J_2 is the second job, and so on. Jobs allocated by the dispatcher *cd* are queued in cloud resources $\boldsymbol{R}$ for execution based on a scheduling order β described as follows:

$$\beta = \bigcup_{m=1}^{n} \mathrm{I}(Q_m) \tag{5}$$

where $\mathrm{I}(Q_m)$ represents indices of fog jobs in the resource-queue Q_m. For instance, $\mathrm{I}(Q_2) = \{4, 1, 3, 6\}$ signifies that fog jobs J_4, J_1, J_3, and J_6 are queued in Q_2 such that fog job J_4 precedes J_1, which in turn precedes J_3, and so on. It is assumed that $z_{i,m}$ represents an allocation of a fog job J_i to either a queue Q_m or a cloud resource R_m associated with that queue Q_m as follows:

$$z_{i,m} = \begin{cases} 1, & \text{Fog job } J_i \text{ is allocated to the schedule of a cloud resource } R_m \\ 0, & \text{No allocation for job } J_i \text{ to a cloud resource } R_m \end{cases} \tag{6}$$

Since fog jobs $\boldsymbol{J}$ are submitted by different fog nodes, they come to the cloud dispatcher cd with diverse computational demands and QoS obligations. Each fog job J_i is thus stamped with a prescribed execution time $\mathcal{E}_i$ and an arrival time a_i, where $\mathcal{E}_i$ denotes the service time required by a cloud resource R_m to execute the demand of the fog job J_i, whereas a_i denotes the arrival time of fog job J_i to the cloud dispatcher cd.

Each fog job J_i waits in the cloud tier to receive service from a cloud resource R_m. The time spent by a fog job J_i in the dispatcher's queue is ignored, modeled by $c\omega^{\beta}_{i|cd}$:

$$c\omega^{\beta}_{i|cd} = 0 \tag{7}$$

However, the time spent by a fog job J_i in resource queues $\boldsymbol{Q}$ is modeled by $c\omega^{\beta}_{i}$, which is formalized according to a scheduling order β in the cloud tier. Once a fog job J_i receives a service and leaves the cloud tier, the time of departure is modeled by d_i, which in turn models a response time rt^{β}_{i} that is a function of the execution time $\mathcal{E}_i$ of fog job J_i in a cloud resource R_m, and the total waiting time $t\omega^{\beta}_{i}$ of a fog job J_i governed by a scheduling order β in the cloud–fog environment so far, as follows:

$$t\omega^{\beta}_{i} = f\omega^{\beta}_{i} + c\omega^{\beta}_{i} \tag{8}$$

$$rt^{\beta}_{i} = \mathcal{E}_i + t\omega^{\beta}_{i} \tag{9}$$

where $f\omega^{\beta}_{i}$ models the waiting time of a fog job J_i in the fog environment, and $c\omega^{\beta}_{i}$ models the waiting time of fog job J_i in the cloud computing environment. Fog jobs $\boldsymbol{J}$ are governed by various SLAs, each of which entails a job's service deadline $\mathcal{L}_i$ that in turn stipulates a target completion time $c^{(t)}_{i}$ for the fog job J_i in the cloud environment. The $c^{(t)}_{i}$ represents an explicit QoS obligation on the cloud service provider to complete the servicing of the fog job J_i, which incurs a waiting time allowance $l\omega_i$ that represents a service deadline $\mathcal{L}_i$ at the level of resource queuing $\boldsymbol{Q}$ as follows:

$$\begin{aligned} \mathcal{L}_i &= c^{(t)}_{i} - a_i \\ &= \mathcal{E}_i + l\omega_i \end{aligned} \tag{10}$$

$$J_i = \left\{ a_i, \mathcal{E}_i, c^{(t)}_{i} \right\} \tag{11}$$

For a fog job J_i that starts the service at its allocated cloud resource R_m, an SLA violation α^{β}_{i} occurs when its response time rt^{β}_{i} is higher than its pre-defined service deadline $\mathcal{L}_i$, which accordingly incurs a QoS penalty described as follows:

$$(rt^{\beta}_{i} - \mathcal{L}_i) = \begin{cases} \alpha^{\beta}_{i} > 0, & \text{The fog job } J_i \text{ is not satisfied of the cloud service} \\ \alpha^{\beta}_{i} \leq 0, & \text{The fog job } J_i \text{ is satisfied of the cloud service} \end{cases} \tag{12}$$

Utilizing such execution and service factors, a penalty cost $\mathcal{C}$ and an energy cost ϵ are accordingly formalized to model and evaluate the system performance across the cloud–fog computing environment. The penalty cost $\mathcal{C}$ and energy cost ϵ are formulated based on the communication and queue waiting $t\omega^{\beta}_{i}$ across the cloud–fog environment, the SLA violation α^{β}_{i} in the cloud environment, the service $\mathcal{E}_i$ in the cloud environment, and the bandwidth allocation Γ_i in the fog environment. Such QoS attributes are selected to measure system performance because they can be easily captured and predicted in the queuing system adopted to model the system design

6.2.1. Communication Penalty Cost for Bandwidth Allocation Γ_i in Fog Environment

Each fog job J_i demands a pre-defined bandwidth requirement denoted by Γ_i allocated to communicate data between fog nodes and the cloud tier. A cost Λ_i of bandwidth usage is incurred per data unit of a fog job J_i, modeled by an exponential distribution of bandwidth penalty mean μ_Γ as follows:

$$\Lambda_i = \exp(\mu_\Gamma) \tag{13}$$

The communication bandwidth Γ_i allocated for a fog job J_i in the fog tier is subject to an SLA that stipulates an exponential bandwidth penalty cost curve modeled by ρ_i^Γ, formulating the total penalty cost of bandwidth usage per time unit of data as follows:

$$\rho_i^\Gamma = \kappa_\Gamma * (1 - \mathrm{e}^{-\nu\ \Lambda_i\ \Gamma_i}) \tag{14}$$

where κ_Γ is a monetary cost factor for the bandwidth allocation penalty and ν is an arbitrary scaling factor.

6.2.2. QoS Penalty Cost for Queue Waiting $t\omega_i^\beta$ across the Cloud–Fog Environment

For each fog job J_i waiting in resource queues $\boldsymbol{Q}$ of the cloud tier to receive service, there exists a waiting cost denoted by ψ_i for each time unit of waiting $t\omega_i^\beta$, modeled by an exponential distribution with a waiting penalty mean μ_ω as follows:

$$\psi_i = \exp(\mu_\omega) \tag{15}$$

As explained in 8, the waiting $f\omega_i^\beta$ of a fog job J_i in the fog environment and its waiting $c\omega_i^\beta$ in the cloud environment compose a total waiting $t\omega_i^\beta$. Thus, the waiting $t\omega_i^\beta$ of a fog job J_i reaching the cloud tier is subject to an SLA that stipulates an exponential waiting penalty cost curve modeled by ρ_i^ω, formulating the penalty cost for each time unit of waiting $t\omega_i^\beta$ as follows:

$$\rho_i^\omega = \kappa_\omega * (1 - \mathrm{e}^{-\nu\ \psi_i\ (f\omega_i^\beta + c\omega_i^\beta)}) \tag{16}$$

where κ_ω is a monetary cost factor for the waiting penalty.

6.2.3. QoS Penalty Cost for Cloud Service $\mathcal{E}_i$ in the Cloud Environment

After waiting for $c\omega_i^\beta$ in the cloud tier and $t\omega_i^\beta$ in total, a fog job J_i starts the execution $\mathcal{E}_i$ in a cloud resource R_m with a service cost denoted by ξ_i per time unit of execution, which is modeled by an exponential distribution with a penalty execution mean $\mu_\mathcal{E}$ as follows:

$$\xi_i = \exp(\mu_\mathcal{E}) \tag{17}$$

The service execution $\mathcal{E}_i$ of a fog job J_i in a cloud tier is subject to an SLA that stipulates an exponential service penalty cost curve modeled by $\rho_i^\mathcal{E}$, forming the cost of servicing a fog job J_i in a cloud resource R_m as follows:

$$\rho_i^\mathcal{E} = \kappa_\mathcal{E} * (1 - \mathrm{e}^{-\nu \sum_{m=1}^{n} (\ \xi_i\ \mathcal{E}_i\ z_{i,m})}) \tag{18}$$

where $\kappa_\mathcal{E}$ is a monetary cost factor for execution penalty.

6.2.4. QoS Penalty Cost for Cloud SLA Violation α_i^β in the Cloud Environment

A violation in the SLA agreed upon with the cloud service provider is caused if a fog job J_i waits for a time longer than the waiting time allowance $l\omega_i$ prescribed in the

SLA. A time unit of SLA violation α_i^{β} incurs an SLA cost ζ_i modeled by an exponential distribution with an SLA penalty mean μ_α as follows:

$$\zeta_i = \exp(\mu_\alpha) \tag{19}$$

The service-level violation α_i^{β} of a fog job J_i in a cloud tier is subject to an SLA that stipulates an exponential penalty cost curve modeled by ρ_i^{α} as follows:

$$\rho_i^{\alpha} = \kappa_\alpha * (1 - \mathrm{e}^{-\nu \sum_{m=1}^{n} (\, \zeta_i \, \alpha_i^{\beta} \, z_{i,m})}) \tag{20}$$

where κ_α is a monetary cost factor for the SLA violation penalty.

6.3. Problem Formulation: Minimum Cost of QoS and Energy Penalty

The problem is modeled by analyzing the performance penalty cost of QoS and energy for allocating and serving a fog job J_i across the cloud–fog computing environment, represented by $\mathcal{C}$ and ϵ, respectively.

6.3.1. Penalty Cost $\mathcal{C}$ of QoS across the Cloud–Fog Environment

The total penalty cost of scheduling the stream ℓ across the cloud–fog computing environment is given by $\mathcal{C}$, which formulates the performance of:

- The communication penalty cost ρ_i^{Γ} of bandwidth Γ_i allocated to transmit a fog job J_i;
- The service penalty cost $\rho_i^{\mathcal{E}}$ to execute a time unit of $\mathcal{E}_i$ for a fog job J_i in a cloud resource R_m;
- The waiting penalty cost ρ_i^{ω} for each time unit of waiting $t\omega_i^{\beta}$ to queue a fog job J_i in resource queues $\boldsymbol{Q}$ of the cloud tier;
- The violation penalty cost ρ_i^{α} of not fulfilling SLA of a fog job J_i.

Thus, the schedule penalty cost $\mathcal{C}$ across the cloud–fog computing environment is modeled by:

$$\mathcal{C} = \sum_{i=1}^{l} (\chi_i^{\Gamma} \, \rho_i^{\Gamma} + \chi_i^{\omega} \, \rho_i^{\omega} + \chi_i^{\mathcal{E}} \, \rho_i^{\mathcal{E}} + \chi_i^{\alpha} \, \rho_i^{\alpha}) \tag{21}$$

$$\sum_i (\chi_i^{\Gamma} + \chi_i^{\omega} + \chi_i^{\mathcal{E}} + \chi_i^{\alpha}) = 1, \quad \forall i \in [1, l] \tag{22}$$

where χ_i^{Γ}, χ_i^{ω}, $\chi_i^{\mathcal{E}}$, and χ_i^{α} are scaling factors for communication, service, waiting, and SLA-violation penalty costs, respectively.

The objective is to formalize the performance penalty cost by allocating a stream ℓ of fog jobs $\boldsymbol{J}$ in the cloud tier with a scheduling order β, such that the QoS penalty cost $\mathcal{C}$ is minimized at the level of the cloud–fog computing environment, and thus the schedule performance is optimized, as follows:

$$\underset{\beta}{\text{minimize}} \; (\mathcal{C}) \equiv \underset{\beta}{\text{minimize}} \; \sum_{i=1}^{l} (\Lambda_i \, \Gamma_i + \psi_i \, t\omega_i^{\beta} + \xi_i \, \mathcal{E}_i + \zeta_i \, \alpha_i^{\beta}) \tag{23}$$

Each cloud resource R_m can only execute one fog job J_i at a time. The service execution discipline of a fog job J_i is non-preemptive; a fog job J_i cannot be interrupted once it starts the execution on a cloud resource R_m. Cloud resources $\boldsymbol{R}$ are homogeneous, and hence the cost of servicing any fog job J_i on any cloud resource R_m is similar.

6.3.2. Penalty Cost ϵ of Energy across the Cloud–Fog Environment

Scheduling the stream ℓ of fog jobs $\boldsymbol{J}$ across the cloud–fog computing environment incurs an energy cost ϵ that formulates the energy performance of bandwidth allocation Γ_i, waiting $t\omega_i^{\beta}$ in a resource queue Q_m in the cloud tier, service time $\mathcal{E}_i$ in a cloud resource R_m, and SLA violation α_i^{β} with the cloud service provider.

As such, a fog job J_i is delivered by a fog device for execution in cloud resources $\boldsymbol{R}$. Allocating a bandwidth Γ_i for a fog job J_i in the fog tier incurs an energy cost e_{γ,Γ_i} per time unit of bandwidth allocation, that is, a function of an energy consumption per bit u modeled by an exponential distribution with a rate λ_Γ and a communication bit-rate q modeled by a uniform distribution as follows:

$$e_{\gamma,\Gamma_i} = u \times q \tag{24}$$

$$u = \exp(\lambda_\Gamma) \tag{25}$$

$$q = \text{uniform}() \tag{26}$$

which, as a result, incurs a total bandwidth energy cost $E_{i,\Gamma}$ modeled by:

$$E_{i,\Gamma} = \sum_{\gamma=1}^{F} \left(\Gamma_i \times e_{\gamma,\Gamma_i} \times z_{i,\gamma} \right) \tag{27}$$

Once a fog job J_i arrives to the cloud computing environment, the cloud dispatcher cd allocates a resource R_m that fulfills the job's QoS waiting requirements with the least energy cost. There exists an energy cost e_{m,ω_i} per time unit of waiting $t\omega_i^\beta$ in a resource queue Q_m in the cloud tier modeled by an exponential distribution with a rate λ_ω, which accordingly incurs a total waiting energy cost $E_{i,\omega}$ modeled by:

$$E_{i,\omega} = \sum_{m=1}^{n} \left(t\omega_i^\beta \times e_{m,\omega_i} \times z_{i,m} \right) \tag{28}$$

$$e_{m,\omega_i} = \exp(\lambda_\omega) \tag{29}$$

When a fog job J_i is delivered from a resource queue Q_m to start the service in a cloud resource R_m, an energy cost $e_{m,\mathcal{E}_i}$ is incurred for each time unit of service $\mathcal{E}_i$ in the cloud resource R_m modeled by an exponential distribution $\lambda_\mathcal{E}$, which thus incurs a total service energy cost $E_{i,\mathcal{E}}$ modeled by:

$$E_{i,\mathcal{E}} = \sum_{m=1}^{n} \left(\mathcal{E}_i \times e_{m,\mathcal{E}_i} \times z_{i,m} \right) \tag{30}$$

$$e_{m,\mathcal{E}_i} = \exp(\lambda_\mathcal{E}) \tag{31}$$

If, however, an SLA violation occurs, an energy cost e_{m,α_i} is developed per each time unit of SLA violation α_i^β with the cloud service provider modeled by λ_α, which therefore incurs a total SLA violation energy cost $E_{i,\alpha}$ modeled by:

$$E_{i,\alpha} = \sum_{m=1}^{n} \left(\alpha_i^\beta \times e_{m,\alpha_i} \times z_{i,m} \right) \tag{32}$$

$$e_{m,\alpha_i} = \exp(\lambda_\alpha) \tag{33}$$

As such, the entire cost of energy ϵ for a stream ℓ of fog jobs $\boldsymbol{J}$ across the cloud–fog computing environment is modeled by:

$$\epsilon = \sum_{i=1}^{l} \left(E_{i,\Gamma} + E_{i,\omega} + E_{i,\mathcal{E}} + E_{i,\alpha} \right) \tag{34}$$

The objective is to formulate a schedule for a stream ℓ of fog jobs $\boldsymbol{J}$ in the cloud tier with a scheduling order β such that the entire cost of energy ϵ is minimized at the level of the cloud–fog computing environment, as follows:

$$\underset{\beta}{\text{minimize}}\ (\epsilon) = \sum_{i=1}^{l} \Bigg(\sum_{\gamma=1}^{F} (\Gamma_i \times e_{\gamma,\Gamma_i} \times z_{i,\gamma}) + \sum_{m=1}^{n} \big((t\omega_i^{\beta} \times e_{m,\omega_i} + \mathcal{E}_i \times e_{m,\mathcal{E}_i} + \alpha_i^{\beta} \times e_{m,\alpha_i}) \times z_{i,m} \big) \Bigg) \tag{35}$$

7. Evaluation

The case study conducted in this paper evaluates the efficacy of the cost-aware scheduling framework in serving heterogeneous job workloads. The cloud–fog computing environment is built in Java, in which queues are utilized to implement the cloud layer. Service demands, QoS requirements, and service cost of energy for each fog job are generated using the mathematical model proposed in this paper. The implementation of the framework is coded using a Workstation with 8GB main memory in a Core i7-8550U CPU @ 1.80 GHz 1.99 GHz.

7.1. Workload Characterizations and Design of the Cloud–Fog Computing Environment

The IoT layer consists of devices distributed throughout the fog environment, that are with various platforms and architectures. Such devices are modeled in this paper by sensors that detect, collect, and transmit data for processing in fog nodes and the cloud computing environment. Jobs delivered by such device sensors are thus heterogenous in their service demands, costs, and QoS requirements. The cloud layer consists of computing servers resided in data centers that provide on-demand services with high processing performance. A one-tier cloud layer is adopted with three servers $[R_1, R_2, R_3]$, each of which respectively entails a queue $[Q_1, Q_2, Q_3]$ to buffer fog jobs $\boldsymbol{J}$ for execution and each server follows the $M/M/1$ queuing system.

Fog nodes $\mathbb{F}$ deliver a set of ℓ atomic, independent fog jobs $\boldsymbol{J}$ to the dispatcher cd of the cloud layer. The dispatcher cd utilizes a scheduling strategy to allocates each fog job J_i on a particular cloud queue $[Q_1, Q_2, Q_3]$ for execution. Since the service demand $\mathcal{E}_i$ for each fog job J_i can be estimated beforehand using workload prediction models, the $\mathcal{E}_i$ is thus assumed to be known in advance and modeled by an exponential distribution with a service mean $\mu_{\mathcal{E}} = 1$ as follows:

$$\mathcal{E}_i = \exp(\mu_{\mathcal{E}} = 1) \tag{36}$$

7.2. Modeling for Penalty Cost $\mathcal{C}$ of QoS

The bandwidth penalty mean at the fog layer is set for $\mu_\Gamma = 1.0$, which makes the cost Λ_i of bandwidth usage per data unit of a fog job J_i to be $\Lambda_i = \exp(\mu_\Gamma = 1.0)$, according to Equation (13). At the cloud layer, the waiting penalty mean is to be $\mu_\omega = 1.0$, and the execution penalty mean is to be $\mu_{\mathcal{E}} = 1.0$, which respectively produce a waiting cost $\psi_i = \exp(\mu_\omega = 1.0)$ per time unit of waiting using Equation (15) and a service cost $\xi_i = \exp(\mu_{\mathcal{E}} = 1.0)$ per time unit of execution according to Equation (17). An SLA penalty mean is similarly set for $\mu_\alpha = 1.0$, which, as a result, incurs an SLA cost $\zeta_i = \exp(\mu_\alpha = 1.0)$ per time unit of service violation as in Equation (19).

7.3. Modeling for Penalty Cost ϵ of Energy

The energy model determines the consumption at the transmission stage in the fog layer and the waiting/execution stage in the cloud layer. The model presents that the energy consumed to execute a fog job J_i in a cloud resource R_m is higher than the energy required to hold a fog job J_i in a cloud queue Q_m waiting for execution in the cloud resource R_m. Yet, the bandwidth energy consumed at the fog layer required for transmitting a fog job J_i to be executed at the cloud layer is higher than the energy consumed to execute such a job in a cloud resource R_m.

At the fog layer, the energy consumption per bit g (measured in Joule per bit) is modeled by the rate $\lambda_\Gamma = 0.3$, and hence, $g = \exp(0.3)$, according to Equation (25). The communication bit-rate (measured in bits per second) according to Equation (26) is modeled by $q = \text{uniform}()$. As a result, the energy cost per time unit of bandwidth allocation Γ_i is modeled by $e_{\gamma,\Gamma_i} = \exp(0.3) \times \text{uniform}()$, as in Equation (27).

At the cloud layer, the energy cost per time unit of waiting $t\omega_i^\beta$ in a resource queue Q_m is modeled by Equation (29) to become $e_{m,\omega_i} = \exp(1.0)$ with the rate $\lambda_\omega = 1.0$. For a fog job J_i being serviced in a cloud resource R_m, the energy cost per time unit of service $\mathcal{E}_i$ in the cloud resource R_m is modeled using Equation (31) by $e_{m,\mathcal{E}_i} = \exp(0.2)$ with the rate $\lambda_\mathcal{E} = 0.2$. Similarly, the energy cost developed per time unit of SLA violation α_i^β with the cloud service provider is modeled by $e_{m,\alpha_i} = \exp(0.2)$ using Equation (33) with the rate $\lambda_\alpha = 0.2$. It is shown that rates of execution $\lambda_\mathcal{E}$ and SLA violation λ_α are modeled to be higher than the rate of waiting λ_ω, but to be lower than the rate of data communication λ_Γ.

7.4. The Genetic Approach

The process of scheduling fog jobs $\boldsymbol{J}$ in the cloud layer such that the QoS penalty cost $\mathcal{C}$ and the energy penalty cost ϵ are mitigated is an NP problem. The huge number of fog jobs $\boldsymbol{J}$ received at the cloud layer makes it difficult to formulate cost-optimal schedules in a timely manner. However, the permutation genetic algorithm, as a meta-heuristic search strategy, demonstrates its effectiveness in such cases [67–69]. The genetic algorithm and virtualized-queue design scheme proposed in [41,65,66] demonstrate their effectiveness in efficiently exploring and exploiting the scheduling space such that a near-optimal schedule of jobs is formed in a reasonable time, which are adopted in this paper to find a cost near-optimal schedule of fog jobs $\boldsymbol{J}$ at the cloud layer.

As such, a fitness function is formed to evaluate the quality of each virtualized queue (chromosome). The fitness value $f_{r,G}$ of a chromosome r in a generation G represents the penalty cost $\mathcal{C}$ of the QoS and the penalty cost ϵ of energy, each of which computes a normalized fitness value F_r for each schedule candidate. Accordingly, Russian Roulette is used to select a set of schedule candidates to produce the population of the next generation using crossover and mutation operators. Two fitness values are presented: $fQ_{r,G}$ to represent the fitness of the cost $\mathcal{C}$ of QoS penalty and $fE_{r,G}$ to represent the fitness of the cost ϵ of energy penalty. The *Single-Point* crossover and *Insert* mutation genetic operators are utilized to evolve the schedule of fog jobs $\boldsymbol{J}$ at the cloud layer. The rates of such operators are both set to be 0.1 of the population size in each generation. The population size is set to 10, and the maximum number of tours is set to 3000.

7.5. Discussions on Obtained Results

Findings of applying the cost-aware scheduling framework across the cloud–fog computing environment validates the performance and demonstrates the framework's effectiveness in mitigating cost $\mathcal{C}$ of the QoS penalty and cost ϵ of energy for formulated schedules at the cloud layer.

7.6. Cost $\mathcal{C}$ of QoS Penalty of Schedules

The schedules of fog jobs $\boldsymbol{J}$ at the cloud layer are formulated on cloud resources to mitigate the cost $\mathcal{C}$ of the QoS penalty. The performance optimality is measured by evaluating the quality of formulated schedules on cloud queues $\boldsymbol{Q}$. Table 2 presents the assessment of the QoS penalty cost $\mathcal{C}$ by utilizing the genetic approach, where a system state of a virtualized-queue for 30 fog jobs is evaluated.

Table 2. QoS penalty cost of schedules across the cloud–fog computing environment.

	Virtualized [1] Queue	**Initial [2]**		**Enhanced [3]**		**Improvement**	
		$\mathcal{C}$	**Penalty**	$\mathcal{C}$	**Penalty**	**Cost %**	**Penalty %**
Cloud Tier [4] [Figure 3]	30	3.3×10^6	0.963	2.06×10^6	0.865	39.3%	10.2%
Q_1 [Figure 4a]	16	1.62×10^6	0.802	1.11×10^6	0.670	31.4%	16.4%
Q_2 [Figure 4b]	9	0.84×10^6	0.569	0.42×10^6	0.342	50.2%	39.9%
Q_3 [Figure 4c]	5	0.83×10^6	0.565	0.47×10^6	0.376	43.3%	33.4%

[1] It represents the total number of jobs in each queue of the cloud tier. For instance, the first entry of the table means that 16 jobs are allocated to queue of server 1. [2] It represents the penalty cost of QoS for jobs in the virtual queue according to the their initial scheduling before using the genetic solution. [3] It represents the penalty cost of QoS for jobs in the virtual queue according to the their enhanced scheduling formulated after using the genetic solution. [4] It represents the total number of jobs in the cloud tier, the 3 queues combined together.

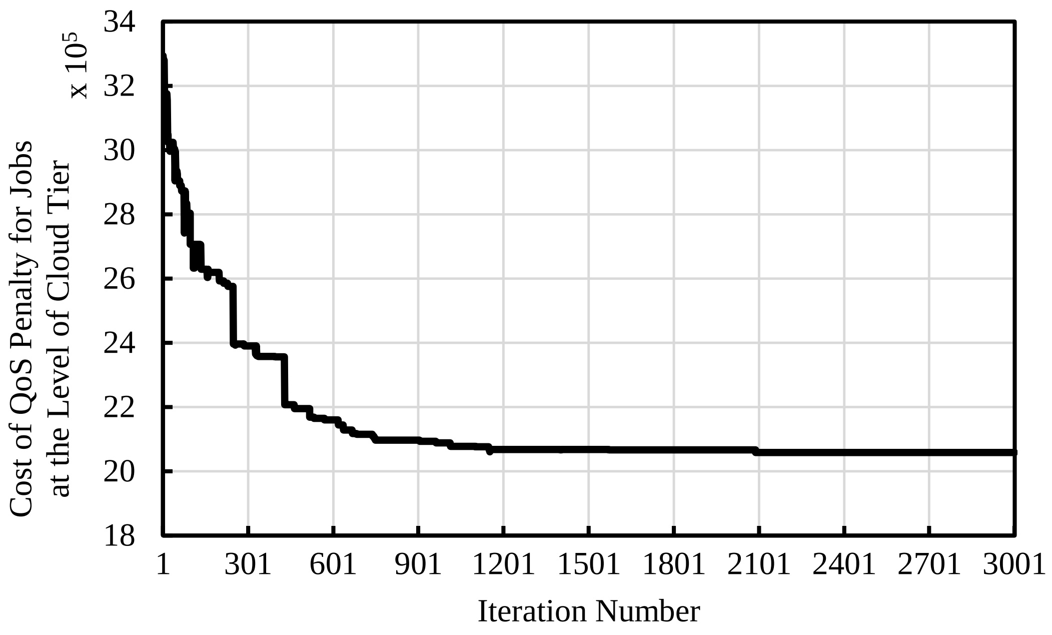

Figure 3. Cost of QoS penalty scheduling for a virtualized queue of 30 jobs across the cloud–fog computing environment.

The cost $\mathcal{C}$ of schedule in the initial state is 3.3×10^6, which results in a 0.963 QoS penalty. When the genetic algorithm is employed, the cost $\mathcal{C}$ of schedule is accordingly enhanced to a near-optimal value of 2.06×10^6 with a 0.865 QoS penalty. Thus, the cost $\mathcal{C}$ and QoS penalty of schedule are improved by 39.3% and 10.2%, respectively. Figure 3 corroborates such findings and shows the mitigation of the QoS penalty cost $\mathcal{C}$ for fog jobs J at the level of cloud layer, in which the genetic algorithm utilizes only 3000 iterations to reach a near-optimal penalty cost.

Furthermore, the QoS penalty cost $\mathcal{C}$ is calculated at the level of each resource queue at the cloud layer. For that, queue Q_1 in Table 2 entails 16 fog jobs organized in a virtualized-queue. The cost $\mathcal{C}$ of QoS in the initial system state is 1.62×10^6, which produces a 0.802 penalty. The cost $\mathcal{C}$ is enhanced to become 1.11×10^6 with a 0.67 penalty. The cost $\mathcal{C}$ and penalty are improved by 31.4% and 16.4%, respectively. Figure 4a affirms such improvements and assures the effectiveness of the genetic algorithm along with the virtualized-queue design scheme in enhancing the cost performance $\mathcal{C}$ of the QoS penalty for fog jobs of queue Q_1 in only 600 iterations.

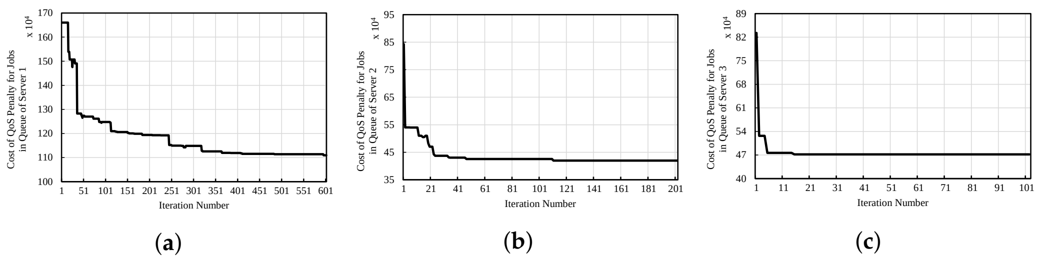

Figure 4. Cost of QoS Penalty scheduling for each server in the cloud environment. (**a**) Virtualized queue of 16 jobs. (**b**) Virtualized queue of 9 jobs. (**c**) Virtualized queue of 5 jobs.

In addition, the cost $\mathcal{C}$ and penalty of queue Q_2 in Table 2 are improved by 51.9% and 45.3%, respectively. Similarly for the virtualized queue of Q_3 with five fog jobs, the improvements are 32.7% on the schedule cost $\mathcal{C}$ and 28.1% on the penalty. The cost of the QoS penalty in Figure 4b for the virtualized queue of Q_2 and Figure 4c for the virtualized queue of Q_3 foster such findings in a reasonable time, which overall emphasizes the performance of reaching a near-optimal QoS penalty cost $\mathcal{C}$ within 200 and 100 iterations, respectively.

7.7. Cost ϵ of the Energy Penalty of Schedules

The schedule optimality is evaluated by computing the cost ϵ of the energy penalty at the cloud layer. Fog jobs $\boldsymbol{J}$ are allocated on resource queues $\boldsymbol{Q}$ of the cloud by utilizing the virtualized queue design scheme. Table 3 presents the cost ϵ of the energy penalty for job schedules at the cloud layer where an allocation of 30 fog jobs on a virtualized queue is assessed.

Table 3. Energy penalty cost of schedules across cloud–fog computing environment.

	Virtualized [1] Queue	**Initial [2]**		**Enhanced [3]**		**Improvement**	
		ϵ	**Penalty**	ϵ	**Penalty**	**Cost %**	**Penalty %**
Cloud Tier [4] [Figure 5]	30	12.46×10^6	0.712	5.53×10^6	0.425	55.6%	40.4%
Q_1 [Figure 6a]	16	2.91×10^6	0.252	1.46×10^6	0.136	49.8%	46.2%
Q_2 [Figure 6b]	9	5.26×10^6	0.409	2.53×10^6	0.224	51.9%	45.3%
Q_3 [Figure 6c]	5	4.29×10^6	0.349	2.89×10^6	0.251	32.7%	28.1%

[1] It represents the total number of jobs in each queue of the cloud tier. For instance, the first entry of the table means that 16 jobs are allocated to the queue of server 1. [2] It represents the penalty cost of energy for jobs in the virtual queue according to the their initial scheduling before using the genetic solution. [3] It represents the penalty cost of energy for jobs in the virtual queue according to the their enhanced scheduling formulated after using the genetic solution. [4] It represents the total number of jobs in the cloud tier, the 3 queues combined together.

The energy cost ϵ at the cloud layer is initially 12.46×10^6, which carries a 0.712 penalty. The genetic approach is applied along with the virtualized queue design scheme, and hence, the energy cost ϵ is enhanced to become 5.53×10^6 with a 0.425 penalty. It is shown that improvements on the cost ϵ and penalty reach 55.6% and 40.4%, respectively. Figure 5 demonstrates such conclusions and shows the efficacy of the genetic algorithm in formulating a near-optimal energy cost schedule in only 3000 iterations for a virtualized queue of 30 fog jobs.

In addition, the energy cost ϵ and penalty are measured at the level of each cloud queue. For instance, the allocation of nine fog jobs of Q_2 on a virtualized queue produces an initial energy cost ϵ of 5.26×10^6 with a 0.409 penalty, which is enhanced to 2.53×10^6 with a 0.224 penalty, as shown in Table 3. The improvements achieved on the energy cost ϵ and penalty of queue Q_2 are 51.9% and 45.3%, respectively.

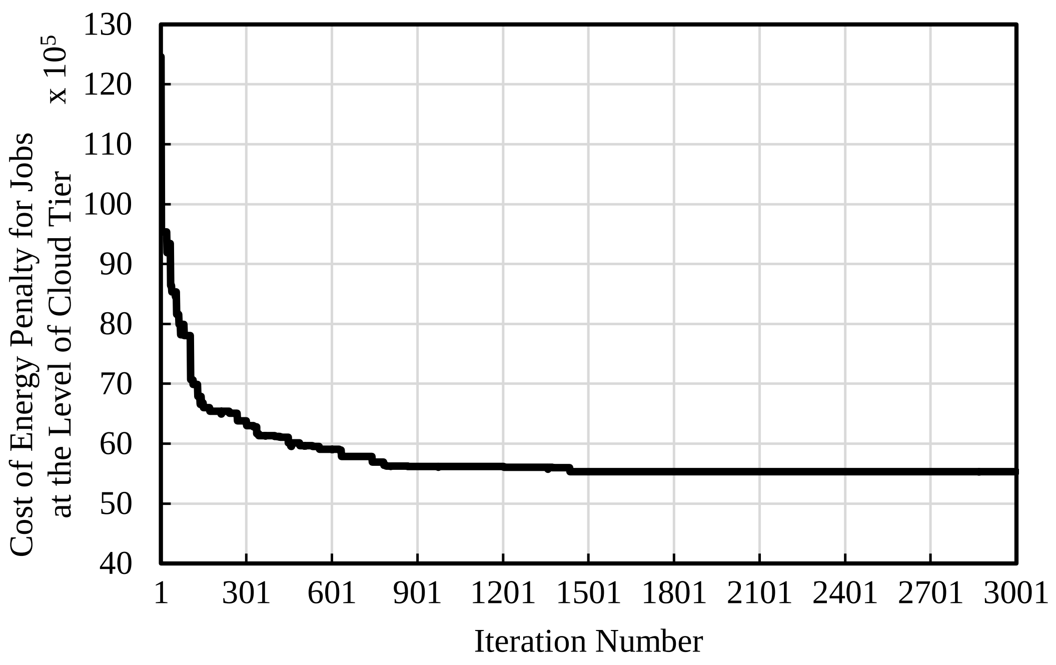

Figure 5. Cost of energy penalty scheduling for a virtualized queue of 30 jobs across the cloud–fog computing environment.

Similarly, the improvements are proven on queues Q_1 and Q_3. For queue Q_1, the energy cost ϵ is mitigated from 2.91×10^6 to 1.46×10^6 with a 49.8% enhancement, whereas the penalty is mitigated from 0.252 to 0.136 with a 46.2% reduction. For queue Q_3, reductions in the energy cost ϵ and penalty reach 32.7% and 28.1%, respectively. Figure 6a–c illustrate the mitigation of the cost ϵ of energy penalty in a reasonable time for queues Q_1, Q_2, and Q_3, respectively, wherein only 200 iterations are utilized to reach a near-optimal cost ϵ of energy penalty.

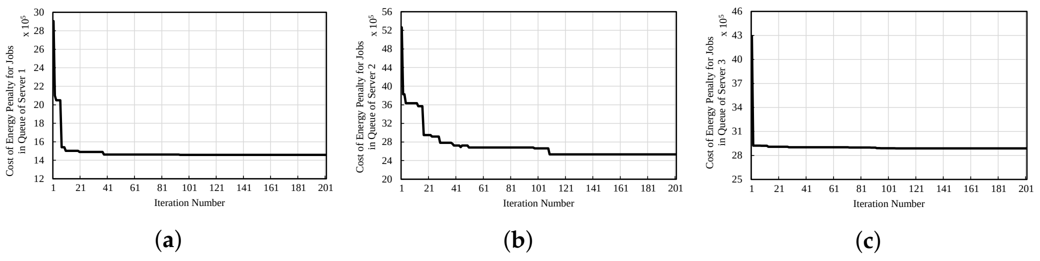

Figure 6. Cost of energy penalty scheduling for each server in the cloud environment. (**a**) Virtualized queue of 16 jobs. (**b**) Virtualized queue of 9 jobs. (**c**) Virtualized queue of 5 jobs.

8. Conclusions

The cost-aware framework demonstrates its efficacy in managing the allocation and execution of fog jobs in a cloud–fog computing environment. Scheduling and load balancing decisions are frequently triggered at run-time such that quality and service obligations of fog jobs are fulfilled. It is shown that the framework emphasizes the notion of energy-efficient scheduling based on the QoS penalty of fog jobs, in which the formulations of scheduling decisions in the framework tolerate risks of delays and energy on the cost performance.

The scheduling mechanisms employed in the framework demonstrate the effectiveness of decisions in incorporating the impacts of SLA obligations and energy incurred due to communication, service, and waiting performance metrics on cost reduction. Such decisions mitigate the cost of energy and cost of QoS penalty required to execute fog workloads, as well as cope with heterogeneity and variations in IoT workloads experienced in the cloud computing environment with considerations to SLA obligations for each fog job.

The improvement in energy cost at the level of the cloud tier has reached around 55%, which results in around a 40% enhancement in the QoS penalty. At the queuing level of the tier, the improvements in cost and penalty reach around 52% and 45%, respectively.

The genetic-based approach utilized in the framework shows a great enhancement in forming near-optimal schedules in a reasonable time, and the approach improves the cost performance of energy and QoS penalties as well. It is shown that an optimal schedule with a reduced cost of energy penalty is formulated at the queuing level of the cloud tier by utilizing only 200 genetic iterations. In addition, only 3000 genetic iterations are employed to mitigate the cost of the energy penalty at the tier level of the cloud environment. Future directions include proposing a resource allocation framework, in which the goal is to decide on an optimal set of resource configurations and setups such that QoS requirements are met. It involves proposing an SLA penalty and profit models based on workloads' heterogeneity and client demands for resources, that are to be utilized by the framework so that client satisfactions are maximized.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Kumari, N.; Yadav, A.; Jana, P. Task offloading in fog computing: A survey of algorithms and optimization techniques. *Comput. Netw.* **2022**, *214*, 109137. [CrossRef]
2. Alli, A.; Alam, M. The fog cloud of things: A survey on concepts, architecture, standards, tools, and applications. *Internet Things* **2020**, *9*, 100177. [CrossRef]
3. Aslanpour, M.; Gill, S.; Toosi, A. Performance evaluation metrics for cloud, fog and edge computing: A review, taxonomy, benchmarks and standards for future research. *Internet Things* **2020**, *12*, 100273. [CrossRef]
4. Aburukba, R.; Landolsi, T.; Omer, D. A heuristic scheduling approach for fog-cloud computing environment with stationary IoT devices. *J. Netw. Comput. Appl.* **2021**, *180*, 102994. [CrossRef]
5. Laroui, M.; Nour, B.; Moungla, H.; Cherif, M.; Afifi, H.; Guizani, M. Edge and fog computing for IoT: A survey on current research activities & future directions. *Comput. Commun.* **2021**, *180*, 210–231.
6. Gedawy, H.; Habak, K.; Harras, K.; Hamdi, M. RAMOS: A resource-aware multi-objective system for edge computing. *IEEE Trans. Mob. Comput.* **2020**, *20*, 2654–2670. [CrossRef]
7. Tong, L.; Li, Y.; Gao, W. A hierarchical edge cloud architecture for mobile computing. In Proceedings of the 35th Annual IEEE INFOCOM International Conference on Computer Communications, San Francisco, CA, USA, 10–14 April 2016; pp. 1–9.
8. Brogi, A.; Forti, S. QoS-aware deployment of IoT applications through the fog. *IEEE Internet Things J.* **2017**, *4*, 1185–1192. [CrossRef]
9. Wang, B.; Wang, C.; Song, Y.; Cao, J.; Cui, X.; Zhang, L. A survey and taxonomy on workload scheduling and resource provisioning in hybrid clouds. *Clust. Comput.* **2020**, *23*, 2809–2834. [CrossRef]
10. Malik, U.; Javed, M.; Zeadally, S.; Islam, S. Energy-Efficient Fog Computing for 6G-Enabled Massive IoT: Recent Trends and Future Opportunities. *IEEE Internet Things J.* **2022**, *9*, 14572–14594. [CrossRef]
11. Kashani, M.; Rahmani, A.; Navimipour, N. Quality of service-aware approaches in fog computing. *Int. J. Commun. Syst.* **2020**, *33*, e4340. [CrossRef]
12. Murtaza, F.; Akhunzada, A.; ul Islam, S.; Boudjadar, J.; Buyya, R. QoS-aware service provisioning in fog computing. *J. Netw. Comput. Appl.* **2020**, *165*, 102674. [CrossRef]
13. Wang, Z.; Gao, F.; Jin, X. Optimal deployment of cloudlets based on cost and latency in Internet of Things networks. *Wirel. Networks* **2020**, *26*, 6077–6093. [CrossRef]
14. Deng, R.; Lu, R.; Lai, C.; Luan, T.; Liang, H. Optimal Workload Allocation in Fog-Cloud Computing toward Balanced Delay and Power Consumption. *IEEE Internet Things J.* **2016**, *3*, 1171–1181. [CrossRef]
15. Kochovski, P.; Paśćinski, U.; Stankovski, V.; Ciglarić, M. Pareto-Optimised Fog Storage Services with Novel Service-Level Agreement Specification. *Appl. Sci.* **2022**, *12*, 3308. [CrossRef]
16. Li, J.; Gu, C.; Xiang, Y.; Li, F. Edge-cloud Computing Systems for Smart Grid: State-of-the-art, Architecture, and Applications. *J. Mod. Power Syst. Clean Energy* **2022**, *10*, 805–817. [CrossRef]

17. Akram, J.; Tahir, A.; Munawar, H.; Akram, A.; Kouzani, A.; Mahmud, M. Cloud-and Fog-Integrated Smart Grid Model for Efficient Resource Utilisation. *Sensors* **2021**, *21*, 7846. [CrossRef]
18. Nasr, A.; El-Bahnasawy, N.; Attiya, G.; El-Sayed, A. Cost-effective algorithm for workflow scheduling in cloud computing under deadline constraint. *Arab. J. Sci. Eng.* **2019**, *44*, 3765–3780. [CrossRef]
19. Alahmadi, A.; Che, D.; Khaleel, M.; Zhu, M.; Ghodous, P. An Innovative Energy-Aware Cloud Task Scheduling Framework. In Proceedings of the IEEE 8th International Conference on Cloud Computing, New York, NY, USA, 27 June–2 July 2015; pp. 493–500.
20. Liu, X.; Liu, P.; Li, H.; Li, Z.; Zou, C.; Zhou, H.; Yan, X.; Xia, R. Energy-Aware Task Scheduling Strategies with QoS Constraint for Green Computing in Cloud Data Centers. In Proceedings of the Conference on Research in Adaptive and Convergent Systems, Honolulu, HI, USA, 9–12 October 2018; pp. 260–267.
21. Ben-Allah, S.; Ben-Allah, H.; Touhafi, A.; Ezzati, A. An Efficient Energy-Aware Tasks Scheduling with Deadline-Constrained in Cloud Computing. *Computers* **2019**, *8*, 46. [CrossRef]
22. Mebrek, A.; Merghem-Boulahia, L.; Esseghir, M. Efficient green solution for a balanced energy consumption and delay in the IoT-Fog-Cloud computing. In Proceedings of the IEEE 16th International Symposium on Network Computing and Applications (NCA), Cambridge, MA, USA, 30 October–1 November 2017; pp. 1–4.
23. Mebrek, A.; Merghem-Boulahia, L.; Esseghir, M. Energy-efficient solution using stochastic approach for IoT-Fog-Cloud Computing. In Proceedings of the International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), Barcelona, Spain, 21–23 October 2019; pp. 1–6.
24. Bui, D.M.; Yoon, Y.; Huh, E.N.; Jun, S.; Lee, S. Energy efficiency for cloud computing system based on predictive optimization. *J. Parallel Distrib. Comput.* **2017**, *102*, 103–114. [CrossRef]
25. Li, C.; Tang, J.; Luo, Y. Service Cost-Based Resource Optimization and Load Balancing for Edge and Cloud Environment. *Knowl. Inf. Syst.* **2020**, *62*, 4255–4275. [CrossRef]
26. Baek, B.; Lee, J.; Peng, Y.; Park, S. Three Dynamic Pricing Schemes for Resource Allocation of Edge Computing for IoT Environment. *IEEE Internet Things J.* **2020**, *7*, 4292–4303. [CrossRef]
27. Klusáček, D.; Parák, B.; Podolníková, G.; Ürge, A. Scheduling Scientific Workloads in Private Cloud: Problems and Approaches. In Proceedings of the 10th International Conference on Utility and Cloud Computing, Austin, TX, USA, 5–8 December 2017; pp. 9–18.
28. Panda, S.; Jana, P. An Energy-Efficient Task Scheduling Algorithm for Heterogeneous Cloud Computing Systems. *Clust. Comput.* **2019**, *22*, 509–527. [CrossRef]
29. Borgetto, D.; Maurer, M.; Da-Costa, G.; Pierson, J.M.; Brandic, I. Energy-efficient and SLA-aware management of IaaS clouds. In Proceedings of the 3rd IEEE International Conference on Future Systems: Where Energy, Computing and Communication Meet (e-Energy), Madrid, Spain, 9–11 May 2012; pp. 1–10.
30. Goyal, S.; Bhushan, S.; Kumar, Y.; Rana, A.u.H.S.; Bhutta, M.R.; Ijaz, M.F.; Son, Y. An Optimized Framework for Energy-Resource Allocation in a Cloud Environment based on the Whale Optimization Algorithm. *Sensors* **2021**, *21*, 1583. [CrossRef]
31. Saraswat, S.; Gupta, H.P.; Dutta, T. Fog based energy efficient ubiquitous systems. In Proceedings of the 10th International Conference on Communication Systems & Networks (COMSNETS), Bengaluru, India, 3–7 January 2018; pp. 439–442.
32. Oma, R.; Nakamura, S.; Enokido, T.; Takizawa, M. An Energy-Efficient Model of Fog and Device Nodes in IoT. In Proceedings of the 32nd International Conference on Advanced Information Networking and Applications Workshops (WAINA), Krakow, Poland, 16–18 May 2018; pp. 301–306.
33. Zhao, H.; Qi, G.; Wang, Q.; Wang, J.; Yang, P.; Qiao, L. Energy-Efficient Task Scheduling for Heterogeneous Cloud Computing Systems. In Proceedings of the IEEE 21st International Conference on High Performance Computing and Communications, Zhangjiajie, China, 10–12 August 2019; pp. 952–959.
34. Matrouk, K.; Alatoun, K. Scheduling Algorithms in Fog Computing: A Survey. *Int. J. Netw. Distrib. Comput.* **2021**, *9*, 59–74. [CrossRef]
35. Campeanu, G. A mapping study on microservice architectures of Internet of Things and cloud computing solutions. In Proceedings of the 7th Mediterranean Conference on Embedded Computing (MECO), Budva, Montenegro, 10–14 June 2018; pp. 1–4.
36. Narayana, P.; Parvataneni, P.; Keerthi, K. A Research on Various Scheduling Strategies in Fog Computing Environment. In Proceedings of the International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE), Vellore, India, 24–25 February 2020; pp. 1–6.
37. Arunarani, A.; Manjula, D.; Sugumaran, V. Task scheduling techniques in cloud computing: A literature survey. *Future Gener. Comput. Syst.* **2019**, *91*, 407–415. [CrossRef]
38. Jeon, H.; Prabhu, V. Modeling Green Fabs—A Queuing Theory Approach for Evaluating Energy Performance. In Proceedings of the Advances in Production Management Systems. Competitive Manufacturing for Innovative Products and Services, Rhodes, Greece, 24–26 September 2013; pp. 41–48.
39. Madni, S.; Latiff, M.; Coulibaly, Y.; Abdulhamid, S. Recent Advancements in Resource Allocation Techniques for Cloud Computing Environment: A Systematic Review. *Clust. Comput.* **2017**, *20*, 2489–2533. [CrossRef]
40. Atiewi, S.; Yussof, S.; Ezanee, M.; Almiani, M. A review energy-efficient task scheduling algorithms in cloud computing. In Proceedings of the IEEE Long Island Systems, Applications and Technology Conference (LISAT), Farmingdale, NY, USA, 29 April 2016; pp. 1–6.

41. Suleiman, H.; Basir, O. SLA-Driven Load Scheduling in Multi-Tier Cloud Computing: Financial Impact Considerations. *Int. J. Cloud Comput. Serv. Archit.* **2020**, *10*, 1–24. [CrossRef]
42. Yang, Y.; Wang, K.; Zhang, G.; Chen, X.; Luo, X.; Zhou, M.T. MEETS: Maximal Energy Efficient Task Scheduling in Homogeneous Fog Networks. *IEEE Internet Things J.* **2018**, *5*, 4076–4087. [CrossRef]
43. Suleiman, H.; Hamdan, M. Adaptive Probabilistic Model for Energy-Efficient Distance-based Clustering in WSNs (Adapt-P): A LEACH-Based Analytical Study. *J. Wirel. Mob. Netw. Ubiquitous Comput. Dependable Appl. JoWUA* **2021**, *12*, 65–86.
44. Dong, Z.; Liu, N.; Rojas-Cessa, R. Greedy scheduling of tasks with time constraints for energy-efficient cloud-computing data centers. *J. Cloud Comput.* **2015**, *4*, 1–14. [CrossRef]
45. Tadakamalla, U.; Menascé, D. Autonomic resource management using analytic models for fog/cloud computing. In Proceedings of the IEEE International Conference on Fog Computing (ICFC), Prague, Czech Republic, 24–26 June 2019; pp. 69–79.
46. Hoang, D.; Dang, T. FBRC: Optimization of task Scheduling in Fog-Based Region and Cloud. In Proceedings of the IEEE Trustcom/BigDataSE/ICESS, Sydney, Australia, 1–4 August 2017.
47. Tsai, J.F.; Huang, C.H.; Lin, M.H. An optimal task assignment strategy in cloud–fog computing environment. *Appl. Sci.* **2021**, *11*, 1909. [CrossRef]
48. Guo, M.; Guan, Q.; Ke, W. Optimal Scheduling of VMs in Queueing Cloud Computing Systems with a Heterogeneous Workload. *IEEE Access* **2018**, *6*, 15178–15191. [CrossRef]
49. dos Anjos, J.; Gross, J.; Matteussi, K.; González, G.; Leithardt, V.; Geyer, C. An Algorithm to Minimize Energy Consumption and Elapsed Time for IoT Workloads in a Hybrid Architecture. *Sensors* **2021**, *21*, 2914. [CrossRef]
50. Alamro, S.; Xu, M.; Lan, T.; Subramaniam, S. Shed+: Optimal Dynamic Speculation to Meet Application Deadlines in Cloud. *IEEE Trans. Netw. Serv. Manag.* **2020**, *17*, 1515–1526. [CrossRef]
51. Perret, Q.; Charlemagne, G.; Sotiriadis, S.; Bessis, N. A Deadline Scheduler for Jobs in Distributed Systems. In Proceedings of the 27th International Conference on Advanced Information Networking and Applications Workshops, Barcelona, Spain, 25–28 March 2013; pp. 757–764.
52. Sharma, S.; Saini, H. A novel four-tier architecture for delay aware scheduling and load balancing in fog environment. *Sustain. Comput. Inform. Syst.* **2019**, *24*, 100355. [CrossRef]
53. Wu, H.Y.; Lee, C.R. Energy efficient scheduling for heterogeneous fog computing architectures. In Proceedings of the IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC), Tokyo, Japan, 23–27 July 2018; Volume 1, pp. 555–560.
54. Xue, S.; Zhang, Y.; Xu, X.; Xing, G.; Xiang, H.; Ji, S. QET: A QoS-Based Energy-Aware Task Scheduling Method in Cloud Environment. *Clust. Comput.* **2017**, *20*, 3199–3212. [CrossRef]
55. Nguyen, T.; Doan, K.; Nguyen, G.; Nguyen, B.M. Modeling Multi-Constrained Fog-Cloud Environment for Task Scheduling Problem. In Proceedings of the IEEE 19th International Symposium on Network Computing and Applications (NCA), Cambridge, MA, USA, 24–27 November 2020; pp. 1–10.
56. Ben-Alla, H.; Ben-Alla, S.; Touhafi, A.; Ezzati, A. A novel task scheduling approach based on dynamic queues and hybrid meta-heuristic algorithms for cloud computing environment. *Clust. Comput.* **2018**, *21*, 1797–1820. [CrossRef]
57. Arora, N.; Banyal, R.K. Performance Analysis of Different Task Scheduling Algorithms in Cloud Computing under Dynamic Environment. In Proceedings of the International Communication Engineering and Cloud Computing Conference, Prague, Czech Republic, 8–10 October 2019; pp. 1–5.
58. Salido, M.; Escamilla, J.; Giret, A.; Barber, F. A genetic algorithm for energy-efficiency in job-shop scheduling. *Int. J. Adv. Manuf. Technol.* **2016**, *85*, 1303–1314. [CrossRef]
59. Zhang, R.; Chiong, R. Solving the energy-efficient job shop scheduling problem: A multi-objective genetic algorithm with enhanced local search for minimizing the total weighted tardiness and total energy consumption. *J. Clean. Prod.* **2016**, *112*, 3361–3375. [CrossRef]
60. Lin, J.; Cui, D.; Peng, Z.; Li, Q.; He, J. A Two-Stage Framework for the Multi-User Multi-Data Center Job Scheduling and Resource Allocation. *IEEE Access* **2020**, *8*, 197863–197874. [CrossRef]
61. Cui, D.; Peng, Z.; Xiong, J.; Xu, B.; Lin, W. A Reinforcement Learning-Based Mixed Job Scheduler Scheme for Grid or IaaS Cloud. *IEEE Trans. Cloud Comput.* **2020**, *8*, 1030–1039. [CrossRef]
62. Zhang, C.; Wang, Y.; Wu, H.; Guo, H. An Energy-Aware Host Resource Management Framework for Two-Tier Virtualized Cloud Data Centers. *IEEE Access* **2021**, *9*, 3526–3544. [CrossRef]
63. Zhao, X.; Guo, X.; Zhang, Y.; Li, W. A Parallel-Batch Multi-Objective Job Scheduling Algorithm in Edge Computing. In Proceedings of the IEEE International Conference on Internet of Things (iThings), Halifax, NS, Canada, 30 July–3 August 2018; pp. 510–516.
64. Paul, D.; Zhong, W.D.; Bose, S.K. Energy efficient scheduling in data centers. In Proceedings of the IEEE International Conference on Communications (ICC), London, UK, 8–12 June 2015; pp. 5948–5953.
65. Suleiman, H.; Basir, O. Service Level Driven Job Scheduling in Multi-Tier Cloud Computing: A Biologically Inspired Approach. In Proceedings of the International Conference on Cloud Computing: Services and Architecture, Toronto, ON, Canada, 13–14 July 2019; pp. 99–118.
66. Suleiman, H.; Basir, O. QoS-Driven Job Scheduling: Multi-Tier Dependency Considerations. In Proceedings of the International Conference on Cloud Computing: Services and Architecture, Toronto, ON, Canada, 13–14 July 2019; pp. 133–155.

67. Li, X.; Gao, L. An effective hybrid genetic algorithm and tabu search for flexible job shop scheduling problem. *Int. J. Prod. Econ.* **2016**, *174*, 93–110. [CrossRef]
68. Yang, X.; Zeng, J.; Liang, J.; Liang, J. A Genetic Algorithm for Job Shop Scheduling Problem Using Co-Evolution and Competition Mechanism. In Proceedings of the International Conference on Artificial Intelligence and Computational Intelligence, Sanya, China, 23–24 October 2010; pp. 133–136.
69. Nouiri, M.; Bekrar, A.; Jemai, A.; Niar, S.; Ammari, A. An effective and distributed particle swarm optimization algorithm for flexible job-shop scheduling problem. *J. Intell. Manuf.* **2018**, *29*, 603–615. [CrossRef]