Hindawi
Security and Communication Networks
Volume 2021, Article ID 6688162, 16 pages
https://doi.org/10.1155/2021/6688162

Research Article

DAWM: Cost-Aware Asset Claim Analysis Approach on Big Data Analytic Computation Model for Cloud Data Centre

M. S. Mekala,[1,2] **Rizwan Patan**,[3] **SK Hafizul Islam**,[4] **Debabrata Samanta**,[5] **Ghulam Ali Mallah**,[6] **and Shehzad Ashraf Chaudhry**[7]

[1]*Department of Information and Communication Engineering, Yeungnam University, Gyeongsan 38544, Republic of Korea*
[2]*RLRC for Autonomous Vehicle Parts and Materials Innovation, Yeungnam University, Gyeongsan 38544, Republic of Korea*
[3]*Department of Computer Science and Engineering, Velagapudi Ramakrishna Siddhartha Engineering College, Vijayawada, Andhra Pradesh, India*
[4]*Department of Computer Science and Engineering, Indian Institute of Information Technology, Kalyani, West Bengal 741235, India*
[5]*Department of Computer Science, CHRIST University, Bengaluru 560029, India*
[6]*Department of Computer Science, Shah Abdul Latif University, Khairpur, Sindh 66020, Pakistan*
[7]*Department of Computer Engineering, Faculty of Engineering and Architecture, Istanbul Gelisim University, Avcılar, Istanbul 34310, Turkey*

Correspondence should be addressed to Shehzad Ashraf Chaudhry; sashraf@gelisim.edu.tr

Received 25 December 2020; Revised 23 February 2021; Accepted 8 May 2021; Published 29 May 2021

Academic Editor: Zhiyuan Tan

The heterogeneous resource-required application tasks increase the cloud service provider (CSP) energy cost and revenue by providing demand resources. Enhancing CSP profit and preserving energy cost is a challenging task. Most of the existing approaches consider task deadline violation rate rather than performance cost and server size ratio during profit estimation, which impacts CSP revenue and causes high service cost. To address this issue, we develop two algorithms for profit maximization and adequate service reliability. First, a belief propagation-influenced cost-aware asset scheduling approach is derived based on the data analytic weight measurement (DAWM) model for effective performance and server size optimization. Second, the multiobjective heuristic user service demand (MHUSD) approach is formulated based on the CPS profit estimation model and the user service demand (USD) model with dynamic acyclic graph (DAG) phenomena for adequate service reliability. The DAWM model classifies prominent servers to preserve the server resource usage and cost during an effective resource slicing process by considering each machine execution factor (remaining energy, energy and service cost, workload execution rate, service deadline violation rate, cloud server configuration (CSC), service requirement rate, and service level agreement violation (SLAV) penalty rate). The MHUSD algorithm measures the user demand service rate and cost based on the USD and CSP profit estimation models by considering service demand weight, tenant cost, and energy cost. The simulation results show that the proposed system has accomplished the average revenue gain of 35%, cost of 51%, and profit of 39% than the state-of-the-art approaches.

1. Introduction

Nowadays, cloud computing has become a backbone for government enterprises and education sectors because of providing continuous resource (memory, CPU, and bandwidth) allocation service to ensure their application service reliability. The cloud service supplier shares the resources among end-users based on cost function's value (CF) to meet the demand of system performance. Many service suppliers estimate the server cost based on bandwidth usage rate (BUR) and energy usage rate (EUR). As per the Gartner report, the cloud service provider (CSP) market would grow approximately 331.2 billion dollars in 2022 [1]. The cloud global report [2] confines 623.3-billion-dollar market growth rate in 2023 for data computation. The statistical analysis states that cloud computing has a notable impact on

the Internet of Things (IoT), blockchain, and soft computing measurement systems with artificial intelligence models.The tasks are divided into subtasks with relative attribute definitions through DAG theory. The DAG approach shows a prominent impact while dealing with complex workflow applications such as systematic mathematical applications [3–5]. Data analytic languages such as Hive and Pig [6–8] platforms handle the MapReduce model queries. Thus, the DAG theory's importance tremendously changed over the past decade since it influences the service execution time and resource usage. Therefore, this issue is formulated as NP-hard [9], and many heuristic approaches resolved the same issue through resource usage consolidation [10–12].

Each machine enables a list of resource attributes (e.g., CPU, RAM size, and hard disc space) provided by CSP. In our solution, the cloud resource cost is optimized by estimating user service demands (such as CPU, IOPS, memory, and storage). For instance, an online incremental learning method has been designed in [13–15] to estimate service completion time based on heuristic algorithms by allocating the arrived service requests to the correct VM. However, these approaches have not considered server size and machine resource usage rates which causes performance delay. Therefore, in our approach, we consider CSC size, effective resource management of machines, and resource autoscaling methods; these are not present in state-of-the-art approaches. Several examinations were carried out for designing effective resource allocation methods to reduce allocation cost by satisfying service request requirements. Most current studies [16] have not considered the pricing models and data analysis models; some on-demand pricing models are considered with an inadequate measurement index. Several recent studies [17] recognize the importance of both on-demand data analytical models and reserved pricing models to minimize resource allocation costs. However, our solution assesses the server resource capacity rate, profit, and cost based on the data analysis model. The user service demand measurement algorithm is essential for profit maximization by autoscaling the resource allocation certainty.

Our research work aim is to design a novel profit optimization model for CSPs to enhance their revenue maximization (RM) by maintaining reliable quality of service (QoS). The profit optimization model must impact active server count, cost, and speed to meet the end-user satisfaction, influencing their service continuity. If there is no precise profit optimization model, then the profit and service quality and revenue generation factors will be affected. However, CSP revenue maximization has become a billion-dollar question in the competitive service computing market because of heterogeneous resource-required application tasks.

To address the listed issues, we develop two algorithms for profit maximization and adequate service reliability. First, a belief propagation-influenced cost-aware asset scheduling approach is derived based on the data analytic weight measurement (DAWM) model for effective performance and server size optimization. Second, the multiobjective heuristic user service demand (MHUSD) approach is formulated based on the CPS profit estimation model and the user service demand (USD) model with dynamic acyclic graph (DAG) phenomena for adequate service reliability. The DAWM model classifies prominent servers to preserve the server resource usage and cost during an effective resource slicing process by considering each machine execution factor (remaining energy, energy and service cost, workload execution rate, service deadline violation rate, cloud server configuration (CSC), service requirement rate, and service level agreement violation (SLAV) penalty rate). The MHUSD algorithm measures the user demand service rate and cost based on the USD and CSP profit estimation models by considering service demand weight, service tenant cost, and machine energy cost.

1.1. Key Contributions. The trade-off between cost optimization and revenue maximization models is extensively examined in Section 2. Our manuscript's key contributions are summarized as follows:

(1) Develop a data analytic weight measurement (DAWM) approach to optimize service quality and price of CSP during an effective resource slicing process by considering each machine cost and revenue, and profit.

(2) Develop a multiobjective heuristic user service demand (MHUSD) based on the CPS profit estimation model and the user service demand (USD) model to measure the user demand service rate cost by considering service demand weight, service tenant cost, and machine energy cost. Subsequently, the MHUSD algorithm also considers maximum baring wait-time of end-user to maximize CSP revenue and optimize operational energy cost.

(3) Simulation results confirm the advantage of the proposed approaches, enhancement rate of revenue, and the CSP's profit attributes. The impacts of mathematical key factors are being analyzed theoretically and practically.

The manuscript's respite is designed as Section 2 briefly explains research gaps and problem statements of extant approaches. Section 3 describes the proposed system and its mathematical models with an algorithm in detail. Section 4 evaluates the investigation outcomes, and Section 5 concludes the manuscripts.

2. Related Work

This section describes the examination of related research work, which is classified into 3 steps, such as profit maximization, green data center, and graph theory-based task consolidation approaches.

2.1. Profit Maximization. Several profit maximization methods are proposed for the sustainability of green computing. We can observe the current scenario and requirement analysis of revenue in Figure 1. In [18], the broker management system has been designed to maximize the VM cost and minimize user cost. The author formulates

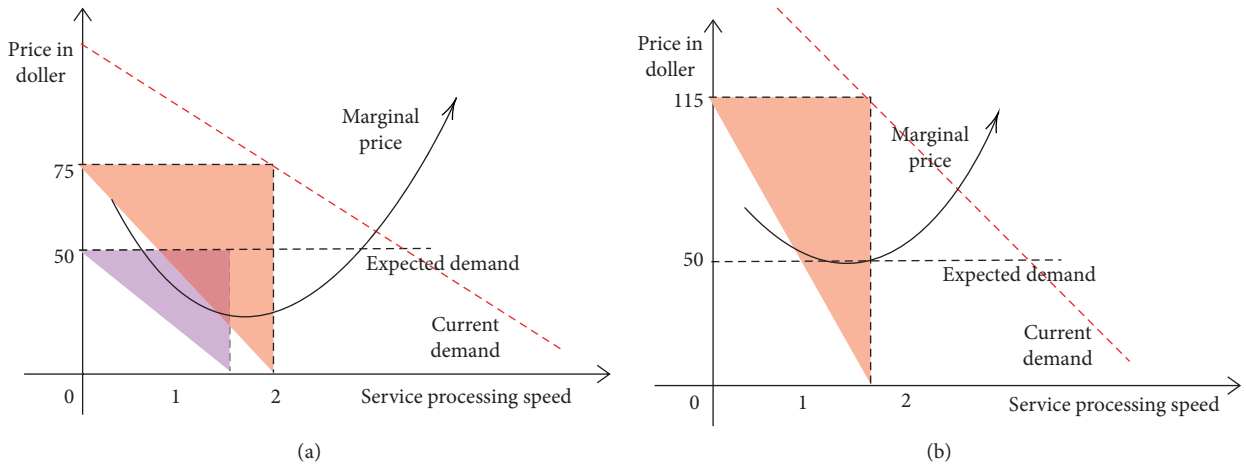

Figure 1: Formulation of revenue maximization. (a) Haphazard cost impact. (b) Expected cost importance.

multiserver configuration cost as a profit maximization issue, and a heuristic method has been designed to solve this issue. The delay-sensitive workload dimensionality has been examined based on a novel online heuristic approach to optimize the system's cost and profit [19]. Subsequently, the offline issue is formulated as NP-hard, and it has been resolved by a linear programming concept. In [20], a dynamic cost charging method has been designed to fix specific prices to servers as per the resource demand. A pricing approach has been designed to regulate the prices dynamically as per the demand of a kind. In [21, 22], the service penalty has diminished and enhances the profit by VM replacement approach through a mixed-integer nonlinear program called NP-hard; subsequently, a novel heuristic method has been designed to optimize the penalties and profits.

CPS profit maximization approaches have been extensively examined in this literature survey. In [23], the authors designed a stochastic programming scheme for the subscription of computing resources to maximize service providers' profit during user request uncertainty. In [24], a profit control policy has been designed to assess machine computing capacity, which decides to maximize the service provider profit. In [25–27], an SLA-based resource allocation issue has formulated with profit maximization objective with the consideration of 3 dimensions (processing, storage, and communication). In [28], a service request (SR) distribution approach is designed to enhance the profit with quality of service rate as per the service demand. In [29], the author has addressed the service provider revenue maximization issue by consolidating the service tenant cost and power consumption cost. A joint optimization scheduling model has been designed to manage delay-tolerant batch services based on pricing decisions to maximize service provider revenue [30]. In [31], the authors designed a model to maximize the service provider revenue based on the machine's tenant cost, resource demand size, and the application workload. A suitable online algorithm has been designed for the geo-distributed cloud with an adaptive VM resource cost scheme to maximize the service provider revenue [32]. The relationship between load balance, revenue, and the cost has concentrated on maximizing the service provider revenue than state-of-the-art approaches [33]. In [34, 35], a virtual resource rental strategy has been designed based on tenant cost, task urgency, and task uncertainty to enhance provider profit.

A hill-climbing algorithm has been designed to estimate customer service satisfaction by analyzing demand mark and profit fluctuations [36]. It assesses the customer satisfaction from economic growth ratio by leveraging the cloud server configuration (CSC), task arrival rate, and profit up-downs. Therefore, the CSC directly impacts the cloud user service satisfaction rate and the inadequate customer satisfaction also has a direct impact on service request arrival rate. However, there is a lack of an accurate decision-making system and data analysis system that affects the server's profit and performance cost. A profit estimation model has been designed by considering CSC, service requirement rate, SLA, SLAV penalty rate, energy cost, tenant cost, and current CSP margin profit [37]. A server task execution speed-based power usage model is also designed to assess the CSP profit.

2.2. Green Data Centre. In [38], a mixed-integer linear program has been designed for resource allocation to optimize the data center cost and energy consumption. Green computing accomplishes the proficient process and usage of assets by limiting the vitality utilization. An enhanced ant colony approach for optimal VM execution has been developed to enhance vitality utilization and to optimize the cost of cloud environment [39–42]. The practical swarm optimization (PSO) approach resolves the task allocation issue by consolidating data center count and task demand. In distributed computing, the assets have to schedule effectively to achieve a high-performance rate. Accordingly, the multitarget PSO approach remains preferable to enhance the resource usage rates. Therefore, this approach effectively increases the usage of assets and lessens energy and makespan. The outcomes delineated that the proposed strategy multiobjective practical swarm optimization

(MOPSO) performance is quite beneficial than concerned existing models. A VM scheduling approach has been designed based on multidimensional resource imperatives, for example, link capacity, to diminish the quantities of dynamic PMs to preserve energy utilization. The 2-step heuristic approach resolves the VM scheduling through migration and VM positioning models [43, 44]. The designed method has consolidated the execution time than extant systems in a simulation platform. Asset overburdening is still an issue, and live relocation does not uphold the change of VM performance. In [45], the energy-aware asset allocation approach has been investigated to improve the energy productivity of a server farm without SLA negotiations. An asset scheduling strategy with a hereditary method has been proposed to improve the usage of assets and save the expense of energy in distributed computing [46, 47]. It utilizes a migration approach dependent on 3 load degrees (CPU usage, the throughput of organization, and pace of circle I/O). The calculation succeeds in improving the usage of assets, and saving energy by run-time asset scheduling is high. An energy preservation system is classified by assorting the asset into four distinct classifications (CPU, memory, storage, and networks). Additionally, the author designed a unique asset scheduling system dependent on cloud assets' energy streamlining with assessment technique [48]. The study [49] evaluates every machine's fitness value, which helps assess the machine rank based on the performance and resource usage rate. However, the machine rank evolution process consumes more time which influences the performance, and task scheduling policy leads to high-performance cost. The complexity rate is high over large-scale frameworks.

2.3. Graph Theory-Based Resource/Task Scheduling. Dynamic acyclic graph (DAG) has been used for task scheduling by considering PM capacity and task resource weight to formulate the issue [50]. Here, $X[i, j]$ matrix identifies the errand evolution time of all VMs under different instances. To address all these issues, we design a data analytic weight measurement (DAWM) approach to optimize a cloud service provider's quality and price during an effective resource slicing process by considering each machine's cost and revenue, and profit. The entire cost does not iteratively consider traditional DAG-based models during the measurement of data analysis. Subsequently, we design a multiobjective heuristic user service demand (MHUSD) algorithm based on the CPS profit estimation model and the user service demand (USD) model to measure the user demand service rate and cost by considering service demand weight, service tenant cost, and machine energy cost.

3. DAWM System Model

A belief propagation-influenced data analysis model is designed for CSP profit maximization by formulating DAG task and resource scheduling policy, as shown in Figure 2. The CSP receives a service request from the cloud user, and by default, the CSP has three service modes: on-demand, advanced reservation, and spot resource allocation, which helps to slice the resources as per resource demand. As per the received service request, the CSP assesses its demand, cost, performance, profit, and required server size factors. The CSP consolidates the overprovisioning machines by optimizing the service execution cost and machine asset usage. Cloud service suppliers drive the data utility analytic method on machines to classify the high- and low-resource usage rate machines, preserve CDC usage and performance cost, and avoid instant repudiations/migrations.

It classifies adaptive servers after the first iteration by concocting an exact data analytic weight measurement (DAWM) model. First, a belief propagation influences a cost-aware asset scheduling approach based on the data analytic weight measurement (DAWM) model, which effectively optimizes the performance cost and server size. The DAWM model classifies prominent servers to preserve the server resource usage and cost during an effective resource slicing process by considering each machine execution factor (remaining energy, energy and service price, workload execution rate, service deadline violation rate, cloud server configuration (CSC), service requirement rate, and service level agreement violation (SLAV) penalty rate). Second, the multiobjective heuristic user service demand (MHUSD) approach is processed based on the CPS profit estimation model and the user service demand (USD) model with dynamic acyclic graph (DAG) phenomena for adequate service reliability. The MHUSD algorithm prognosticates the user demand service rate and cost based on the USD and CSP profit estimation models by considering service demand weight, service tenant cost, and machine energy cost. The USD model estimates the resource service demand to estimate the profit and revenue gain and the system's performance cost. The CSP profit estimation model helps assess the service profit by forecasting the server's performance cost, energy usage, and resource tenant cost. Each subsection describes a subcomponent of the framework mathematically and theoretically.

3.1. Cloud Service Provider Model. The CSP offers various services to cloud end-users. For instance, infrastructure is a service, where the resources are being offered as VMs to meet the end-user satisfaction by running their applications. The user service request (USR) is submitted to the service provider, which runs on a multiserver system to deliver the response for the received service requests. Consider a multiserver system (MSS) enables N homogeneous servers with m speed, and these are modeled based on the $(M/M/M)$ queuing system. Assume that the MSS framework receives a number of user service requests with a rate of u. The service time $v = (x/m)$, where x refers to required instruction count to execute the USR and mean $\overline{v} = (\overline{x}/m)$. The service rate of the USR is denoted as $q = (1/\overline{v}) = (m/\overline{x})$. The server utilization rate is estimated with equation (1), and it is denoted with Z:

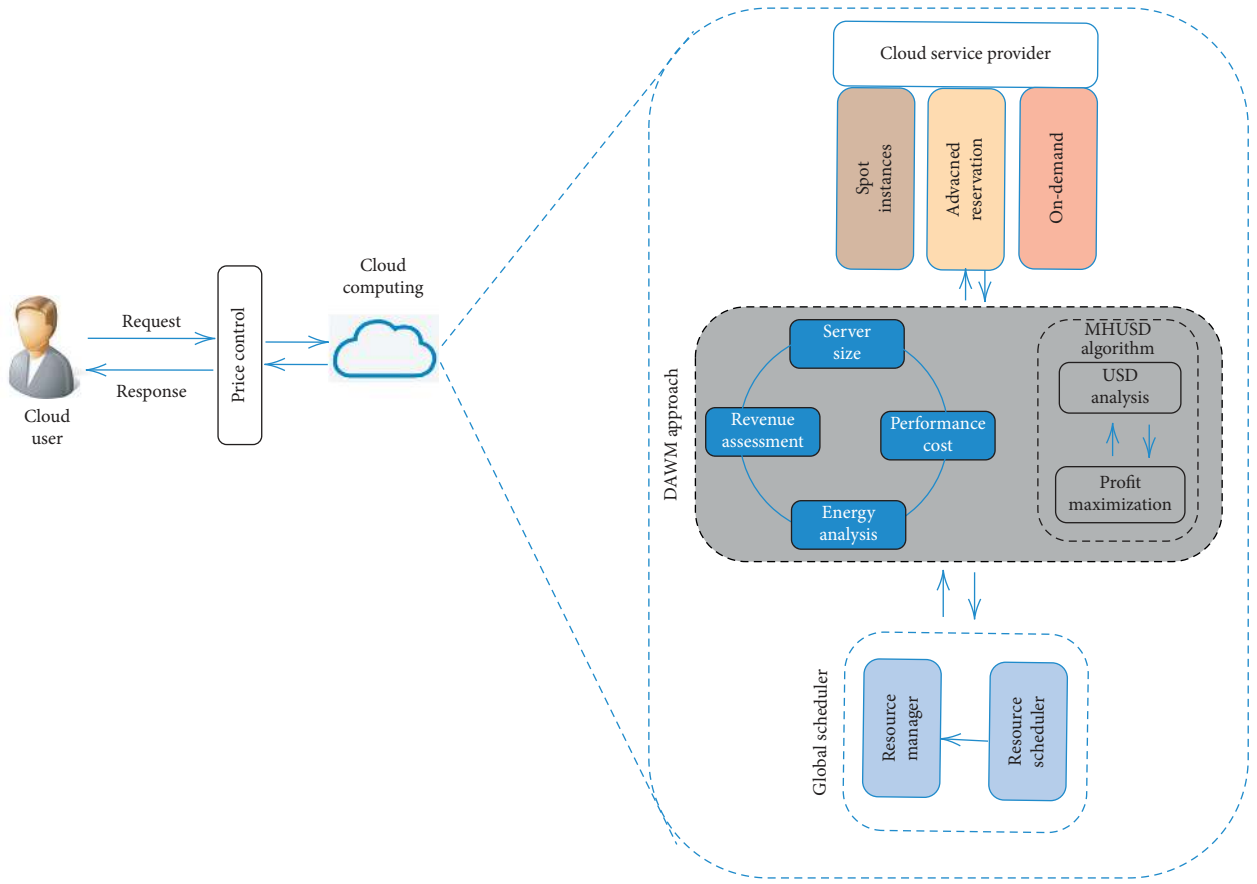

Figure 2: DAWM system model.

$$Z = \frac{u}{N \cdot q} = \frac{u}{N \cdot (m/\overline{x})} = \frac{u \cdot \overline{x}}{N \cdot m}, \tag{1}$$

$$\rho_r = \begin{cases} \rho_0 \dfrac{(N \cdot \rho)^r}{r!}, & r \le N, \\ \rho_0 \dfrac{\left(N^N \cdot \rho^r\right)^r}{N!}, & r > N, \end{cases} \tag{2}$$

where ρ_r refers to probability of r service requests which are executing at a server. In case if there are no tasks/service requests, then the probability of zero service request is

$$\rho_0 = \left(\sum_{r=0}^{N-1} \frac{(N \cdot \rho)^r}{r!} + \frac{(N \cdot \rho)^N}{N!} \cdot \frac{1}{1 - Z} \right). \tag{3}$$

Subsequently, ρ_b is the probability of new arrived SRs, which should wait when the server system is busy executing assigned tasks where ρ_N refers to probability of all N SRs. The probability density function is defined with equation (5), and d refers to service waiting time:

$$\rho_a = \sum_{r=N}^{\infty} \rho_r = \frac{\rho_N}{1 - N}, \tag{4}$$

$$\rho_d en(t) = (1 - \rho_a) \cdot d + N \cdot q \cdot \rho_N \cdot e^{-(1-\rho)N \cdot q(t)}. \tag{5}$$

Figure 3 illustrates the DAG task classification and scheduling scheme that accomplishes by evaluating cost price/unit of the machine, which is magnified with ample of time required for task completion. Therefore, for instance, n is the number of VMs of $F[i]$ type with weight $W[r[i]]$, $\therefore \forall 1 \le i \le n$. Let τ be the required time to finish all the errands on a set of VMs through the DAG-based approach. The collected value/unit time is $\sum_{1 \le i \le n} W[r[i]]$. Appropriately, the complete performance weight is $(\varphi, \vartheta(t), \varpi(t))$, and it is characterized as

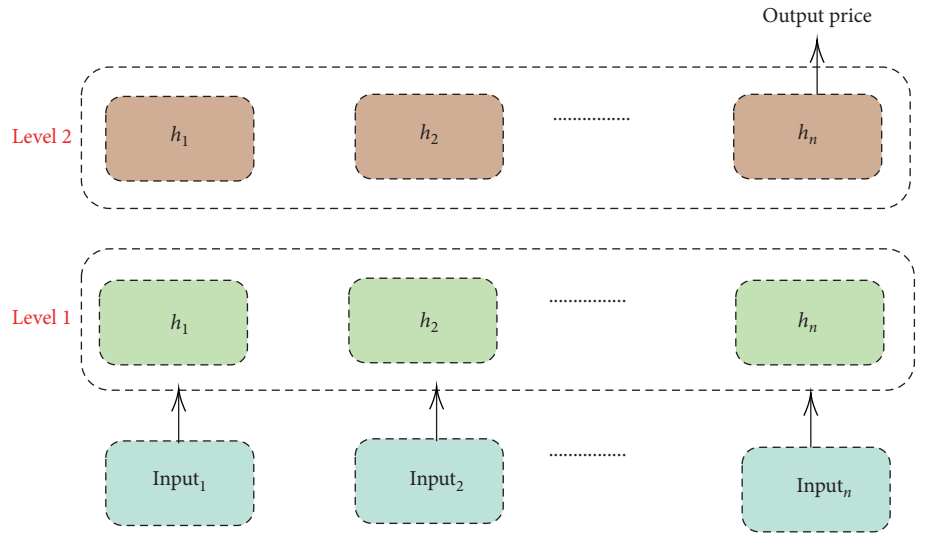

Figure 3: Belief propagation-influenced MRS cost assessment submodel.

$$\text{Weight}(\varphi, \vartheta(t), \varpi(t)) = \sum_{1 \le i \le n} W[r[i]] \times \tau. \tag{6}$$

3.2. Service Level Agreement Model. The SLA is a method which maintains a trade-off between price and service quality between end-user and CSP. Here, the required service attribute x is executed within the response time T, to meet the application deadline:

$$S(x,T) = \begin{cases} cx, & \text{if } 0 \le d \le \left(\dfrac{b}{m_0} - \dfrac{1}{m}\right) \cdot x, \\ \left(c + \dfrac{b \cdot p}{m_0} - \dfrac{p}{m}\right)x - p \cdot d, & \text{if } \left(\dfrac{b}{m_0} - \dfrac{1}{m}\right) \cdot x < d \le \left(\dfrac{c}{p} + \dfrac{b}{m_0} - \dfrac{1}{m}\right) \cdot x, \\ 0, & \text{if } d > \left(\dfrac{c}{p} + \dfrac{b}{m_0} - \dfrac{1}{m}\right) \cdot x, \end{cases} \tag{7}$$

where a is the service cost/unit, d is the penalty cost if any SLA violation, b is the constant weight of SLA, and m_0 is the expected service processing speed. There are three conditions listed even the service request has under waiting time. Therefore, $T = ((d + x)/m)$:

(1)If d has low value than $bc \times m_0$, it provides high-quality, reliable service

(2) If d is in-between the $((b/m_0) - (1/m)) \cdot x < d \le ((c/p) + (b/m_0) - (1/m)) \cdot x$, time interval leads to moderate service quality

(3) If d is longer than $((c/p) + (b/m_0) - (1/m)) \cdot x$, then the service is free because the service request waited long time in queue

Equation (7) is used to assess the prognosticated service charge of the CSP based on 5 parameters: c, p, b, d, and m. Here, c refers to service cost/unit, p refers to SLAV penalty cost, m_0 refers to expected service speed, b refers to SLA constant weight, and d is the average service waiting time.

3.3. User Service Satisfaction Model. User service satisfaction (USS) is estimated in two ways: quality of service (QoS) and price of service (PoS). QoS describes the discrepancy between users' expectations (how to server SR) and users' perceptions (how to perform service). The user's quality of service $(\eta_i^{sq}(x,T))$ is evaluated with

$$\eta_i^{sq}(x,T) = \begin{cases} 1, & \text{if } J_{ac} \ge J_{ex}, \\ e^{-\left|(J_{ac} - J_{ex})/J_{ex}\right|}, & \text{if } J_{ac} \ge J_{ex}. \end{cases} \tag{8}$$

The $\eta_i^{tc} = e^{((S_{ex}-S_{ac})/S_{ex})}$ is a fundamental expression to assess the price of service (PoS) with equation (9). Here, S_{ex} and S_{ac} refer to expected cost and actual cost, respectively:

(1)If $S_{ex} = S_{ac}$, then $\eta_i^{tc} = 1$, shows there is not impact on user satisfaction

(2) If $S_{ex} > S_{ac}$, then it leads to the higher service cost ($\eta_i^{tc} < 1$), and it decreases by increasing the actual price

(3) If $S_{ex} < S_{ac}$, then it leads to the lower service cost ($\eta_i^{tc} > 1$), and it increases by decreasing the actual price

$$\eta_i^{tc}(x,T) = \begin{cases} 1, & \text{if } 0 \le d \le \left(\frac{b}{m_0} - \frac{1}{m}\right) \cdot x \\ e^{\left((1/m)+(d/x)-\left(b/m_0\right)\right)\cdot(p/c)}, & \text{if} \left(\frac{b}{m_0} - \frac{1}{m}\right) \cdot x < d \le \left(\frac{c}{p} + \frac{b}{m_0} - \frac{1}{m}\right), \\ e, & \text{if } d > \left(\frac{c}{p} + \frac{b}{m_0} - \frac{1}{m}\right) \cdot x. \end{cases} \tag{9}$$

The USS (η_i^{sa}) is defined as product of service price and quality of service ($\eta_i^{sa} = \eta_i^{sq}(x,T) + \eta_i^{tc}(x,T)$) with (10).

Such that,

$$\eta_i^{sa} = \begin{cases} 1, & \text{if } 0 \le d \le \left(\frac{b}{m_0} - \frac{1}{m}\right) \cdot x, \\ e^{\left(\left(1+\left((1/m)-\left(b/m_0\right)\right)\cdot(p/c)+d\right)/\left(x\cdot\left((p/c)-\left(1/\left(\left(b/m_0\right)-(1/m)\right)\right)\right)\right)}, & \text{if } \left(\frac{b}{m_0} - \frac{1}{m}\right) \cdot x < d \le \left(\frac{c}{p} + \frac{b}{m_0} - \frac{1}{m}\right), \\ e^{2-d/\left(\left(b/m_0\right)-(1/m)\right)\cdot x}, & \text{if } d > \left(\frac{c}{p} + \frac{b}{m_0} - \frac{1}{m}\right) \cdot x. \end{cases} \tag{10}$$

The product of sum is calculated with equation (8) and equation (9).

3.4. User Demand Service Estimation Model and Algorithm. The user service demand weight factor ($\eta_{i,k}^{\text{expec}}$) assessment plays an essential role to optimize the cost of cloud service provider, and it is estimated with equation (11):

$$\eta_{i,k}^{\exp ec} = \sum_{k=1}^{x} K_i(\chi_k - \gamma_k), \tag{11}$$

where x refers to a list of service attributes, K_i refers to the service attribute weight, χ_k refers to the attribute perception, and γ_k refers to the attribute expectation.

The service demand is formulated as the product of potential demand and user service demand weight factor. It is defined as

$$\eta_{i,k}^{\text{dema}} = 0.25 \times \left(\alpha + \beta \times \eta_{i,k}^{\text{expec}}\right), \quad (\text{where } \alpha, \beta > 0), \tag{12}$$

where α and β refer to constant basic demand and constant potential demand. Subsequently, both values must be greater than >0, such as $\alpha, \beta > 0$.

MHUDS algorithm 1 assesses the user service demand adequately. Lines 1-2 define the entail parameters and attributes for estimation of the user service demand. Line 4 assesses all the service attributes of the cloud service provider and also checks the CPS set. Line 5 helps assess the lower and upper bound value that should not be less than <$\mathbb{R}$. Line 6 estimates the median value of the service attribute demand. Line 7 assesses the $\eta_{i,x}^{\text{dema}}(u_m^k)$ which should not be less than 0. $\eta_{i,x}^{\text{dema}}(u_m^k)$ refers to user service demand of attribute k with middle-range value. Similarly, the rest of the two variables refer to higher and lower values of the user service demand rate. Lines 12–15 are used to update the concerned value at each iteration of time.

3.5. CPS Profit Estimation Model. The CSP profit is assessed based on the gap between the profits gained by acquiring services to users and the monetary cost of processing user SRs. Equation (13) is defined with function number and server speed (i.e., N and m). The average revenue of CSP is estimated as a product of the expected cost of SR and user service demand:

$$S = -0.25 \times \left(\frac{\rho_a \cdot c \cdot \overline{x}}{\rho_a \cdot c \cdot \overline{x} \left((N \cdot m - u\overline{x}) \times \left((b/m_0) - (1/m) \right) + 1 \right)^2} \right) = S(\mathrm{N}, \mathrm{m}), \quad (13)$$

$$\varphi = S(\mathrm{N}, \mathrm{m}) \times \eta_{i,k}^{\mathrm{dema}}, \quad (14)$$

where $\eta_{i,k}^{\mathrm{dema}}$ refers to USD based on user service attribute value. The CSP cost is defined as a paid infrastructure tenant cost and the power cost of system function, and it is assessed with equation (15). The server energy consumption is also estimated with equation (17):

$$\vartheta(t) = N \cdot s \times t, \quad (15)$$

$$\xi(t) = N \cdot \left(\Omega_{nst} \times \partial + \Omega_{st} \right) \cdot t \cdot \xi_s^n, \quad (16)$$

where ∂ refers to server usage, Ω_{nst} refers to dynamic power usage, and Ω_{st} refers to static power usage. Assuming that $\xi_s^n(t)$ refers to energy usage cost at processing time t, the electricity bill $(\varpi(t))$ is defines as

$$\varpi(t) = \xi(t) \times \xi_s^n(t). \quad (17)$$

The CSP profit at t is described as the revenue minus from the rental and electricity cost, and it is estimated with equation (18):

$$G(\mathrm{N}, \mathrm{m}) = \varphi - \vartheta(t) - \varpi(t). \quad (18)$$

3.5.1. CSP Profit Maximization Factor. The probability of having N SRs is described with equation (19). The Taylor series influences approximately $(N! \approx \sqrt{2\pi N}\,(N/e)^N)$ to assess the CSP profit as follows:

$$\rho_N = \rho_0 \times \frac{(N \cdot Z)^N}{N!}, \quad (19)$$

$$\rho_N = \frac{1 - Z}{1 - Z \times \sqrt{2\pi N} \left(\left(e^{Z-1}/Z \right) \right)^N + 1}, \quad (20)$$

updated derivation

$$\rho_N = \frac{1}{\sqrt{2\pi N} \left(\left(e^{Z-1}/Z \right) \right)^N + 1}. \quad (21)$$

The CSP maximized profit assess as follows:

$$S(\mathrm{N}, \mathrm{m}) = -0.25 \times \rho_a \cdot c \cdot \overline{x} \times \left(\frac{1}{\left((N \cdot m - u\overline{x}) \times \left((b/m_0) - (1/m) \right) + 1 \right)^2 \times \left(1/\sqrt{2\pi N} \left(\left(e^{Z-1}/\mathrm{Z} \right) \right)^N + 1 \right)} \right). \quad (22)$$

3.6. DAG Task Scheduling Methodology. The errands are assigned through a computational method, which comes under the DAG-based process by considering the framework's performance weight. It can be observed in Figure 4. We characterize a graph $G = \{V, E\}$. $V = \{v_1, v_2, v_3, \ldots, v_n\}$ where v_i speaks to a comparing errand t_i and it executes consecutively on a machine. $E = e_1, e_2, e_3, \ldots, e_m$ remains priority connection among errands because of information reliability. An errand is not initiated until the last errand remains finished.

Because of dissimilar conditions in the cloud, each PM ability remains to differ. Therefore, we consider the $X[i, j]$ matrix to identify and for a keen track of each errand processing time t_i on j^{th} VM. Here, we have not considered weight and performance factors to measure the assets. In our system, we deliberately utilize a matrix to measure performance time on various VMs, rather than utilizing a consistent weight factor to estimate execution time. As per the data analysis model dataset, we measure each level (L_i) of the convolution network with DAG-based spark. Specifically, each spark stage alludes a vertex, and the connection among 2 phases is compared with organized point. The apexes with 0 degree remain reflected as phases that complete in parallel (P_i). The 0-degree vertices of DAG indicate with L. The

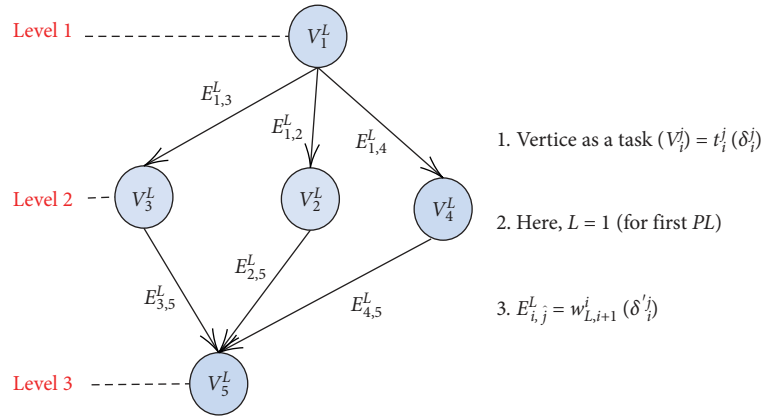

Figure 4: Representation of DAG task, where $v_1 \longrightarrow v_5$ entry and exit nodes with weight factor.

organizing system remains recursively performed and forwards the outcome to any phase of DAG. According to equation (23), we measure most outrageous performance time of all processing phases in parallel (P_i) and that recursively upgrades the task finish time:

$$\underset{i\in S}{\text{Max}}\, T_L^i \le T_{\text{Task}} \le \sum_{i\in S} T_L^i. \tag{23}$$

3.7. Estimation of Optimal Price. The price-demand function estimates optimal price of service by considering the trade-off between service price ϕ and the concern service demand Δ based on their service request mode such as on-demand service, reserved service, and spot instance service. It is formulated as

$$\Delta = \Delta_{re} + (\Delta_{od} - \Delta_{re}) \times \left(\frac{\phi - \phi_{re}}{\phi_{od} - \phi_{re}}\right), \tag{24}$$

where Δ_{od} refers to price-demand of on-demand service and Δ_{re} refers to price-demand of reserved service, and similarly, for price, ϕ_{od} refers to price for on-demand service and ϕ_{od} refers to price for reserved service.

Theorem 1. *Let us assume that the CSP considers $\hbar$ units of time. If service price is φ and average service execution time is t, then the anticipated service price is*

$$\phi_{\exp\, e} = \hbar\phi\left(\frac{1}{1 - e^{-(\hbar/\widehat{t})}}\right). \tag{25}$$

Proof. The CSP considers $\hbar$ units of time, the optimal price is measured with average service execution time t, and it can be measured as

$$\phi(t) = \hbar\left\lceil\frac{t}{\hbar}\right\rceil \times \phi. \tag{26}$$

It is defined as follows: the service request price is $(n+1)\hbar\phi$ in $(n\hbar, (n+1)\hbar]$ time interval.

The probability distribution function of t is

$$g(\tau) = \frac{1}{\tau} \times e^{-(\hbar/\widehat{t})}. \tag{27}$$

The expected price is

$$\begin{aligned}
\phi_{\exp\, e} &= \int_0^\infty \phi(t) \cdot g(t)\mathrm{d}t, \\
&= \sum_0^\infty \int_{n\hbar}^{(n+1)\hbar} g(t)(n+1) \cdot \hbar \cdot \phi\, \mathrm{d}t \\
&= \sum_0^\infty (n+1) \cdot \hbar \cdot \phi\left(-e^{-(\hbar/\widehat{t})}\right)_{n\hbar}^{(n+1)\hbar} \\
&= \hbar \cdot \phi \sum_0^\infty e^{-(n\hbar/\widehat{t})}, \\
&= \hbar \cdot \phi \lim_{n\longrightarrow\infty} \frac{1 - \left(e^{-(n/\widehat{t})}\right)^n}{1 - \left(e^{-(n/\widehat{t})}\right)} \\
&= \hbar \cdot \phi \frac{1}{1 - \left(e^{-(n/\widehat{t})}\right)}.
\end{aligned} \tag{28}$$

Hence, the theorem is proved and the forecasting service arrival demand is approximately $u = \Delta u_{\max} = \Delta_{re} + (\Delta_{od} - \Delta_{re}) \times ((\phi - \phi_{re})/\phi_{od} - \phi_{re})u_{\max}$.

The forecasting service price is $S_{\text{expec}} = \varphi - \text{CSPcost} = \varphi - n\phi_{re}$. So, the maximum price must have to measure $(\partial s_{\text{expec}}/\partial\phi) = 0$, such that

$$\frac{\partial u}{\partial \phi} = \left(\frac{\Delta_{od} - \Delta_{re}}{\phi_{od} - \phi_{re}}\right) u_{\max}, \quad \text{where } \frac{\partial u\phi}{\partial \phi} = u + \left(\frac{\Delta_{od} - \Delta_{re}}{\phi_{od} - \phi_{re}}\right) u_{\max}\phi, \tag{29}$$

where $s^{\text{los}} = ((Z^n e^{n(1-Z)})/\sqrt{2\pi n})$ refers to loss of server profit, but the probability of expected server profit loss is

$$\begin{aligned}
\frac{\partial s^{\text{los}}_{\text{exp } ec}}{\partial \phi} &= \frac{1}{\sqrt{2\pi n}}\left[\frac{\partial\left(e^{n(1-Z)}\right)}{\partial \phi} Z^n + e^{n(1-Z)}\frac{\partial Z^n}{\partial \phi}\right] \\
&= \frac{1}{\sqrt{2\pi n}}\left[\left(Z^n e^{n(1-Z)}\left(-\frac{1}{\ell}\frac{\partial u}{\partial \phi}\right)\right)\right. \\
&\quad \left. + Z^{n-1}\cdot n\cdot e^{n(1-Z)}\left(-\frac{1}{n\ell}\frac{\partial u}{\partial \phi}\right)\right] \\
&= \frac{Z^n e^{n(1-Z)}}{\sqrt{2\pi n}}\frac{1-Z}{\ell Z}\frac{\partial u}{\partial \phi} \\
&= s^{\text{los}}\frac{1-Z}{\ell Z}\frac{\partial u}{\partial \phi}, \quad \text{since } s^{\text{los}} = \frac{Z^n e^{n(1-Z)}}{\sqrt{2\pi n}}.
\end{aligned} \tag{30}$$

Subsequently, the probability of forecasting service price is

$$\begin{aligned}
\frac{\partial s_{\text{expec}}}{\partial \phi} &= \left(u + \frac{\partial u}{\partial \phi}\phi\left(1 - s^{\text{los}}(t)\right) + u\phi t\left(-s^{\text{los}}\frac{1-Z}{\ell Z}\frac{\partial u}{\partial \phi}\right)\right) \\
&= u\left(1 - s^{\text{los}}(t)\right) \\
&\quad + \phi t\frac{\partial u}{\partial \phi}\left[\left(1 - s^{\text{los}}(t)\right) - s^{\text{los}}(t)n(1-Z)\right].
\end{aligned} \tag{31}$$

□

3.8. Estimating Optimal Price. In Algorithm 2, the partial derivative is formulated through s^{los}. It formulates accurate service price though the service arrival rate is high with low profit loss. Lines 1–3 define the input variables, and line 4 applies the models to all arrived service requests. Lines 6–9 estimate the optimal price demand, and lines 10–19 estimate optimal price value based on equations (31) and (13).

3.9. DAWM Algorithm for Cloud Server Size and Cost Analysis. Algorithm 3 assess the server size and performance cost. It assesses the customer satisfaction from the machine economic growth ratio by leveraging the cloud server configuration (CSC) called server size, task arrival rate, and performance cost of the machine. Therefore, the CSC has a direct impact on the cloud user service satisfaction rate and the inadequate customer satisfaction, and it also has direct impact on service request arrival rate. Line 1 defines the essential input parameters to accomplish the objectives. Lines 2–5 assess the service execution cost using equation (13) and update the machine matrix $H[i, j]$, for effective prognostication of server configuration size. Lines 6 and 7 update the all machine execution speed rates and maintained in an array. Lines 8 to 10 assess the performance cost in association with CSC (s), service resource requirement rate (K), SLAV penalty rate (L), and energy and resource tenant cost. Lines 12–15 update the iterative value to mitigate performance rate and system execution cost.

4. Experimental Result Analysis

The proposed DAWM is simulated with real data in MATLAB R2017b, and the system specifications are 8 GB DDR4 memory and an Intel Core i7-6700HQ CPU with 2.6 GHz. We consider DAG $[V, E]$ consisting 25–150 sensors. Every network enables 5% of data centres in the network size, and its capacity varies from 5000 to 75000 GHz. The active servers are varying from 1000 to 1500. The idle server constant energy consumption is 90 – 180 Watt; else the energy consumption is measured based on its energy usage rate, and it is in range $[0.5, 1.5]$; energy price is $([15, 55]/\text{Mwh})$. The link bandwidth between sensors varies from 1500 to 25, 000 Mbps and delay transmission is 3 – 6 ms. The revenue gain is $[0.15, 0.25]$, which is not static. Each service execution bandwidth is set from 15 – 25 Mbps, computing demand is 3 – 5 GHz, and the execution of each service is 5 – 30 (data packets/ms). The simulation parameters related to power cost, constant workloads, CSC, service requirement rate, SLAV penalty rate, energy cost, tenant cost, and current CSP margin profit are listed in Table 1.

Figure 5 illustrates the average execution time required to process the user service request. It has been compared with four state-of-the-art approaches (SPEA2, COMCPM, NSGA-II, and OMCPM) which are published recently. It is noticed that the proposed approach has high-performance rate than remaining approaches such as 41.2%, 55.56%, 59.89%, and 61.52% faster than SPEA2, COMCPM, NSGA-II, and OMCPM, respectively.

Figure 6 illustrates profit, revenue, and cost of the proposed system and SPEA2, COMCPM, NSGA-II, and OMCPM approaches. The proposed system achieved moderately high revenue by 10%, 8.1%, 8.9%, and 8.91% than SPEA2, COMCPM, NSGA-II, and OMCPM approaches. Subsequently, our approach achieves 2.31%, 2.01%, 1.7%, and 1.37% high profit than four approaches, since our approach estimates the demand of service request and it analyses the machine performance before assigning the load. The reason is that user service request (USR) is submitted to the service provider, which runs on a multiserver system to deliver the response for the received service requests. The CSP assesses the machine data with our deep learning data analytical model. It makes an accurate decision to enhance the system performance by preserving service cost and to enhance the revenue gain consolidating each machine performance. The second reason is that the task is being scheduled base on DAG theory which influences the energy and resource of the system leads to enhance the revenue and optimizes the service request cost.

Figure 7 shows the user service demand flexibility impact. We can observe that the active cloud server (from 15 to 75) count and the processing speed m of active servers are

input: CPS: $N = \{N_1 + N_2 + N_3 + \cdots + N_n\}$
output: user demand service
(1) Let initialize $\alpha \neq 0, \beta \neq 0$, $\eta_{i,k}^{\text{expec}} \neq 0$;
(2) Int u, define range $[u_l^k, u_h^k]$, $\eta_{i,x}^{\text{dema}}(u_l^k) > 0$, $\eta_{i,x}^{\text{dema}}(u_h^k) < 0$;
(3) **for** each $N_i \in N$ **do**
(4) Estimate $\eta_{i,k}^{\text{dema}} = 0.25 \times (\alpha + \beta \times \eta_{i,k}^{\text{expec}}) - u$;
(5) **while** $(\eta_{i,x}^{\text{dema}}(u_l^k) - \eta_{i,x}^{\text{dema}}(u_h^k)) > \mathbb{R}$ **do**
(6) $\eta_{i,k}^{\text{dema}}(u_m^k) = ((\eta_{i,k}^{\text{dema}}(u_l^k) - \eta_{i,k}^{\text{dema}}(u_h^k))/2)$;
(7) **if** $\eta_{i,x}^{\text{dema}}(u_m^k) < 0$ **then**
(8) Assign $\eta_{i,x}^{\text{dema}}(u_h^k) \longleftarrow \eta_{i,x}^{\text{dema}}(u_m^k)$;
(9) **else**
(10) Assign $\eta_{i,x}^{\text{dema}}(u_l^k) \longleftarrow \eta_{i,x}^{\text{dema}}(u_m^k)$;
(11) **end**
(12) Update $\eta_{i,x}^{\text{dema}}(u_l^k)$ and $\eta_{i,x}^{\text{dema}}(u_m^k)$;
(13) Estimate $\eta_{i,k}^{\text{dema}}(u_m^k) = ((\eta_{i,k}^{\text{dema}}(u_l^k) - \eta_{i,k}^{\text{dema}}(u_h^k))/2)$;
(14) **end**
(15) Confine it as potential value for next iteration $\eta_{i,k}^{\text{dema}}(u_m^k)$;
(16) Return user demand service value.
(17) **end**

Algorithm 1: MHUDS algorithm.

input: $u, t, n, \phi_{re}, \phi_{od}, \Delta_{re}, \Delta_{od}$
output: optimal price of service
(1) Let $\phi_{\text{opti}} = -\infty, \Delta_{\text{opti}} = -\infty$;
(2) $\phi_{st} \longleftarrow$ least price, server usage<1;
(3) $\phi_{en} \longleftarrow \phi_{od}$;
(4) **for** *each* $N_i \in \mathbf{N}$ **do**
(5) Estimate Δ_{st} and Δ_{st} using equations (30) and (13);
(6) **if** $\Delta_{st} \times \Delta_{st} > 0$ **then**
(7) $\phi_{\text{opti}} = \phi_{st}$;
(8) Estimate Δ_{opti} using (13) and with $S_{\text{expec}} = \varphi - \text{CSPcost} = \varphi - n\phi_{re}$;
(9) **end**
(10) **while** $\Delta_{st} \times \Delta_{ed} >$ error **do**
(11) $\phi_{\text{mid}} = ((\phi_{st} + \phi_{en})/2)$;
(12) Estimate Δ_{mid} using (13) and (31);
(13) **if** $\Delta_{st} \times \Delta_{st} > 0$ **then**
(14) $\phi_{st} \longleftarrow \phi_{\text{mid}}$;
(15) **else**
(16) $\phi_{en} \longleftarrow \phi_{\text{mid}}$;
(17) **end**
(18) **end**
(19) $\phi_{\text{opti}} = ((\phi_{st} + \phi_{e\,d})/2)$;
(20) **end**

Algorithm 2: Optimal price estimation algorithm.

high, but there is no impact on the service execution demand rate. If the server count increases, then the user service demand execution rate does not increase, and it is sometimes stable to cope up the reliable quality of service with adequate computing performance. If the USD is high, the server system is frequently unable to meet the service demand requirement synchronously. In such cases, if the customer waits for a long time, then the USD rate becomes low due to low service demand. Usually, the USD may remain constant when the USD market is stable, which would not affect third-party factors.

Figure 8 shows CSP profit outcomes. As we can observe, the profit rate is drastically decreased when the active servers are increased from 15 to 75. The high server processing speed m has no impact as we expected. The profit ratio is increased due to the USD rate increment than the new active server cost. The revenue enhancement and server size factors are not impacting server cost, but USD will get diminished due to the decrement of CSP profit. Consequently, the profit returns stable when the USD becomes constant. Figure 9 shows the server processing speed comparative study. The server processing speed is decreased when the server size

input: (1) Host set: $\mathbf{N} = \{N_1 + N_2 + N_3 + \cdots + N_n\}$,
(2) Ex: execution time matrix of host
(3) C: cost weight matrix of host/VM
output: performance cost of server
(1) Let $T = \{T_1 + T_2 + T_3 + \cdots + T_t\}$
(2) **for** *each* $\mathrm{N}_i \in \mathbf{N}$ **do**
(3) Find minimum cost-effective host (6) and (13)
(4) $N_j[i] = N_j + H[i, j]$
(5) **end**
(6) $T_{\text{tot}} = \sum_{\forall T_i} T_i$
(7) Update $N_j[i]$, $\longleftarrow$ *SortHostCostQueue*$(\varphi, \vartheta(t), \varpi(t))$
(8) **for** each i to h_m **do**
(9) $\lambda = \text{Cost}(K, S, L) = \sum_{1 \le i \le K} C[S[i]] \times T_{ToT}$
(10) $R_i = \text{Cost}(N_j) - \lambda_i$
(11) **end**
(12) $K' = K - \lambda_i$
(13) **for** each $R_i \in N_n$ **do**
(14) λ_i^{++}
(15) **end**
(16) Return performance cost of server

Algorithm 3: DAWM algorithm.

Table 1: Simulation parameters.

S. no.	Notation	Value
1.	m_0	1.5 BIPS
2.	b	5
3.	$\overline{x}$	1.5 BI
4.	c	20 units/BI
5.	s	20 units/sec
6.	N_c	9.5
7.	p	5
8.	Ω_{st}	2 Watts/sec
9.	$\xi_s^n(t)$	0.1 unit/Watt × sec

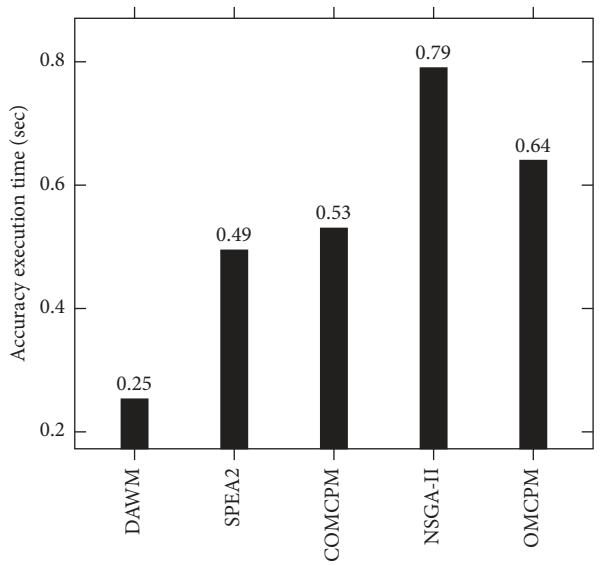

Figure 5: Average execution time of DAWM, SPEA2, COMCPM, NSGA-II, and OMCPM.

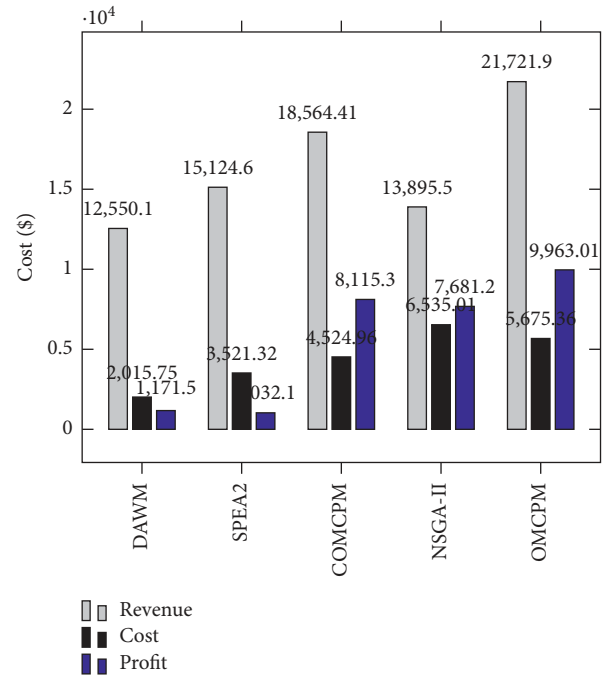

Figure 6: Cost, profit, and revenue of DAWM, SPEA2, COMCPM, NSGA-II, and OMCPM.

increases; the computation size is fixed, which restricts the execution of the services. The increased server count demands to decrease the systems service execution speed. Figure 10 illustrates the increased profit during server size, and USD rates are increased. The high-computation-required USDs are led to enhance the CSP profit. We can observe that the USD is moderate due to server size enhancement. We noticed that if active servers are less but the server speed is high, the profit increases. If we maintain

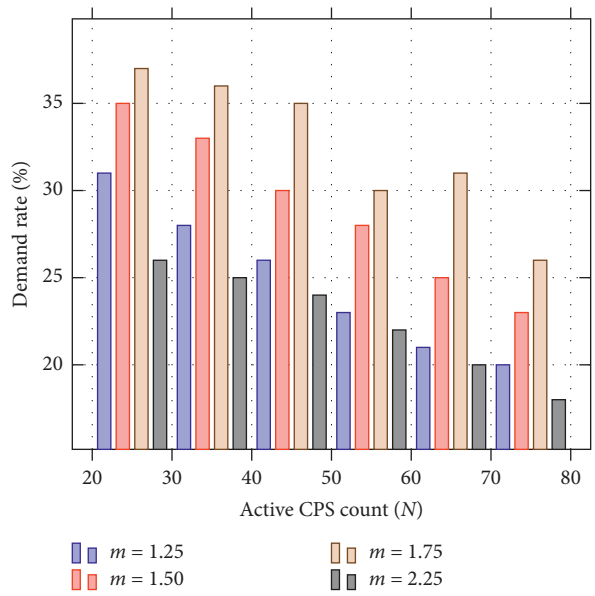

Figure 7: User service demand analysis over server infrastructure size with various service execution speeds.

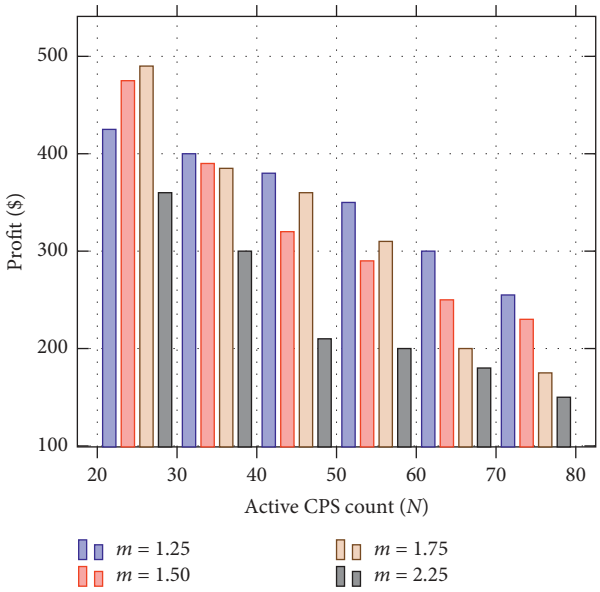

Figure 8: Profit analysis over server infrastructure size with various service execution speeds.

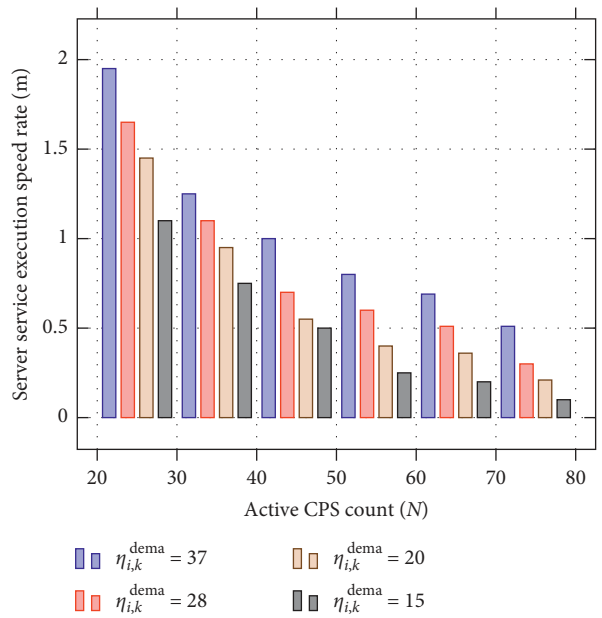

Figure 9: Server service execution speed rate over server infrastructure size with various user service demands ($\eta_{i,k}^{dema}$).

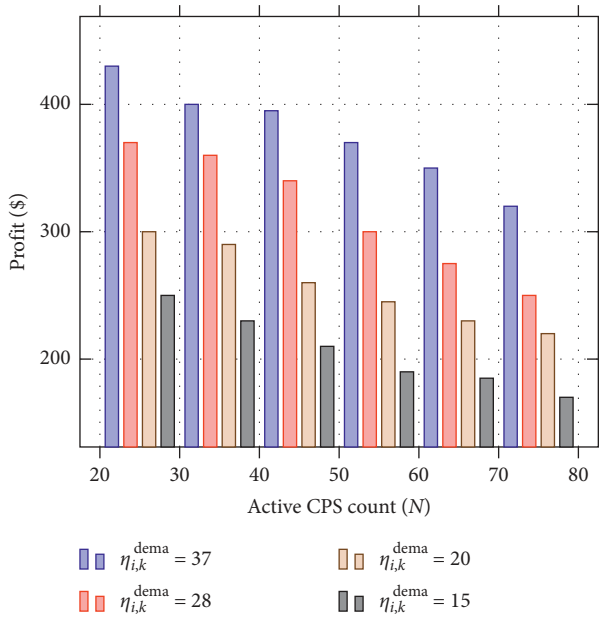

Figure 10: Profit analysis over server infrastructure size with various user service demands ($\eta_{i,k}^{dema}$).

constant computing capacity, the server speed is impacted by the increase of active server count, which causes a decrease in the profit. Therefore, if the server size is peak and speed remains constant, it saves the energy cost and impacts CSP profit.

Figure 11 shows the comparative analyses of the server size and profit by regulating the server speed and USD rate. To assess the outcomes, we have used Table 1 listed parameters. If we increase the m value, then the active server size gets low due to m value increment under USD certainty. The profit gets impact when the energy cost is high and influences service execution speed to diminish CSP profit.

Table 2 shows the comparative study analysis concerning all state-of-the-art approaches. The proposed system has outstanding profit, such as a 35.5% average. Subsequently, the profit is accomplished due to the data analysis model, and also performance rate of our system remains increased than existing approaches. The machine performance and execution cost measurement estimations played an essential role to gain adequate noticeable profit for CSP.

Table 3 illustrates our approach's simulation outcomes with the unit price 0.6$ and average execution time 0.6 ms.

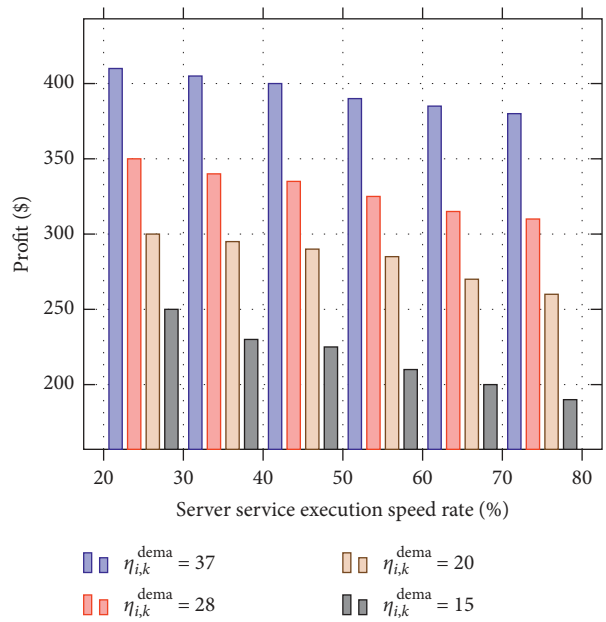

Figure 11: Profit analysis over server infrastructure size with various user service demands ($\eta_{i,k}^{dema}$).

Table 2: Profit comparative analysis based on server size and server speed.

Server size	DAWM		SPEA2		COMCPM		NSGA-II		OMCPM	
	SS	Profit	SS	Profit	SS	Profit	SS	Profit	SS	Profit
m2.6xlarge	4.5	**590**	3.9	520	3.5	480	3	440	2.8	**530**
m2.2xlarge	3.5	**510**	3.3	480	2.9	415	2.2	395	2.1	**490**
m1.2xmedi	2	**440**	2.2	315	2.5	335	1.8	360	1.9	**450**
m1.xsmall	1.5	**370**	1.9	255	1.75	290	1.7	300	1.5	**320**

Note: SS, server speed.

Table 3: Simulation outcome with the unit price 0.6$ and average execution time 0.6 ms.

u_{max}	n_{opti}	ϕ_{opti}	Δ_{opti}	User cost (%)	Error (%)
50	110	7.59	251.32	48.912	2.15
60	118	7.21	310.68	48.245	1.91
70	125	7.84	372.98	48.329	1.88
80	167	7.99	415.25	49.786	1.52
90	182	7.23	490.89	49.791	1.49
100	195	7.51	525.15	51.012	1.32
110	229	7.58	590.69	40.452	1.31
120	250	7.32	610.15	45.697	1.28

The service price, service price-demand, maximum average service arrival rate, error rate, and user cost are assessed with average service execution time.

5. Conclusion

The proposed approach has been designed based on a belief propagation-influenced analytical data model to enhance CSP profit through DAG-based task and resource scheduling policy. It optimizes the CDC asset usage rate by consolidating overprovisioning machines. Cloud service suppliers drive the data utility analytic method on machines with low-resource usage rates to preserve CDC usage and performance cost and avoid instant repudiations/migrations.

It initially recognizes feasible servers after the first iteration by concocting the data analytic weight measurement (DAWM) model. The DAWM model optimizes the cloud service provider's average cost by 51% due to considering each machine's cost and revenue during an effective resource slicing process. The multiobjective heuristic user service demand (MHUSD) algorithm accomplished average server performance by 41% and average CSP revenue gain by 35% due to CPS profit estimation model and the user service demand (USD) model with dynamic acyclic graph (DAG) phenomena by providing adequate service reliability. It

considers service demand weight, service tenant cost, and machine energy cost. Subsequently, the MHUSD algorithm also considers maximum baring wait-time of end-user to maximize CSP revenue and optimize operational energy cost. Google cloud tracer confines the optimized average system profit by 590$, and service execution speed is 4.5 sec/MIPS with the m2.6X large core system. The simulation results show that our system has an average service execution speed faster than the remaining approaches, such as 41.2%, 55.56%, 59.89%, and 61.52% faster than SPEA2, COMCPM, NSGA-II, and OMCPM, respectively. Subsequently, the proposed system achieved moderately high revenue by 10%, 8.1%, 8.9%, and 8.91% than SPEA2, COMCPM, NSGA-II, and OMCPM approaches and profit by 2.31%, 2.01%, 1.7%, and 1.37% than the state-of-the-art approaches.

Data Availability

No data were used to support the findings of this study.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] M. Cisco, "An innovative business model for cloud providers," *Whitepaper*, vol. 2019, 2019, https://www.cisco.com/c/dam/enus/solutions/trends/cloud/docs/aninnovativebusinesswhitepaper.pdf.

[2] J. Ru, Y. Yang, J. Grundy, J. Keung, and L. Hao, "A systematic review of scheduling approaches on multi-tenancy cloud platforms," *Information and Software Technology*, vol. 132, Article ID 106478, 2020.

[3] M. Kumar, S. C. Sharma, A. Goel, and S. P. Singh, "A comprehensive survey for scheduling techniques in cloud computing," *Journal of Network and Computer Applications*, vol. 143, pp. 1–33, 2019.

[4] V. Seethalakshmi, V. Govindasamy, and V. Akila, "Hybrid gradient descent spider monkey optimization (hgdsmo) algorithm for efficient resource scheduling for big data processing in heterogenous environment," *Journal of Big Data*, vol. 7, no. 1, pp. 1–25, 2020.

[5] E. Bugingo, D. Zhang, Z. Chen, and W. Zheng, "Towards decomposition based multi-objective workflow scheduling for big data processing in clouds," *Cluster Computing*, vol. 24, pp. 115–139, 2020.

[6] Y. Wen, J. Liu, W. Dou, X. Xu, B. Cao, and J. Chen, "Scheduling workflows with privacy protection constraints for big data applications on cloud," *Future Generation Computer Systems*, vol. 108, pp. 1084–1091, 2020.

[7] W. Ahmad, B. Alam, S. Ahuja, and S. Malik, "A Dynamic vm provisioning and de-provisioning based cost-efficient deadline-aware scheduling algorithm for big data workflow applications in a Cloud environment," *Cluster Computing*, vol. 24, pp. 249–278, 2020.

[8] Y. Tang, Y. Yuan, and Y. Liu, "Cost-aware reliability task scheduling of automotive cyber-physical systems," *Microprocessors and Microsystems*, Article ID 103507, 2020.

[9] S. Long, W. Long, Z. Li, K. Li, Y. Xia, and Z. Tang, "A game-based approach for cost-aware task assignment with qos constraints in large data centers," *IEEE Annals of the History of Computing*, vol. 32, pp. 1629–1640, 2020.

[10] T. D. Nguyen, V.-N. Pham, L. N. Huynh, M. D. Hossain, E.-N. Huh et al., "Modeling data redundancy and cost-aware task allocation in mec-enabled internet-of-vehicles applications," *IEEE Internet of Things Journal*, vol. 8, no. 3, 2020.

[11] J. Khamse-Ashari, I. Lambadaris, G. Kesidis, B. Urgaonkar, and Y. Zhao, "An efficient and fair multi-resource allocation mechanism for heterogeneous servers," *IEEE Transactions on Parallel and Distributed Systems*, vol. 29, no. 12, pp. 2686–2699, 2018.

[12] K. Karthiban and J. S. Raj, "An efficient green computing fair resource allocation in cloud computing using modified deep reinforcement learning algorithm," *Soft Computing*, vol. 24, no. 19, pp. 14933–14942, 2020.

[13] W. Bai, J. Zhu, S.-W. Huang, and H. Zhang, "A queue waiting cost-aware control model for large scale heterogeneous cloud datacenter," *IEEE Transactions on Cloud Computing*, no. 1, p. 1, 2020.

[14] P. Cong, "Personality-and value-aware scheduling of user requests in cloud for profit maximization," *IEEE Transactions on Cloud Computing*, vol. 99, p. 1, 2020.

[15] K. K. Chakravarthi, L. Shyamala, and V. Vaidehi, "Budget aaware sscheduling aalgorithm for wworkflow aapplications in IaaS cclouds," *Cluster Computing*, vol. 23, pp. 3405–3419, 2020.

[16] C. Li, Y. Zhang, Z. Hao, and Y. Luo, "An effective scheduling strategy based on hypergraph partition in geographically distributed datacenters," *Computer Networks*, vol. 170, no. 4, Article ID 107096, 2020.

[17] P. Cong, G. Xu, T. Wei, and K. Li, "A survey of profit optimization techniques for cloud providers," *ACM Computing Surveys*, vol. 53, no. 2, pp. 1–35, 2020.

[18] J. Mei, K. Li, Z. Tong, Q. Li, and K. Li, "Profit maximization for cloud brokers in cloud computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 1, pp. 190–203, 2019.

[19] Y. Ma, W. Liang, Z. Xu, and S. Guo, "Profit maximization for admitting requests with network function services in distributed clouds," *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 5, pp. 1143–1157, 2019.

[20] J. Wan, R. Zhang, X. Gui, and B. Xu, "Reactive pricing: an adaptive pricing policy for cloud providers to maximize profit," *IEEE Transactions on Network and Service Management*, vol. 13, no. 4, pp. 941–953, 2016.

[21] P. Kaur and S. Mehta, "Resource provisioning and work flow scheduling in clouds using augmented shuffled frog leaping algorithm," *Journal of Parallel and Distributed Computing*, vol. 101, pp. 41–50, 2017.

[22] H. Won, M. C. Nguyen, M.-S. Gil, and Y.-S. Moon, "Advanced resource management with access control for multitenant hadoop," *Journal of Communications and Networks*, vol. 17, no. 6, pp. 592–601, 2015.

[23] M. Tanaka and Y. Murakami, "Strategy-proof pricing for cloud service composition," *IEEE Transactions on Cloud Computing*, vol. 4, no. 3, pp. 363–375, 2014.

[24] C. Liu, K. Li, K. Li, and R. Buyya, "A new cloud service mechanism for profit optimizations of a cloud provider and its users," *IEEE Transactions on Cloud Computing*, vol. 9, no. 1, pp. 14–26, 2017.

[25] H. Xu and B. Li, "Dynamic cloud pricing for revenue maximization," *IEEE Transactions on Cloud Computing*, vol. 1, no. 2, pp. 158–171, 2013.

[26] R. Mohamadi Bahram Abadi, A. M. Rahmani, and S. H. Alizadeh, "Server consolidation techniques in virtualized data centers of cloud environments: a systematic literature review," *Software: Practice and Experience*, vol. 48, no. 9, pp. 1688–1726, 2018.

[27] M. S. Mekala, A. Jolfaei, G. Srivastava, X. Zheng, A. Anvari-Moghaddam, and P. Viswanathan, "Resource offload consolidation based on deep-reinforcement learning approach in cyber-physical systems," *IEEE Transactions on Emerging Topics in Computational Intelligence*, pp. 1–10, 2020.

[28] F. Farahnakian, A. Ashraf, T. Pahikkala et al., "Using ant colony system to consolidate vms for green cloud computing," *IEEE Transactions on Services Computing*, vol. 8, no. 2, pp. 187–198, 2014.

[29] S. Singh and I. Chana, "A survey on resource scheduling in cloud computing: Issues and challenges," *Journal of Grid Computing*, vol. 14, no. 2, pp. 217–264, 2016.

[30] C. Liu, K. Li, and K. Li, "A game approach to multi-servers load balancing with load-dependent server availability consideration," *IEEE Transactions on Cloud Computing*, vol. 9, pp. 1–13, 2018.

[31] I. Mohiuddin and A. Almogren, "Workload aware vm consolidation method in edge/cloud computing for iot applications," *Journal of Parallel and Distributed Computing*, vol. 123, pp. 204–214, 2019.

[32] J. Cao, K. Li, and I. Stojmenovic, "Optimal power allocation and load distribution for multiple heterogeneous multicore server processors across clouds and data centers," *IEEE Transactions on Computers*, vol. 63, no. 1, pp. 45–58, 2013.

[33] M. Guo, Q. Guan, and W. Ke, "Optimal scheduling of vms in queueing cloud computing systems with a heterogeneous workload," *IEEE Access*, vol. 6, pp. 15 178–15 191, 2018.

[34] H. Chen, Y. Li, R. H. Y. Louie, and B. Vucetic, "Autonomous demand side management based on energy consumption scheduling and instantaneous load billing: an aggregative game approach," *IEEE Transactions on Smart Grid*, vol. 5, no. 4, pp. 1744–1754, 2014.

[35] M. S. Mekala and V. Perumal, "Machine learning inspired phishing detection (pd) for efficient classification and secure storage distribution (ssd) for cloud-iot application," in *Proceedings of the 2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 202–210, Canberra, Australia, 2020.

[36] J. Mei, K. Li, and K. Li, "Customer-satisfaction-aware optimal multiserver configuration for profit maximization in cloud computing," *IEEE Transactions on Sustainable Computing*, vol. 2, no. 1, pp. 17–29, 2017.

[37] J. Cao, K. Hwang, K. Li, and A. Y. Zomaya, "Optimal multiserver configuration for profit maximization in cloud computing," *Ieee Transactions on Parallel and Distributed Systems*, vol. 24, no. 6, pp. 1087–1096, 2012.

[38] N. Gholipour, E. Arianyan, and R. Buyya, "A novel energy-aware resource management technique using joint vm and container consolidation approach for green computing in cloud data centers," *Simulation Modelling Practice and Theory*, vol. 104, p. 102127, 2020.

[39] Y.-L. Chou, J.-M. Yang, and C.-H. Wu, "An energy-aware scheduling algorithm under maximum power consumption constraints," *Journal of Manufacturing Systems*, vol. 57, pp. 182–197, 2020.

[40] I. H. Sin and B. D. Chung, "Bi-objective optimization approach for energy aware scheduling considering electricity cost and preventive maintenance using genetic algorithm," *Journal of Cleaner Production*, vol. 244, Article ID 118869, 2020.

[41] M. S. M. and P. Viswanathan, "Equilibrium transmission bi-level energy efficient node selection approach for internet of things," *Wireless Personal Communications*, vol. 108, no. 3, pp. 1635–1663, 2019.

[42] M. S. M. and P. Viswanathan, "A survey: energy-efficient sensor and vm selection approaches in green computing for x-iot applications," *International Journal of Computers and Applications*, vol. 42, no. 3, pp. 290–305, 2020.

[43] M. Basset, R. Mohamed, M. Elhoseny, A. K. Bashir, A. Jolfaei, and N. Kumar, "Energy-aware marine predators algorithm for task scheduling in iot-based fog computing applications," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 7, pp. 5068–5076, 2020.

[44] P. Hosseinioun, M. Kheirabadi, S. R. Kamel Tabbakh, and R. Ghaemi, "A new energy-aware tasks scheduling approach in fog computing using hybrid meta-heuristic algorithm," *Journal of Parallel and Distributed Computing*, vol. 143, pp. 88–96, 2020.

[45] A. S. Abohamama and E. Hamouda, "A hybrid energy-Aware virtual machine placement algorithm for cloud environments," *Expert Systems with Applications*, vol. 150, p. 113306, 2020.

[46] X. Li, W. Yu, R. Ruiz, and J. Zhu, "Energy-aware cloud workflow applications scheduling with geo-distributed data," *IEEE Transactions on Services Computing*, vol. 9, p. 1, 2020.

[47] H. Li, Y. Zhao, and S. Fang, "Csl-driven and energy-efficient resource scheduling in cloud data center," *The Journal of Supercomputing*, vol. 76, no. 1, pp. 481–498, 2020.

[48] S. Ghanavati, J. H. Abawajy, and D. Izadi, "An energy aware task scheduling model using ant-mating optimization in fog computing environment," *IEEE Transactions on Services Computing*, vol. 9, p. 1, 2020.

[49] M. S. Mekala and P. Viswanathan, "Energy-efficient virtual machine selection based on resource ranking and utilization factor approach in cloud computing for iot," *Computers & Electrical Engineering*, vol. 73, pp. 227–244, 2019.

[50] T. Dong, F. Xue, C. Xiao, and J. Li, "Task scheduling based on deep reinforcement learning in a cloud manufacturing environment," *Concurrency and Computation: Practice and Experience*, vol. 32, no. 11, Article ID e5654, 2020.