Hindawi
Mobile Information Systems
Volume 2022, Article ID 7283516, 24 pages
https://doi.org/10.1155/2022/7283516

Review Article

Resource Provisioning Techniques in Multi-Access Edge Computing Environments: Outlook, Expression, and Beyond

S. Durga,[1] **Esther Daniel**,[2] **J. Andrew Onesimu**,[3] **and Yuichi Sei**[4]

[1]*Department of Information Technology, Sri Krishna College of Engineering and Technology, Coimbatore 641008, India*
[2]*Department of Computer Science and Engineering, Karunya Institute of Technology and Sciences, Coimbatore 641114, India*
[3]*Department of Computer Science and Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal 576104, India*
[4]*Department of Informatics, The University of Electro-Communications, Chofu 1828585, Japan*

Correspondence should be addressed to S. Durga; durga.sivan@gmail.com and J. Andrew Onesimu; onesimu@gmail.com

Received 29 April 2022; Revised 25 July 2022; Accepted 8 December 2022; Published 19 December 2022

Academic Editor: Giovanni Nardini

Mobile cloud computing promises a research foundation in information and communication technology (ICT). Multi-access edge computing is an intermediate solution that reduces latency by delivering cloud computing services close to IoT and mobile clients (MCs), hence addressing the performance issues of mobile cloud computing. However, the provisioning of resources is a significant and challenging process in mobile cloud-based environments as it organizes the heterogeneous sensing and processing capacities to provide the customers with an elastic pool of resources. Resource provisioning techniques must meet quality of service (QoS) considerations such as availability, responsiveness, and reliability to avoid service-level agreement (SLA) breaches. This investigation is essential because of the unpredictable change in service demands from diverse regions and the limits of MEC's available computing resources. In this study, resource provisioning approaches for mobile cloud computing are thoroughly and comparatively studied and classified as taxonomies of previous research. The paper concludes with an insightful summary that gives recommendations for future enhancements.

1. Introduction

Currently, hundreds of millions of mobile phones are in use worldwide. The ubiquity of those devices could be leveraged to help utilize the distributed and scalable services from the cloud [1]. In recent years, with the upcoming innovation of mobile phone technologies and the Internet, current mobile users are experiencing a new landscape of services [2]. Apart from the usage of distributed application frameworks, which permit software engineers all around the world to plan progressively complex projects that require more processing power and compute capability [3], mobile cloud computing (MCC) is a new computing paradigm that plays a significant role in urban environments due to the growing number of mobile clients it serves [4–6]. With rapid improvements in the cloud and mobile technologies, the MCC environment was formed, which addresses issues such as battery capacity, processing power, memory, and speed [7]. MCC allows customers to utilize their mobile phones to access third-party cloud-hosted apps. The application's data storage and compute-intensive operations were shifted to the cloud [8]. Dropbox, Asana, and Apple's iCloud services are all instances of cloud-based mobile apps [9].

As per Mordor Intelligence, the worldwide mobile cloud business will be worth $118.70 billion by 2026 as mobile cloud computing capabilities become more prevalent. Mobile cloud ensures that mobile users have access to cloud-based data and mobile-specific apps and services. Gmail, Outlook, and Yahoo Mail are examples of mobile e-mail apps that save data using mobile cloud technology. Mobile social networking sites such as Twitter, Instagram, and Facebook enable real-time data sharing, allowing users to store data and share movies. MCC is used in mobile e-commerce apps to embrace scalable processing capability. MCC's mobile healthcare enables substantial volumes of real-time data storage in the cloud, ensuring rapid and easy

pervasive access to patient records. Other application areas include big healthcare data processing for smart city environments, IoT and cloud-based smart agriculture, IoT and cloud-based smart surveillance, context-aware services in the agriculture and healthcare sectors, and seamless service delivery in urban settings.

Mobile cloud computing offers the following benefits from a corporate perspective:

(i) It provides cloud services on the go.

(ii) It provides flexible services by allowing any time anywhere access via the Internet.

(iii) It alleviates mobile services through cloud resources.

(iv) It enables developers to design platform-independent mobile applications with cloud backend support.

(v) It allows both business and personal users to remotely access their documents, files, images, and other types of data via their smartphones over the Internet.

(vi) It enables mobile app companies and developers to access a bigger market.

However, due to the inherent nature of mobile clients, such as battery constraints, network technology, intermittent connectivity, and mobility, there are obstacles to achieving seamless service provisioning. This paper discusses the comprehensive literature survey on various resource provisioning techniques available in the literature. The key contributions to the field are as follows:

(i) Detailed background on mobile cloud environment and resource provisioning.

(ii) A brief description of the different architectures of MCC such as MEC and fog.

(iii) Comprehensive literature survey on RP techniques.

(iv) Overview of resource provisioning at edge cloud computing environment.

(v) Classification of RP schemes, performance metrics, and comparisons of state-of-the-art techniques.

(vi) Structured overview of future research directions in the RP area, both established and emerging.

The remainder of Section 1 summarizes the background of the filed. The rest of the paper is organized as follows. The mobile edge cloud computing environment is introduced in Section 2. The goal of various resource provisioning approaches is highlighted in Section 3. The recently proposed deadline-based resource provisioning techniques, context-based resource provisioning techniques, and cloudlet support resource provisioning approaches are discussed in Sections 4.1, 4.2, and 4.3, respectively. Various resource management-based resource provisioning approaches and energy-efficient resource provisioning techniques are discussed in Sections 4.4 and 4.5, respectively. The future research directions are presented in Section 5. The conclusions are summarized in Section 6.

1.1. MCC Service Models. The evolution of wireless technologies such as worldwide interoperability for microwave access (WiMAX), vehicle-to-everything (V2X) wireless, and software-defined radio (SDR), as well as telecommunication technologies such as fifth generation (5G) and the emerging sixth generation (6G), helped to grow the MCC [4]. In its service framework, the MCC distinguishes its service model based on the roles and relations between computational entities. Figure 1 shows the classification of current MCC service models.

Table 1 provides definitions for all the acronyms used in the paper. The first type is an ad hoc mobile cloud environment [7] where all mobile devices within the vicinity of the user act as cloud agents and thus allow the user to access the cloud services resident on the Internet. Also, the mobile devices nearby might have a set of services that could be used by other mobile devices, hence forming their local cloud which is known as cyber foraging. Ad hoc mobile cloud interstice characteristic is thus the mobility of the services [8]. The second type of MCC is an infrastructure-based model [9], and unlike the ad hoc mobile cloud, the services remain static. The infrastructure-based cloud provides services to mobile devices via the Internet. Thus, in the infrastructure-based MCC model, IaaS cloud has been deployed with virtualization concepts to deliver on-demand virtual machine (VM) resource computing, storage, and network bandwidth [10].

1.2. Infrastructure-Based MCC Model. Storage and computations occur beyond the mobile device and on the cloud infrastructure-based MCC. In general, the term MCC refers to the most widely used infrastructure-based service model [4, 5]. Figure 2 articulates a simplified MCC environment.

The basic operations of the MCC architecture are described below.

Smartphones, laptops, tablets, and other portable devices are the clients of MCC accessing the cloud server via Wi-Fi or long-term evolution (LTE) connections for accessing the computing, storage, and related services. The mobile client can request cloud services as a partial service client, which migrates only certain computations to the cloud, or as a straight thin client, which migrates all processing to the cloud [11]. Users of mobile devices can access cloud services directly through their mobile applications or through the cloud service provider (CSP) of their choice. Mobile client requests are represented in terms of processing speed, networking bandwidth, memory, and storage capacity. The subscribers' requests are delivered via the Internet to the cloud. The cloud controller entity processes the requests in the cloud to provide the appropriate cloud services to the users [12].

A smart mobile device accesses the cloud for getting services using any one of the following frameworks.

(i) *Indirect Request Frameworks.* When the mobile application was offloaded to the cloud, part or all of the application tasks were stored and processed in

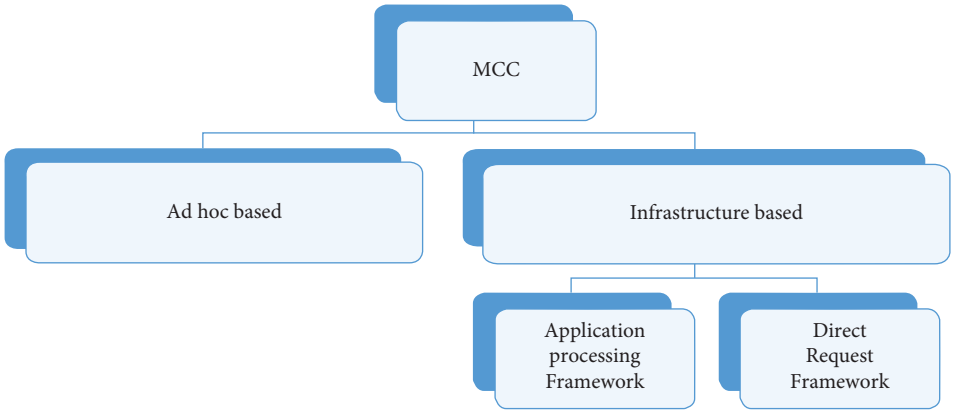

Figure 1: Classification of MCC service models.

Table 1: Abbreviations used in the paper.

Abbreviations	Description
BS	Base station
CPU	Central processing unit
CDC	Cloud data center
CSP	Cloud service provider
EERA	Energy-efficient resource allocation
ECM-RMA	Energy-efficient joint resource management and allocation
GPU	Graphics processing unit
ICT	Information and communication technology
IoT	Internet of Things
I/O	Input/output
IA	Independent authority
LA	Learning automata
LTE	Long-term evolution
MEC	Mobile edge computing
MCs	Mobile clients
MCC	Mobile cloud computing
NP-hard	Non-deterministic polynomial-time hardness
MuSIC	Mobility-aware service allocation on cloud
MAPE	Monitor-analysis-plan-execution
PRIMO	Prioritized metaheuristic virtual machine migration
RP	Resource provisioning
QoS	Quality of service
SUAC	Stochastic user allocation
SDN	Software-defined network
SLA	Service-level agreement
VM	Virtual machine
VIKOR	VIseKriterijumska optimizacija i kompromisno resen
WLAN	Wireless local area network
WORG	Weighted object relation graph
WMAN	Wireless metropolitan area network
Wi-Fi	Wireless fidelity
WiMAx	Worldwide interoperability for microwave access
5G	Fifth generation
6G	Sixth generation
V2X	Vehicle-to-everything

the cloud. The Application task processing in the cloud is made transparent to the user of the application and hence these requests are called indirect requests to the cloud [13]. The framework follows either static or dynamic application partitioning mechanisms based on the heaviness of the task, energy, and other context constraints. According to quality of service (QoS) requirements, application requests are translated to VM requests. The application-specific QoS requirements are transparent to the end-user. Therefore, these requests are called indirect requests.

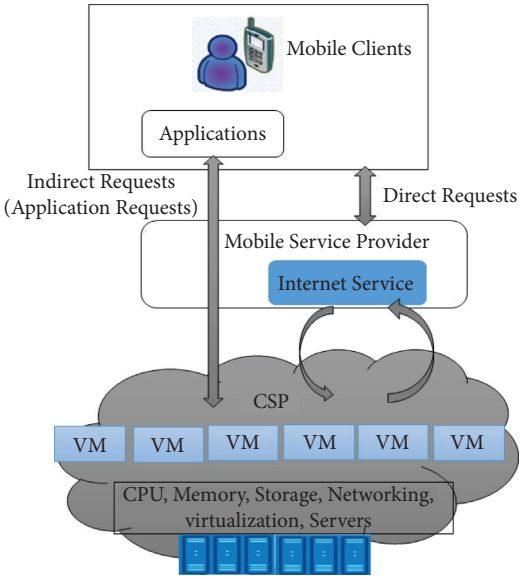

Figure 2: Simplified Infrastructure-based MCC environment.

(ii) *Direct Request Frameworks*. Direct IaaS requests can be sent directly to the cloud by a mobile subscriber. These requests are sent through the console of the service provider that can be easily accessed, viewed, and managed through a web-based user interface.

In the MCC service provisioning context, both direct and indirect service requests are translated in the form of VM resource requests and the VMs are allocated on the physical machine servers. In the IaaS cloud, a workload is a set of the task given by the users to the resources for execution. These workloads are computationally intensive tasks that need resources with high performance. Instead of owning these resources locally and maintaining them, the users rent the cloud resources on an hourly, monthly basis, or for a lifetime and pay as per the use. Resource management oversees all cloud computing events [14]. The resource manager is an entity that enables the CSP to programmatically manage resources in the cloud data center (CDC) [15]. Figure 3 describes the major resource management events.

The two primary tasks of resource management are resource provisioning and resource scheduling [10], and they are explained as follows. The process of discovering and allocating appropriate resources to the job request is called resource provisioning [16]. The assigning of resources to tasks and their execution is known as resource scheduling. Before resource scheduling, resource provisioning is typically done. The capacity of a system to flexibly expand or decrease in adapting to shifting workload requirements, such as a surge in web traffic, is referred to as the elasticity feature of resource scaling [10]. The resource scaling adjusts in real time to match the available resources to demand.

1.3. Motivation for Conducting the Survey. Mobile cloud computing makes essential cloud services accessible to customers and service providers on mobile devices. The

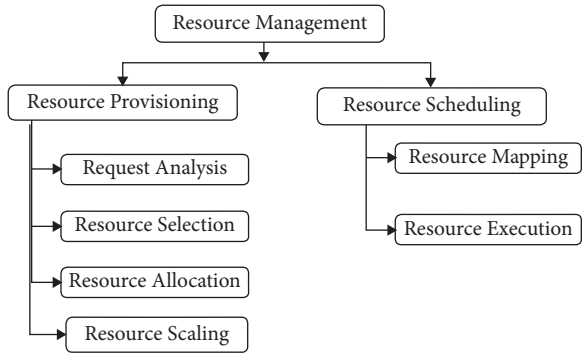

Figure 3: Resource management events.

ability of mobile devices to support the processing of memory-intensive applications with the help of resource-rich cloud servers is referred to as mobile cloud computing. Resource provisioning is used to boost cloud performance and make it possible to deliver the desired QoS. However, because of the fundamental nature of mobile clients, disconnections and limited battery life result in poor QoS [5, 14]. Also, mobile client requests arriving at the cloud system are normally high at peak hours, which demand efficient execution.

Concerning the current context of the mobile clients and the CSPs, the dynamic resource provisioning scheme must deal with the factors affecting the QoS. The existing nature of cloud resource provisioning lacks proper coordination between mobile clients and servers. Resource provisioning techniques should guarantee reduced SLA violations. The mobile client requests the CSP to deliver resources statically or dynamically based on the need [11, 17].

Resource under or over-utilization will occur in a static allocation of cloud resources. The mobile device is indeed given the resources, even when they are not being used. It reduces the mobile device's battery life and raises the cost of the service. Therefore, in mobile cloud computing, dynamic resource allocation based on client-side as well as CSP-side context characteristics is critical. Grounded on these issues, effective resource provisioning for mobile clients is critical. Resource provisioning is the process of identifying and providing suitable resources for the request.

Experts from several notable institutions began developing appropriate resource provisioning strategies for mobile application offloading in the early 2010s. Figure 4 depicts the growth of resource provisioning techniques. After accepting the requests, the service provider prepares and allocates the VM based on the resource requirements. Because mobile users have battery constraints and less capability to supply bandwidth and memory, the availability of cloud services is a challenge [18]. Resource provisioning improves the performance of the mobile cloud through the scheduling of requests, discovering suitable resources, and selecting and mapping the requests with a resource. Several resource provisioning research studies have been carried out to meet the quality of service (QoS) requirements, such as CPU usage, availability, reliability, security, and so on for mobile customer requests.

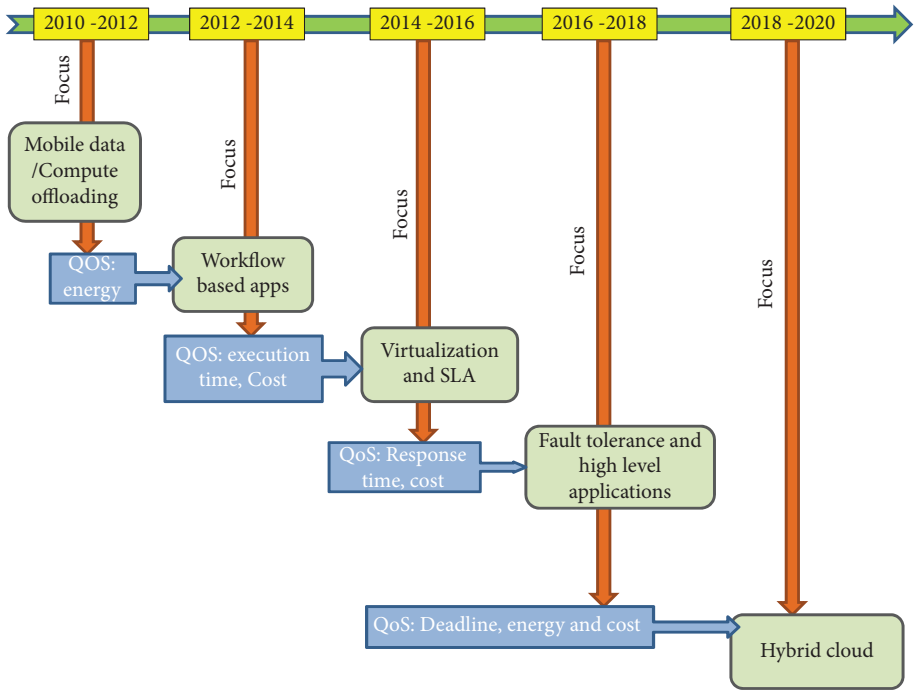

Figure 4: Evolution of resource provisioning techniques.

2. A Brief Description of Different Architectures of MCC

MEC (multi-access edge computing) is a networked topology standard for delivering cloud computing and IT services at the edge of cellular networks and, more broadly, any network [14]. MEC shifts data and application processing from a central server to the network's periphery, bringing it closer to the client. Rather than transmitting all data to the cloud for processing, the network edge analyzes, processes, and stores it locally. Bringing data collection and processing closer to the customer minimizes delay and provides higher app real-time performance.

MEC is composed of a set of services that run at the network's edge and conduct operations with the least amount of latency. Cellular base stations, server farms, Wi-Fi gateways, and hotspots serve as the network's edge. The edge's proximity decreases delay and ensures that consumers have stable connectivity.

An edge computer connects to a device's sensors and controllers before sending data to the cloud. This data traffic is huge and inefficient. MEC is an edge computing design specification, whereas fog computing widens the network's edge. A processing layer between the cloud and the edge is known as fog computing. Before it reaches the cloud, fog analyzes and filters only the essential data [19, 20]. Relevant data are retained in the cloud, while irrelevant data are removed or evaluated at the fog layer for distant access [16].

MCC and MEC have been promoted in recent years as a way to expand the versatility of smart mobile devices. Transferring to a traditional cloud, on the other hand, causes major execution delays, which are cumbersome in near-real-time applications. The solution to this problem has been suggested as MEC [21–23]. With the help of mobile edge computing, the capabilities of the network are expanded by bringing cloud computing to the edge. MEC is an ETSI (European Telecommunications Standards Institute) initiative that was initially intended to be used to deploy edge nodes on mobile networks but has subsequently been expanded to encompass fixed convergent networks. Unlike conventional cloud computing, which takes place on remote servers far away from the user and system, MEC enables processes to be carried out in base stations, central offices, and local aggregation sites [24]. The overview of resource provisioning in the MEC environment is shown in Figure 5.

The generic components of the overall MEC design include the access layer, MEC layer, and cloud layer which are described below.

(i) *Access Layer*. An access layer's main job is to grant users network access. Access network gateways connect to edge layer switches/routers to perform computational and/or storage-intensive task processing. An access layer device allows users to link to the network, whether it has an Ethernet connection to each end station or a local access server. Typically, these devices are smartphones, tablets, laptops, and sensors such as temperature, humidity, smart watch, and smart wrist bands deployed in IoT networks. Access network gateways are used to transform massive volumes of data processing requests into digital streams, which can then be processed. Short-range and long-range smart communication devices are included in the access gateway. It transports requests from the access layer to the MEC processing layer over networks like Wi-Fi, 3G, Bluetooth, RFID, and LoRa.

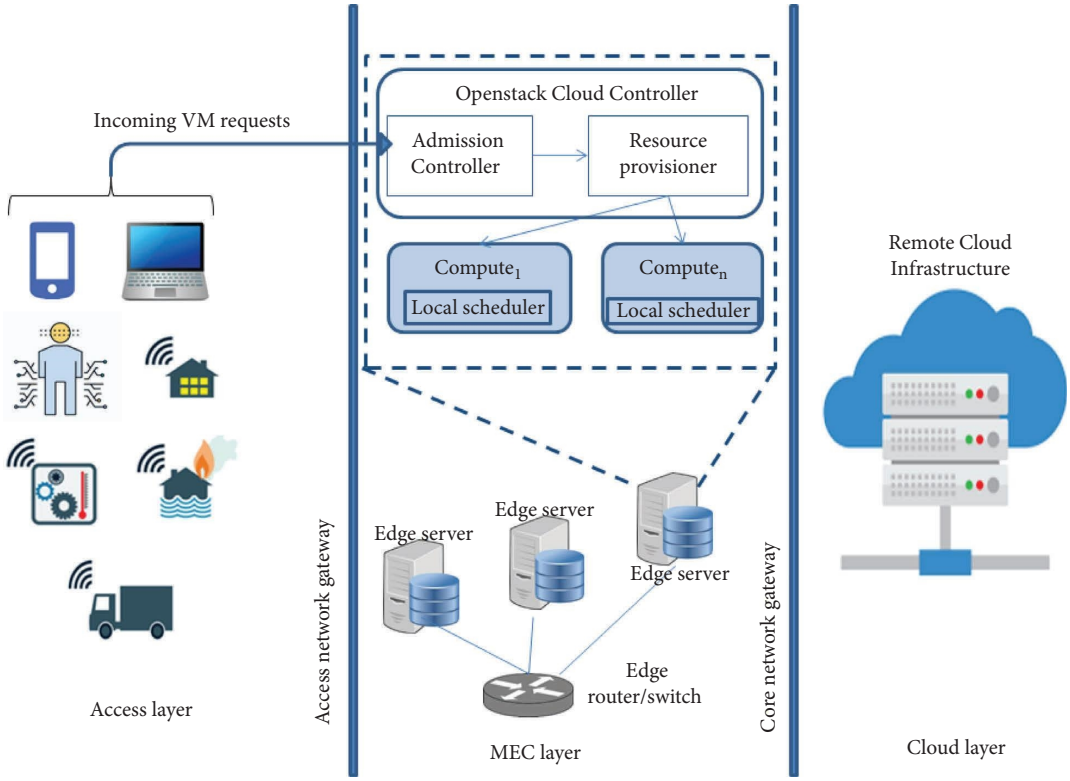

Figure 5: Overview of resource provisioning at edge cloud computing environment.

(ii) *MEC Layer*. The mobile edge layer consists of several mobile edge servers that provide a virtualization infrastructure for mobile clients including computing, storage, and network services [25]. The MEC servers are placed in the wireless access network, near the mobile clients. As discussed in Section 2, mobile requests have a range of QoS specifications including requests that are both delay-sensitive and delay-tolerant. Edge hosts handle these requests, whereas cloud instances that are dynamically tenanted manage outsourced delay-tolerant requests.

(iii) *Cloud Layer*. Unlike a private cloud strategy, which requires administrators to handle the environment individually, remote public clouds depend on a single plane of management [26]. The number of computing jobs conducted in the cloud is normally significantly more than the amount of computing work performed on each MEC server, and their delay tolerance is usually lower. MEC servers must finish received computing jobs in a short time frame to provide QoS, while the cloud must complete received computing duties as rapidly as feasible. The MEC servers and the cloud are linked via wired core networks, and they can exchange computational power according to their requirements.

3. Resource Provisioning in MCC: Background

Cloud provides resources such as GPU/CPU, storage, memory, and networking as a service to the customers via the resource provisioning model [27]. Cloud services are considered a major component of the cloud in terms of how a client obtains resources from the CSP.

In the MCC, resource provisioning (RP) is defined as the selection, deployment, and runtime management of hardware resources (e.g., processor, memory, storage, and network) to ensure guaranteed performance. Figure 6 further shows the steps involved in resource provisioning. Resource provisioning enables virtualized resources to be allocated to the users based on the demand. The provision of resources takes into account the SLA for the provision of services for cloud users. An SLA is an agreement between a service provider and a user that specifies the level of service that the user expects from the provider. It ensures QoS parameters such as response time, minimum delay, availability, cost, and reliability [28].

Resource provisioning is done in three phases, such as (1) request analysis, (2) resource selection, and (3) resource allocation. These phases are described as follows:

(i) *Request Analysis*. Initially, the cloud subscriber submits the service request to the request analyzer for analyzing the resource requirements. Based on its requirements, the request is translated into the VM allocation request.

(ii) *Resource Selection*. The resource provisioning technique quantifies and selects suitable resources in terms of VM and physical machine (PM) for a specific service request based on its QoS requirements.

(iii) *Resource Allocation*. The resource provision technique provides a bundle of resources, e.g., the VM which consists of certain virtual CPU, memory, and

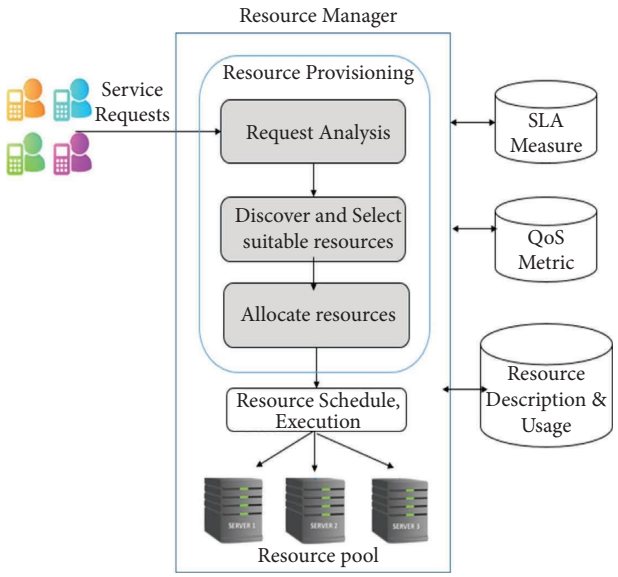

Figure 6: Resource provisioning steps.

storage is provided according to the user requirements. SLA violations can occur from any flaws in resource management. SLA, for example, creates target dates based on obligations. When a target date is missed, the SLA is broken, resulting in low customer satisfaction and, as a result, a loss of money and customers.

3.1. Benefits of Resource Provisioning Technique. CDC adopts resource provisioning to achieve a better QoS. The benefits of adopting resource provisioning techniques are as follows:

(i) The CSP can quickly submit a variety of requests on-demand using resource provisioning techniques.

(ii) The resource provisioning technique reduces the requirement for human intervention in the provision and management of computing resources.

(iii) The resource provisioning technique enables the cloud supply model to scale the cloud resources up and down based on short-term usage requirements.

(iv) The resource provisioning technique can locate and allocate suitable services in the federated cloud.

(v) It eliminates large upfront investments and hence provides potential cost savings.

3.2. Resource Provisioning Technique. Resource provisioning is performed transparently to the users running their applications in the cloud. In general, resource provisioning is performed using one of three basic techniques, namely, static resource provisioning, dynamic resource provisioning, and self-service provisioning, and they are described as follows:

(i) *Static Resource Provisioning.* Static provisioning is applied when the demand for applications is predictable and constant. The resources are reserved based on the contract between the CSP and the customer. Further, the CSP prepares appropriate resources ahead of the service. A one-time fee or a monthly fee will be invoiced to the client.

(ii) *Dynamic Resource provisioning.* Dynamic resource provisioning is applied when the demand for applications is unpredictable and varies dynamically. The CSP can virtually add or remove hardware resources on-demand with dynamic resource provisioning. The users are charged based on the number of times they use the service. Dynamic resource provisioning is used for the application that runs in a private cloud and breaks into a public cloud when demand for computing capacity increases. Cloud bursting is the term for this deployment approach.

(iii) *Self-Service Provisioning.* Self-service provisioning is a dynamic provisioning method that allows anyone to customize and launch apps and services in the cloud without the intervention of an IT company or a service provider. This offers customers greater flexibility in how they use services, as long as they stay within the supplier's boundaries [10].

3.3. Research Methodology. Properly answering significant research questions is made possible by the research methodology. Creating a research question is the first stage in conducting an effective and worthwhile study. Several researchers developed their study questions using the specific framework known as population, intervention, comparison, and outcome (PICO), which also makes it easier to review the literature [29]. Search terms were constructed using keywords and are shown in Table 2. In digital libraries, keywords are used to find relevant study content.

The most reliable digital databases that have undergone technical and scientific peer review have been used to find articles, journals, conference papers, and workshop papers. Commonly used digital libraries are shown in Table 2.

3.4. Performance Metrics. The following crucial performance indicators are used to measure the performance of resource provisioning: mean service response time, percentage of requests meeting the deadline, system throughput, and percentage of rejected requests [30]. The mean service response time is calculated by averaging the time between the end of a request and the start of a response. The percentage of requests that reach the due is calculated by multiplying the number of requests that meet the deadline by the total number of requests filed in a unit of time multiplied by a hundred. System throughput is a metric that measures the number of requests that are fulfilled in a certain amount of time. The percentage of rejected requests is calculated by dividing the number of requests that were rejected by the total number of requests that were submitted.

3.5. Resource Provisioning in MEC Environment. It is undeniably challenging to provide a resource in any kind of cloud environment. Although setting up a new request on a

Table 2: Research questions and search keywords.

Research questions	Search keywords	Article selection	Digital libraries
Why is there an urge to migrate to edge cloud?	Need, benefits, requirements, motivation, MEC migration	Survey	(i) ACM Digital Library
What are the challenges in MEC?	Challenges, issues, mobile edge cloud, mobility	Survey	(ii) Springer Link (iii) ScienceDirect
What are the existing solutions or tools for MEC resource provisioning?	Tools, solutions, framework, process	Survey, research manuscript	(iv) IEEE Xplore (v) Google Scholar
What is the role of 5G in MEC?	5G and MEC, issues, need of 5G and MEC	Survey, research manuscript	(vi) Scopus (vii) Wiley
What distinguishes MEC, fog, and cloud computing from one another?	Differences, compare, fog, MEC, cloud	Survey, research manuscript	(i) Compendex (ii) Whitepaper (iii) Hindawi

cloud and ensuring a reasonable share of resources is complicated, it is currently well known. The problem becomes more complex in a hybrid cloud environment, where services must be provisioned both at the edge and in the cloud. Because the computing power of edge hosts is fixed and limited, the resource provisioning problem is a critical issue in MEC [25, 31]. Open-source cloud computing software like Apache CloudStack and OpenStack has been described as one of the solutions for deploying and managing virtual machine networks in a highly accessible and scalable environment. Figure 6 depicts the MEC infrastructure implemented using OpenStack tools. The MEC cloud controller is responsible for resource identification, monitoring, allocation, task migration, failure management, and network management. The admission controller intercepts the request and analyzes the resource requirements of each of the incoming requests. Based on its requirements, the request is translated into the VM allocation request. The resource provisioning technique offers a package of resources, such as a virtual machine (VM) with specific vCPU, memory, and storage, depending on the user's needs. Any errors in resource management can result in a breach of the SLA [10]. The resource provisioner quantifies and selects suitable resources in terms of VM and allocates the resources on the computing (compute) machine for processing the requests based on the request's QoS requirement. When the demand for computing resources is very high, the edge server will offload compute-intensive requests to a resource-rich remote cloud. However, if the MEC servers have a surplus of processing power, they might sell some of it to the cloud to boost their profitability [21].

4. Categorization of Resource Provisioning Techniques

In general, resource provisioning techniques have been broadly classified into static and dynamic resource provisioning techniques. The goal of resource provisioning (RP) techniques for mobile clients in cloud environments is to ensure that the cloud service is adequately resourced as client demand grows. It is also necessary to ensure that the application requirements are correctly met. Hence, this study focuses on the categorization of RP techniques. It helps the researchers toward understanding the objectives and diverse requirements of offering services.

Based on their objectives, the resource provisioning techniques have been further classified into five categories, namely, deadline-based resource provisioning techniques, context-based resource provisioning techniques, cloudlet support resource provisioning techniques, resource management-based resource provisioning techniques, and energy-efficient resource provisioning techniques.

Various deadline-based techniques, context-based techniques, and cloudlet support techniques for mobile clients have been studied and analyzed using the parameters, namely, request analysis process, resource selection, resource allocation, performance metrics, and its limitations. They are discussed in Sections 4.1, 4.2, and 4.3, respectively. Various resource management-based resource provisioning techniques have been studied and analyzed using parameter resource management mechanisms, performance metrics, and limitations. They are discussed in Section 4.4. The review has been carried out on energy-efficient resource provisioning techniques using essential parameters, namely, improvement in resource provisioning, performance metrics, and limitations, and they are all discussed in Section 4.5. Table 3 shows the objective of five categories of resource provisioning techniques.

The underlying processes, such as request analysis, resource discovery, resource allocation, and management mechanisms, are used to evaluate resource provisioning techniques. Table 4 shows a description of these critical processes.

The commonly used performance evaluation metrics of resource provisioning are discussed under Section 3.4. Additionally, Table 5 highlights the matrices more specific to each of the five categories of resource providing. Table 5 presents case studies of resource provisioning methods to serve as an inspiration for the MEC, fog, and cloud research journeys.

4.1. Deadline-Based Resource Provisioning Technique. The deadline is the time limit for executing applications and is thus defined in the service-level agreement (SLA). The deadline and an estimate of the execution time of each application task are provided by the user during the submission of a job. The predominant intention of deadline-based resource provisioning techniques is to allocate cloud resources to the tenants for meeting the deadlines of these applications.

Table 3: Categorization of resource provisioning techniques.

Resource provisioning techniques	The objective of the techniques
Deadline-based resource provisioning techniques	Provisioning of resources to requests which have the least time (earliest deadline) to complete to ensure on-time service
Context-based resource provisioning techniques	Resource provisioning based on the current context of the clients and the CDC to avoid service failure
Cloudlet support resource provisioning techniques	Provisioning of resources from the servers available within the proximity of the user to reduce response delay
Resource management-based resource provisioning techniques	Provisioning of resources to request with consideration of load balancing to avoid over or under-utilization of resources at servers
Energy-efficient resource provisioning techniques	To improve the QoS, provisioning is carried out by considering parameters, namely, energy efficiency, cost, and so on

Table 4: Description of the critical processes.

Process and mechanisms	Description
Request analysis process	Analyze the request based on its requirements to translate it into the VM allocation request
Resource selection process	Quantify and select the resources according to the request
Resource allocation mechanism	Affirm the SLA before allocating the resources needed to fulfill the request
Resource management mechanism	Efficient and effective deployment of resources

Li et al. proposed DCloud [32], the deadline-aware resource allocation algorithm. The technique requires that a user define both the resource needed and the deadline. The quantity of VMs and the associated bandwidth are used to calculate the necessary resource. DCloud allows dynamic adjustments in VM launching time and bandwidth according to the deadline constraint. This algorithm helps to balance the cloud resource's peak demand and thus reduces the request rejection rate. The allocation is done based on the deadline and load conditions of the cloud but is insufficient to allocate resources to mobile customers.

Durga et al. proposed the energy-efficient deadline-based task scheduling model [33] for mobile cloud computing. The model focuses on the partitioning of mobile applications into compute-intensive and non-compute-intensive tasks. Based on the mobile device characteristics such as the energy level and wireless connection type, the compute-intensive tasks were scheduled to be performed in the cloud. This model considered task dependency, data transmission time, deadline, and cost to solve an energy consumption optimization problem. The challenge of energy minimization was resolved using a genetic algorithm. However, mobility and the server's current context have not been examined.

Islam et al. proposed a lightweight resource allocation framework for big data applications in Apache Spark [34]. A master node manages the resources of one or more worker nodes. The master node decides and allocates the resources to the application request. This framework used the deadline requirement and the predicted task completion time to select the appropriate resources. The prediction method was only based on the profile made from the past application runs. The load and resource availability parameters of the CDC were not considered to predict the execution time, thereby affecting the performance of the framework. In addition to the insufficient parameters that affect the execution time prediction, the algorithm was not considering the mobile client characteristics.

An energy-efficient scheduling policy [21] is presented by Ma et al. for cloud-assisted mobile edge computing applications. They designed an algorithm for scheduling the tasks to be carried out in the cloud. The decision to offload the code was taken to achieve minimal energy consumption on the mobile device while meeting the deadline. The mobile application tasks were represented in a directed acyclic graph. The task scheduling problem was defined as a constrained shortest path problem and was solved by using the Lagrangian relaxation-based algorithm. The experiment of this method considered only the application deadline for making resource allocation decisions.

Chang et al. proposed deadline and cost-based workflow scheduling in a hybrid cloud [35]. The combination of private and public cloud was used according to the demand of the workflow. Cost optimization was done while allocating resources for a job completed within the deadline. They also considered level-based scheduling to find suitable resources for cost savings and complete the execution within the deadline. Their scheduling approach attempted to save the cost significantly by maintaining deadlines.

Praveen et al. proposed a cost-efficient resource provisioning approach [36]. A novel autoscaling mechanism was proposed to finish all the jobs before the user-specified deadline at minimum cost. The jobs were specified by workflows and quantified as VM type. Resource provisioning was done in three steps. (1) Instance type for each incoming job was identified. (2) The number of instances for each type was determined. (3) The earliest deadline first algorithm was used to schedule jobs on each VM. The approach was evaluated in terms of deadline miss rate and cost. This work focused solely on deadline-based resource allocation and does not address the load conditions of the server.

A data-aware resource provisioning technique using the Aneka platform has been proposed by Tuli et al. [37]. The algorithm suggested to satisfy user-defined deadline criteria

Table 5: Performance metrics and case studies.

Resource provisioning techniques	Case studies	Performance metrics
Deadline-based resource provisioning techniques	Big healthcare data processing for smart city environments using MEC/fog	(i) Average service response time (ii) Deadline meeting percentage (iii) Throughput (iv) Average no. of rejected requests (v) Resource utilization (vi) Load to the resource (VM, host, or CDC) (vii) Delay/latency in network (viii) No. of provisioned resource
Context-based resource provisioning techniques		(i) Average service response time (ii) Deadline meeting percentage (iii) Percentage of rejected requests (iv) Energy consumption (v) Current context analysis time (vi) Availability and trustworthiness (vii) Security (viii) Privacy
Cloudlet support resource provisioning techniques	(i) Context-aware services in the agriculture and healthcare sectors using edge and fog computing (ii) Seamless service delivery in urban settings using cloud (iii) IoT and cloud-based smart agriculture using edge cloud (iv) IoT and cloud-based smart surveillance using edge cloud	(i) Edge node/cloudlet lifetime (ii) Customer mobility speed (iii) Network speed (iv) Mean service response time (v) Percentage of requests meeting its deadline (vi) Percentage of rejected requests (vii) Energy consumption (viii) Scheduling time (ix) Completion ratio (x) Cost/profit
Resource management-based resource provisioning techniques		(i) Resource utilization (ii) System load (iii) Energy consumption (iv) Scheduling time (v) Completion ratio (vi) Cost/profit (vii) Fault tolerance (viii) Under and over-utilized resources (ix) SLA violation ratio
Energy-efficient resource provisioning techniques	Resource provisioning method considering both client and server's energy context	(i) Energy consumption (ii) Mobile device battery context (iii) Scheduling time (iv) Completion ratio (v) Cost/profit (vi) Edge node/cloudlet lifetime (vii) Customer mobility speed (viii) Service/resource discovery time (ix) Under and over-utilized resources

for data-intensive applications. Parallel jobs make up the workload for the program. Their system determines the extra resources required to finish application activities by the deadline by taking into account variables like data location, cloud startup time, network bandwidth, and data transfer time. The average runtime of the public cloud resources was measured separately by the authors, who took into account the bandwidth's capacity for data transmission time. This work did not mention how to approximate a job's execution time.

Durga et al. proposed the resource provisioning technique based on the application's QoS constraints [38]. Memory, CPU, and data transmission requirements determined the type of resource requests. The scheduling problem was formulated based on a binary integer program. This technique is dependent on the estimated execution time to ensure QoS. Although this technique leads to improved performance in terms of reduced delay and cost, ignoring the specific characteristics of mobile requests leads to SLA violations in the MCC domain.

Tuli et al. presented a deadline-driven cloud provisioning algorithm in the hybrid cloud [37]. The incoming request was described as a scientific workflow of similar tasks. The execution of the pending tasks can be achieved with the average execution time and the number of resources used. If the estimated completion time of a job is longer than the remaining time of a job, more resources have been added to perform the job. The resource release mechanism was also proposed to meet the job completion. The result showed that the provided resources increased with a tighter deadline.

Nayak and Tripathy proposed a VIKOR-based task scheduling algorithm [30]. The conflict among similar kinds of tasks was resolved by the VIKOR decision maker. The rank of the tasks has been computed by taking into account two criteria, such as the execution time and the execution deadline. The ideal task among similar backfilled tasks has been found using the VIKOR method. The VMs were given ideal resources based on the deadline. This work provides better resource utilization and reduces task rejection, but the algorithm did not mention how to approximate response time.

A dynamic strategy was developed by Alsadie et al. for distributing resources and addressing QoS restrictions like the deadline and the budget [39]. Both task scheduling and resource provisioning problems were addressed. The resource provisioning algorithm was based on resource utilization. The algorithm allocates fixed resources initially and adjusts the number of resources to the extent that the application uses them. There is a consideration for uniform resource distribution and budget in this approach; however, most of the important parameters in the mobile cloud environment such as deadline violation and job completion rate were not optimized. Additionally, a single type of VM is considered a resource.

Lu et al. presented a deadline-constrained heuristic for scheduling cloud-based scientific applications [40]. The resource type and the variance in VM performance are the two parameters considered when modelling resource provision as an optimization issue. Particle swarm optimization was used to solve and produce a schedule of the task to the resource. The algorithm was designed to meet the deadlines at a lower cost. An integer number was used to indicate the resource index for scheduling the resources to the workflow. The resource index, however, does not accurately depict the properties of the resources. Thus, randomizing the particle movement could be accomplished by learning from the resource index.

Verma and Kaushal presented a genetic algorithm-based workflow scheduling technique [41]. Deadline and budget have been considered for making a scheduling decision. It achieves reduced cost while completing the task execution before its deadline. The major issue with their heuristic algorithm is the mono-objective property of the problem. Table 6 shows the comparison of the state-of-the-art deadline-based resource provisioning techniques.

4.2. Context-Based Resource Provisioning Technique. Context-based resource provisioning aims at providing personalization and customization of cloud services based on mobile cloud system contexts. Mobile device profile, environmental information, and request requirements are considered the context information in the mobile cloud environment. The mobile device context includes a client ID, device energy, location, time, wireless connection details, and so on. Integrating data security to the process of resource provisioning increases the security of cloud data and mitigates the risks involved [45–48]. For the cloud data center, the context could be defined by utilization percentage, load, and resource availability. This section reviews several resource provisioning techniques which make use of the current context of the mobile cloud system to achieve the desired QoS level. The primary contributions, advantages, and disadvantages of existing context-based resource providing strategies are summarized in Table 7.

MuSIC, a mobility-aware cloud service allocation framework [49], had been developed to find the best set of services based on the location and time workflow of the mobile user. A heuristic algorithm based on simulated annealing was used to consider user mobility while allocating resources. The workflow depends on the mobility pattern of each user. The user mobility pattern was supplied by a set of user location tuples and the amount of time the user spends at each location. Two parameters such as power and delay have been used to evaluate the system. Two types of connections, Wi-Fi and 3G, have been considered. An allocation decision was made using the power consumed by the mobile device when connecting to the cloud. However, the time delay for allocating resources has not been taken into account, which leads to a violation of the SLA.

A container-based platform called Rattrap [67] is employed for mobile cloud offloading. This platform contains a cloud Android container to replace VMs for mobile computing offloading. The offloading has been enhanced by scheduling resources at the process level rather than at the VM level. The operating systems with different kernel features run inside containers with driver extensions. Hence, the limitation of virtualization at the OS level has been partially removed. According to the evaluation, the Rattrap enhances offloading efficiency by speeding up the initialization of the runtime environment. There is a consideration for cloud server-side contexts in this approach; however, mobile client-side context information such as mobility, energy, and job completion rate has not been measured.

Lee et al. proposed a code-offloading architecture for native applications in the MCC environment [51]. The offloading architecture allows computer-intensive tasks to be performed on remote servers to speed up execution and

Table 6: Comparison of the deadline-based resource provisioning techniques.

References	Resource provisioning techniques used	Major contribution	Pros	Cons
Shahidinejad and Ghobaei-arani [31]	DCloud: deadline-aware resource allocation algorithm	Dynamic adjustments in VM resource allocation	Dynamic resource allocation according to the deadline constraint	Allocating resources to mobile clients is insufficient
Li et al. [32]	Energy-efficient deadline-based task scheduling	Compute-intensive jobs and non-compute-intensive tasks were scheduled in the cloud based on mobile device variables such as battery level and wireless connection type	Cuckoo search-based optimization algorithm was used to solve the NP-hard problem	The client's mobility and the server's current context have not been considered while evaluating the performance
Durga et al. [33]	Prediction-based resource provisioning	Lightweight resource allocation framework	A profile created from a previous program run was used to predict the outcome	The CDC's load and resource availability factors were not taken into account while predicting the execution time
Chang et al. [35]	Level-based scheduling to find the suitable resources for cost savings and complete the execution within the deadline	Time and budget-aware scheduling algorithm for hybrid cloud	Cost optimization for resources allocated within the deadline	Other mobile client characteristics are not considered
Praveen et al. [36]	Deadline and cost-based workflow scheduling	Cost optimization was done while allocating resources for a job	Level-based scheduling to find suitable resources for cost saving	The case of resource contention was not considered
Tuli et al. [37]	Data-aware resource provisioning algorithm using Aneka	Deadline specifications for applications requiring a lot of data	The mean runtime of public cloud services is computed based on available bandwidth	Does not indicate how to estimate the execution duration of a job
Nayak and Tripathy [30]	VIKOR-based task scheduling algorithm	VIKOR decision maker used to schedule similar tasks	Better resource utilization and reduces task rejection	Does not consider how to approximate response time
Malawski et al. [42]	Resource provisioning and task scheduling algorithm based on resource utilization	It coordinates the scheduling of related jobs and settles disputes between them while allocating resources	(i) Allocates fixed resources initially and adjusts the number of resources to the need of applications (ii) Considers uniform resource distribution and budget in this approach	Deadline violation and job completion rate were not optimized
Nayak and Tripathy [30]	Genetic algorithm-based workflow scheduling	Deadline and budget have been considered for making a scheduling decision	It achieves a lower cost while completing the task ahead of schedule	The problem's single objective characteristic is the main flaw in their heuristic technique
Alsadie et al. [39]	DTFA: a dynamic threshold-based fuzzy approach	Allows dynamic adjustments in VM launching time and bandwidth	Balances the cloud resource's peak demand and thus reduces the request rejection rate	Insufficient to allocate resources to mobile customers
Lu et al. [40]	Energy-efficient scheduling algorithm	Energy-efficient scheduling policy for cloud-assisted mobile computing applications	Offloads the code to achieve minimal energy consumption	Application deadline alone is considered for resource allocation decisions
Shahidinejad et al. [17]	Novel autoscaling mechanism	Cost-efficient resource provisioning approach	Completes all the jobs within the deadline at minimum cost	Does not address the load conditions of the server
Nadjaran Toosi et al. [43]	Data-aware resource provisioning algorithm	Data-intensive apps with user-defined deadlines	Data location, cloud startup time, network bandwidth, and data transfer time are considered	Does not consider job execution time
Lai et al. [44]	Optimized stochastic user allocation (SUAC) algorithm	Optimized allocation of user's multi-dimensional resource requirements	Effective and stable system for multi-criteria user requirements	Allocation for massive real-world requirements to be considered

Table 7: Comparison of the context-based resource provisioning techniques.

References	Resource provisioning techniques used	Major contribution	Pros	Cons
Zhang et al. [49]	Combined genetic algorithm and simulated annealing algorithm-based method	Mobility-aware cloud service allocation framework	Allocation decision was made using the power consumed by the mobile device when connecting to the cloud	Time delay for allocating resources not considered, thus leading to SLA violation
Manukumar [50]	Agent-based offloading decision maker	The decision maker chooses the compute component that runs on the cloud and mobile sides	Adaptable for bandwidth fluctuation applications	The context of the mobile requests has not been considered
Lee et al. [51]	Computer-intensive tasks to speed up execution and improve performance are utilized	Code-offloading architecture for native applications in the MCC environment	Native offloader improves execution speed	Optimized cross-platform translation of SIMD instructions is necessary for native offloading frameworks for multimedia applications
Ascigil et al. [52]	Function-as-a-Service (FaaS) for resource allocation	Application providers install their latency-critical processes that handle user requests with constrained turnaround times	Delay tolerance threshold of the user considered for offloading	The decision to offload is considered only based on power consumption
Nawrocki et al. [53]	Context-aware resource allocation using supervised learning agent architecture and service selection algorithm	Minimizes the cost while meeting mobile client requests' deadlines	Cost-efficient scheduling based on energy context and Internet connection type	Impact of device mobility not analyzed
Farahbakhsh et al. [54]	Context-aware resource allocation	The monitor-analysis-plan-execution (MAPE) cycle is used to collect and analyze the circumstances before making decisions on offloading	Faster provisioning of dynamic resources	Does not provide better QoS for mobile customers due to the dynamic environmental changes
Chase and Niyato [55]	Optimal solution using deterministic equivalent formulations	Resource provisioning technique with joint optimization of VM and bandwidth reservation	Cost-efficient	Random network delays and VM migrations deviate the pricing
Mireslami et al. [56]	Branch-and-bound technique to obtain realistic discrete solutions	Cloud resource allocation problem with concurrent cost and QoS optimization	Provides optimized resource allocation	Handles workloads from only the web
Midya et al. [57]	Hybrid particle swarm optimization	A three-tier architecture made up of the local cloudlet, the centralized cloud, and the mobile cloud	QoS achieved	Slow convergence time
Chunlin and Layuan [58]	Lagrangian multiplier method	Multiple context-based service scheduling	More parameters are considered and thus there is improvement in mobile user's QoS experiences	Impact of device mobility not analyzed
Quian and Andresen [59]	Offloading computations to resourceful servers	Energy-aware computation offloading is supported by the Jade runtime engine for smartphone platforms	Improves the performance of mobile applications and lowers energy usage	Compatibility issues for non-Android users
Niu et al. [60]	Bandwidth partitioning scheme based on weighted object relation graphs (WORGs)	Bandwidth-adaptive application partitioning algorithms and optimization models	Reduced energy use and enhanced performance with bandwidth adjustability	Requires an optimal solution for large-scale real-time applications
Zhou et al. [61]	Deadline-based resource provisioning	Mobile cloud offloading framework considering the user and cloud contexts	Reduced energy consumption for migrating apps	Applicability is for specific applications and single user
Durga et al. [62]	Context-aware resource allocation	Optimized allocation of resources based on cuckoo optimization	Minimized cost while meeting mobile client requests' deadlines	Device mobility is not considered
Naqvi et al. [63]	Context-aware and cloud-based resource allocation	Visual augmented reality techniques	Lower latency and reduced memory load	Cost benefits yet to be analyzed

Table 7: Continued.

References	Resource provisioning techniques used	Major contribution	Pros	Cons
Naha et al. [64]	Dynamic context resource allocation	Resource ranking based on the constraints for the fog-cloud environment	Reduced processing time and cost	Failure handling to be included
Durga et al. [65]	Optimal resource allocation for balancing the costs and benefits of mobile users and cloud servers	Cuckoo search-based optimization algorithm and a context-aware cloud resource management algorithm	Improved QoS	Fog and edge devices can be utilized for load balancing
Spatharakis et al. [66]	Resource profiling mechanism	Two-level edge computing architecture with location estimation technique and scaling and allocating resource techniques	Increased performance by considering resource under-utilization and QoS criteria	Accuracy to be improved and can be extended for time-specific applications
Jia et al. [67]	QoS-aware cloudlet load balancing	Fast heuristic algorithm and distributed genetic algorithm	Minimizes the maximum task response time	Increased execution time

improve performance. The state-of-the-art on-demand resource provisioning approach was employed to provide cloud resources. Instead of virtualization and code annotation, this architecture relies on a VM front-end compiler for intermediate representation (IR). A unique virtual address is assigned to the mobile device and the server to share memory objects efficiently by the compiler, which also handles translation codes. The four phases for compiling IR binaries during runtime are target selection, target-specific optimizations, memory unification code generation, and application partitioning. Additionally, this design uses the copy-on-demand technique to move live programs from a mobile device to the server. The experimental finding demonstrates that the native offloader achieves efficiency in execution time and smartphone energy in all states.

Phone2cloud [52] is a framework for computation offloading from smartphones to raise energy efficiency and performance levels. The authors described a deadline-based resource provisioning approach. The average execution time was compared to the user's criterion for delay tolerance. The application task will be offloaded to the cloud if the threshold is lower than the standard execution time. Otherwise, the the decision engine will check to see whether the power used to operate the application in the cloud is more than the power used to run it locally on the device. Executing the application on the cloud uses less energy compared to the location execution.

A supervised learning agent architecture and service selection technique for context-aware resource allocation was introduced by Andrew and Kathrine et al. in 2021 [47]. The deadline for mobile customer requests was a major consideration. However, the current mobile device and mobility context information has not been examined. Farahbakhsh et al. proposed context-aware resource allocation in 2021 [54]. The monitor-analysis-plan-execution (MAPE) cycle has been used to collect and evaluate the contexts to make offloading decisions for quicker provisioning of dynamic resources. However, it does not provide a higher QoS for mobile users due to dynamic changes in the environment.

Chase and Niyato presented the resource provisioning technique with joint optimization of VM and bandwidth reservation [55]. Based on past demand, the CSP is used to analyze deterministic equivalent formulations to find an optimum solution. The price sensitivity analysis was applied to measure cost efficiency. Factors such as random network delays and VM migration are ignored. Due to unpredictable network latency and VM migrations, it is infeasible to collect the internal cost data for cloud networking.

Mireslami et al. [56] solved the simultaneous cost and QoS optimization problem for cloud resource allocation using the branch-and-bound method. Their method provides the best resource combination while taking the needs of the customer into consideration, but it is only applicable to web workloads. Midya et al. [57] proposed the multi-objective optimization algorithm. They suggested a three-tier architecture made up of a centralized cloud, neighborhood cloudlets, and a mobile cloud. Their algorithm was designed using hybrid particle swarm optimization to efficiently handle multiple requests from on-road users while maintaining the QoS. However, the performance of the architecture was affected due to the slow convergence time of the hybrid algorithm.

Chunlin and Layuan proposed a cost and energy-aware cloud service provisioning model [58] for mobile clients. The model emphasizes how cloud service providers' services can be composed to optimize the overall system utility in terms of cost, energy, and revenue. The algorithm was involved in the optimization of the CSP and the optimization of the mobile cloud user by two routines. The Lagrangian method was applied to solve the optimization problem. Energy consumption, user budget, allocation efficiency, and execution success ratio were considered as a quality of service parameters. The main limitation of the model is that the model has not considered user mobility and intermittent connection during resource allocation.

The application code is analyzed and partitioned in the Jade system [59]. The authors of Jade used a technique where the status of application and device such as communication cost, workload variation, and energy has been considered to

partition the application code at the class level. An offloading decision could be taken based on the device status and therefore encourages energy efficiency in mobile devices. The developer could control the partitioning as the system supported Android and non-Android servers (Windows/Linux). However, non-Android servers need Java installed to use Jade.

By dividing mobile apps using the branch-and-bound and min-cut-based techniques, Niu, Song, and Atiquzzaman sought to increase execution performance [59]. Examples of this are application object profiling and static analysis. A weighted object relationship graph (WORG) has been created to represent the objects and their relationships. The branch-and-bound method ensures optimal partitioning outcomes for small applications, whereas the min-cut method works well for large applications. A program has been divided into client-side and server-side execution codes using the bandwidth parameter and WORG.

mCloud [61] is a mobile cloud offloading framework that includes application virtualization, code partitioning, and migration. The network characteristics, CPU speed, and energy consumption were the kinds of context information considered to decide on partitioning. The mobile application is divided by moving a thread at a chosen location from the mobile device to the cloud clone. The authors recommended a deadline-based resource provisioning technique for running the code on the cloud. This framework attempted to reintegrate the migrated thread back into the mobile device. The cost model data for offloading under various execution settings are obtained from execution traces. It provides performance enhancement and energy savings for migrated applications.

Durga et al. presented the framework for context-aware resource allocation using cuckoo optimization [62]. The resource provisioning technique enables the optimum allocation of resources for certain tasks promptly to achieve the desired service quality. The optimization models for resource allocation involve how to best make use of available resources subject to certain constraints to optimize certain objective functions. The framework monitors the use of energy context and Internet connection type to make task scheduling and decision on resource allocation. The main aim was to decrease the cost while meeting mobile client requests' deadlines. The impact of device mobility on the proposed algorithms was not analyzed. Durga et al. offered a model for context-aware service delivery in the Platform-as-a-Service layer [62] and explained how to deliver context-aware cloud services to mobile cloud clients.

Naha et al. [64] presented dynamic resource provisioning algorithm for a fog-cloud environment. The dynamic changes in the behaviour of the users are investigated. The findings demonstrate that the proposed resource and latency-aware algorithm reduces the processing time, delay, and cost.

Multiple context-based optimal resource provisioning was proposed for balancing the cost and benefits of mobile users and cloud servers [65]. The authors proposed the resource provisioning model comprising two levels. In the first level, a cuckoo search-based optimization algorithm is used to optimize the cost, mobile client waiting time, and cloud server utilization. To increase the system's performance, a context-aware cloud resource management algorithm is proposed in the second level. Under this model, the authors claim the improvement in mobile users' QoS experiences by considering multiple system parameters.

Durga et al. [65] addressed the scalability issues of the massive heterogeneous environment. The authors proposed the design and evaluation of scalable two-tier edge computing architecture for resource profiling in the context of the smart city. The cost effectiveness and efficient system utilization improve the overall system's performance. Spatharakis et al. [66] suggested the QoS-aware load balancing of cloudlets in WMAN with two effective techniques for distributing the workload across cloudlets to reduce the maximum task response time [68]. Experimental simulation results showed that the proposed methods performed well and showed promise.

4.3. Cloudlet Support Resource Provisioning Technique. The long distance between a mobile client and its associated CDC results in high latency that in turn increases the overall execution time. A cloudlet is proposed to alleviate this problem [69]. Cloudlet is a resource-rich server located at the edge of the network. The main objective is to support interactive and resource-intensive mobile applications by giving mobile devices with lower latency powerful computational resources [44, 69]. This section studies various resource provisioning techniques in cloudlet support environments.

The process of leveraging external resources to increase the capabilities of the resource-limited mobile device is called cyber foraging. There are two kinds of applications that run on the mobile devices, namely, offloading applications and thin client-server applications [4–6, 70], and they are described as follows:

(i) *Offloading Applications*. The application is partitioned into compute-intensive and non-compute-intensive parts. The dynamic decision is made to offload the compute-intensive part to the nearby cloudlet.

(ii) *Thin Client-Server Applications*. These are the applications whose entire processing happens at a cloudlet. The application is statically partitioned as a thin client and computation parts. The thin client version of the application runs on a mobile device.

Alam et al. and Sivan and Sellappa presented an overview of various mechanisms of mobility augmented service provisioning in the MCC environment [71, 72]. The arrival of mobile client requests is unpredictable due to their mobility feature. The mobility augmented services support constantly moving wireless devices that access cloud service seamlessly and extensively. Echeverría et al. proposed a green cloudlet architecture in the context of mobile cloud computing [73]. The SDN-based network was included in the architecture to provide efficient connections between various entities. Cloudlet network file system has also been proposed to ensure failure recovery.

Islam et al. [74] proposed cloudlet-based cyber foraging for mobile devices in resource-constrained edge contexts. The application is independent of the mobile device due to the cloudlet's ability to replicate operating system capabilities. The cloudlet deploys the program and notifies the client that the application server is ready to use. The application package and metadata are the first things sent from the mobile device to the cloudlet. The cloudlet provisioning was done through VM synthesis. Then, the server component was installed on a base VM image to make it ready for application task execution. A provisioning script is sent from the mobile device to the cloudlet at runtime. By running the provisioning script, the cloudlet creates and launches a suitable VM. This technique has provided effective offloading capabilities at the edge level. However, the architecture lacks in addressing the disconnected operations which is crucial.

On-demand VM provisioning for the cyber foraging environment has been presented to provide the services using an on-demand micro-data center connected to a network [73]. Their mechanism provisioned a service VM at runtime by leveraging the advantage of enterprise provisioning tools. When a cloudlet receives a request with baseline VM metadata, it finds a matching baseline VM on the cloudlet. Once the match is found, the cloudlet manager creates a new service VM on the identified baseline VM. The proposed VM provisioning framework was only applicable for application task execution and was not suitable for another type of request.

Sun and Ansari [75] proposed a hierarchical paradigm comprised of cloudlet facility and mobile device infrastructures. Cloudlet coverage zones are typically small due to their reliance on Wi-Fi availability. The primary goal of the proposed model is to provide more coverage to mobile consumers. Collaboration across multiple cloudlets is required to overcome a single cloudlet's limited ability to meet user requirements. The authors studied how to manage large-scale applications across multiple cloudlets with resources. However, the complexity of resource management for different requests was not studied. Also, the dynamic nature of the mobile client requests was not considered during the resource provisioning process.

Islam et al. proposed an ant colony optimization-based joint VM migration model for a cloudlet-based MCC environment [74]. The prioritized metaheuristic virtual machine migration (PRIMO) algorithm was designed to migrate resources among the set of cloudlets. This model focused on reducing the time required to complete the task and reducing the over-provisioning of resources in mobile cloud computing. The PRIMIO paradigm considers a cloudlet's mobility and computational load while migrating VMs. Minimizing overall execution time and minimizing resource wastage are the two main goals of the optimization problem. Ant colony optimization was applied to solve the problem.

Tawalbeh et al. attempted to enhance the performance with local cloud server (cloudlet) [76]. An incentive system has been proposed to coordinate the resource auctions between mobile devices and cloudlets. Performance analysis demonstrated that the mechanism attains truthfulness for both clients and servers. But the price is being fixed between the client and the server and will not be changed based on the demand and supply of these, which in turn affect the MCC system efficiency. Table 8 compares the state-of-the-art cloudlet-based resource provisioning techniques.

4.4. Resource Management-Based Resource Provisioning Technique. Park et al. proposed two-phase resource management for big data processing in ad hoc-based MCC environment [79]. Mobile cloud application users are experiencing poor QoS in terms of latency and poor resource utilization. This work overcomes the problem using a grouping strategy based on utilization and movement rates. The cutoff point strategy based on entropy levels has been employed for separating the mobile devices' group. They also proposed a two-phase method of grouping to reduce group management overhead. During the first phase, grouping of mobile devices was done according to the information collected from them. Concerning n cutoff points, the total number of groups formed is $(n+1)^2$. The overhead depends upon the increase in the number of cutoff points with the number of groups. In the second phase, this challenge was addressed. The second phase includes similar groups to keep the number of groups manageable.

An autonomic resource management system (ARMS) for cloud and computing has been presented to handle job scheduling and resource allocation automatically [80]. The resource manager handles the allocation of resources and task planning processes on mobile nodes. The time, cost, and energy needed to complete the task execution were estimated by the cloud manager, and accordingly, the resource allocation decision was made. The work focused only on the efficient allocation of resources. But the factors such as load, utilization, and mobility of devices affecting the resource management were not considered.

Si et al. proposed QoS-aware resource management for heterogeneous MCC networks [80]. Cloud resource management and cross-network radio were proposed to increase tenant revenue while maintaining the required QoS. This work focused on the problem of restless bandits and guarantees for low complexity and scalable and distributed feature. The request reaches the cloud via mobile access networks, after which decisions were taken on the management of radio resources and cloud computing resources. This management process was not appropriate for the pragmatic nature of MCC applications.

Durga et al. proposed a two-stage task and resource management model for cloud media computing requests [33]. A queue-based task management approach and a heuristic-based resource management approach are proposed. Each task has a deadline as part of its SLA. The optimization problem to minimize the waiting time and cost was formulated. A queue and Markov analysis-based task management algorithm has been proposed to solve the problem.

A cooperative resource management framework for the MCC environment has been proposed to address the

Table 8: Comparison of the cloudlet support resource provisioning techniques.

References	Resource provisioning techniques used	Major contribution	Pros	Cons
Sun and Ansari [75]	Green cloudlet-based workload offloading	Cloudlet network file system and a green energy supplement are proposed	Low end-to-end delay and saves energy	Have to minimize the operating expenditure of the cloudlet providers
Lewis et al. [77]	Resource-intensive computation is offloaded to cloudlets	Cloudlet-based cyber foraging	Enhanced offloading capabilities and computation	Lacks in addressing the disconnected operations
Echeverría et al. [73]	Cloudlet-based cyber foraging discovering nearby resource-rich nodes	On-demand VM provisioning	Flexibility, energy consumption, maintainability	Applicable for application task execution and not suitable for other request types
Jararweh et al. [78]	Hierarchical cloudlet model	Software-defined system for MEC (SDMEC)	Increased coverage area	Dynamicity and resource management for different requests not addressed
Islam et al. [74]	Migrating resources among the set of cloudlets	Prioritized metaheuristic virtual machine migration (PRIMO) algorithm	Optimal solution for gathering all the resources for users	Mobility and context-aware task computation and execution time to be considered
Tawalbeh et al. [76]	Enhanced performance of local cloudlet	A truthful incentive mechanism	Truthfulness for both clients and servers	Fixed cost for all demands

resource management and sharing problem [81]. As computation offloading is involved with communication and computing elements, cooperation between cells is essential for the most effective distribution of radio access and computer requests. This framework enabled cooperation at the radio level and at the application level to reduce interference and implement distributed cloud capability. Both communication resources and computer resources have been considered for offloading decisions. However, this work focuses only on the decision making for resource coalition to increase the revenue of service providers.

Liu et al. proposed a hierarchical architecture for cloud-based vehicular networks that allows vehicles to share computational, storage, and network resources [82]. The resource management problem was formulated to address the resource competition between VMs. A game-theoretical approach was used to resolve this issue. A resource reservation strategy was used to solve virtual resource migration caused by vehicle mobility. Performance results showed that there was a significant reduction in the service dropping rate. However, other factors affecting the resource allocation such as intermittent connections, resource availability, cost, and system utilization were not considered. The comparison of the state-of-the-art resource management-based resource provisioning techniques is shown in Table 9.

4.5. Energy-Efficient Resource Provisioning Technique. Karamoozian et al. proposed QoS-aware resource allocation for media services in the MCC environment using the learning automata (LA) technique [84]. This technique includes three parameters, namely, the total response time of various services, uncertainty level which is measured by susceptibility to failure level, and computational resource requirements of the requests. The LA technique was used to reward or penalize the VM for each specific media service. The queue model was applied to analyze the incoming requests. The aim was to select an optimal resource based on its performance factor for a particular VM configuration. Response time and failure rate were the two performance metrics considered. But there are a few other factors such as SLA violations, system throughput, and energy that are not considered to ensure QoS.

An energy-efficient joint resource management and allocation (ECM-RMA) approach is provided to lessen time-averaged energy usage in a multi-user multi-task mobile cloud environment to address the resource provisioning dilemma [85]. Performance evaluations have been done based on the arrival rate, the number of cloud nodes, and the data size to prove the QoS. However, the effect of device mobility to ensure QoS was not considered for the proposed service allocation approach.

Singh and Chana proposed Q-aware: QoS-based resource provisioning in cloud computing [85]. Cloud workloads were evaluated and clustered using workload patterns. There was a unique set of QoS metrics for each task. It investigated how workload and resource density affected execution time and cost. Results showed a significant time and cost reduction while achieving the deadline. Q-aware was not able to characterize the mobile client requests.

Hassan proposed a cost-effective provisioning scheme for the multimedia cloud environment [86]. The time limit for the multimedia requests was taken into consideration for resource provisioning. In addition to that, the cost-efficient resource allocation and management focused on the Nash bargaining solution. The results showed the efficiency of the bargaining algorithm in terms of utilization, reduced migrations, and the number of active servers. Their algorithm assumed, however, that the system already knows the execution time information, which is not always true. An efficient model to estimate the execution time of the incoming request is crucial.

To predict the arrival of VM requests, an integrated energy-aware resource provisioning [87] system for cloud data centers has been suggested by Dabbagh et al. This research also forecasts the amount of CPU and memory

Table 9: Comparison of the resource management-based resource provisioning techniques.

References	Resource provisioning techniques used	Major contribution	Pros	Cons
Park et al. [79]	Two-phase resource management-grouping technique	Grouping by cutoff points based on entropy values	Reduced overhead	Factors related to edge and fog context requests are not considered
Tadakamala and Menasce [83]	Autonomic resource management system	Task scheduling and resource allocation in a dynamic mobile environment	Reduced cost and energy	Factors such as load, utilization, and mobility of devices are not considered
Si et al. 2014 [80]	QoS-aware resource management for heterogeneous MCC networks	Cross-network radio and cloud resource management scheme	QoS requirements satisfied	Not appropriate for the pragmatic nature
Yu et al. [81]	Cooperative resource management	Computation offloading framework	Reduced interference and had distributed cloud capability	Focuses only on the decision making for resources coalition
Liu et al. [82]	Hierarchical architecture for cloud-based vehicular networks	Game-theoretical framework	Reduced service dropping rate	Factors affecting the resource allocation such as intermittent connections, resource availability, and cost are not considered

resources required for each of these requests, as well as reliable estimates of the number of physical machines (PMs) required by cloud data centers to support their clients. The authors created a greedy heuristic method for resource prediction that employs simulated annealing to produce a solution that is close to optimal. Two types of applications, such as heavy computing and streaming, are used to test this technique. However, if Wi-Fi networks are used locally, scaling issues arise due to the increasing number of users, latency, packet loss, and reliability of application.

Mitra et al. proposed a novel mobility management system called the M^2C^2 for mobile cloud computing [88]. The M^2C^2 system supports three mechanisms, namely, (1) multi-homing, (2) cloud and network probing, and (3) cloud and network selection. The system allows users to roam seamlessly in heterogeneous access networks (HANs) while accessing local and public cloud resources. Probing mechanisms were used to provide awareness of the network and cloud QoS to the mobile nodes when it roams in HANs. It supports multi-homing and includes cloud and network selection metrics based on user applications and network and cloud load. For validating the M^2C^2, the authors developed a system prototype and performed extensive experimentation. Zhang et al. [89] proposed a resource allocation model based on an auction mechanism with premium and discount factors. The buyers and sellers participate in the auction. Cloud resources are divided into multiple groups (e.g., processing, storage, and communication). The combinatorial auction mechanism is used to formulate the resource allocation problem. This mechanism does not fit the cloudlet architecture due to the problem of allocating M resources to N users in one MCC service provider.

Sood and Sandhu presented the proactive resource provisioning model to estimate the number of resources needed for a mobile client with an independent authority [90]. IA scans the mobile device operating system, the number and nature of the installed applications, and the network status to allocate cloud resources initially. The authors predicted resource usage using a two-dimensional resource provisioning matrix. An IA used this matrix with an artificial neural network to forecast future resource requirements. The emphasis was on the details gathered from the mobile user and its precision and not on any other QoS parameter. In addition to this, the processing time and the communication cost involved with the independent authority were not considered.

Energy-efficient green solution [91] has been proposed by Din et al. for hierarchical resource management in MCC environment. This research described the energy resource allocation problem as NP-hard and implemented a hierarchical system architecture based on 5G constraints. The foglet layer, service layer, and communication layer are the three phases of hierarchical device architecture. The foglet layers allow efficient resource sharing, while the service layers aid in this process.

EERA [92], an energy-efficient resource allocation strategy for mobile cloud workflows, has been proposed by Li et al. To illustrate the trade-off between energy usage and execution time, researchers developed the concept of utility cost. An optimization model is used to solve the resource allocation problem, aiming to reduce utility costs while meeting the execution time constraints of mobile cloud workflow applications. EERA achieved the possible balance between energy consumption and QoS for mobile cloud workflow applications. However, the effect of dynamic network bandwidth on data communication time was not addressed.

Zhang et al. [93] created an energy-efficient resource allocation method for a multi-user mobile edge computing system. With negligible and non-negligible base station (BS) execution durations, the best resource allocation is achieved. To solve the NP-hard problem, Johnson's algorithm was used.

Guo et al. [94] proposed the ENERDGE framework, which addresses full task offloading and resource allocation issues in a multi-site setting. Task offloading is considered for applications with various characteristics and requirements. In terms of latency and energy consumption, an optimal resource allocation framework that incorporates edge resources outperforms traditional load-balancing techniques. Table 10 compares the approaches to energy-efficient resource provisioning.

5. Future Research Directions in Resource Provisioning Technique

Cloud providers are quickly integrating pay-per-use resource provisioning for mobile consumers as technology advances. However, there are still some obstacles to overcome in this field. The rapid growth of mobile Internet technologies has placed severe demands on MCC infrastructure, which has led to new investigations for solving unsolved issues in the area. The key takeaways of the future research include the following.

(i) The intrinsic characteristics such as battery life, disconnections, and mobility of the mobile clients lead thus to delayed service that in turn ends up in deadline violations. Also, the occasional workload burst causes sudden peaked computation demand and accordingly extends the response time. Hence, an appropriate technique is required to consider both the mobile device context and SLA while dynamically configuring the resource. Multiple contextual data are expected to be exploited to give tailored and customized services to mobile clients and to improve the overall experience. Device mobility induces a location change. The mobility feature of the client decreases the performance gain of using cloud resources through high communication latency. This degrades the QoS and may even result in the loss of user access. Therefore, the MCC desires a time-efficient resource provisioning scheme based on the proximity of the user to the data center.

(ii) The effective synchronization between mobile clients and servers plays a vital role that is currently lacking in resource provisioning solutions [84]. This causes the resources to remain allocated even when the client is unable to receive results from the cloud which in turn increases the cost. These techniques consider only the resource provisioning benefits either on the CSP side or on the client side. Hence, knowledge of the mobile client, as well as CSP states, is needed for efficient resource provisioning [96]. Location and capacity limits are influencing the direction of the proposed strategies [86], as existing solutions target single cloud providers rather than federations of cloud providers that allow strategic pooling of resources. The resource provisioning techniques may be extended to support federated mobile cloud environments [26].

(iii) Mobile cloud services are accomplished through centralized processing at the cloud data center. Communication latency is caused by the vast distance between the cloud and the mobile device [87]. The current technological shift enables mobile devices to experience real-time services with edge and fog computing. The resource provisioning algorithms can be extended for IoT Cloud, edge, and fog computing environments to achieve reliable resource allocation [65].

(iv) MEC authorizes many forms of access near the edge. Beyond mobile, Wi-Fi and stable network technologies also improve the outcome. It entails moving computing and storage capabilities closer to the consumer. MEC is a crucial 5G technology driver that improves the quality of content distribution and services. MEC shortens the network path, which lowers latency, increases reliability, and improves total network efficiency. Organizations may gather and handle enormous amounts of real-time data to optimize various operational systems by utilizing the 5G and edge computing pairing. Without edge computing, user devices would be transmitting data straight to the cloud. As a result, the bandwidth needed to transmit data to the cloud would increase dramatically, negating the benefits of a 5G network. Businesses can achieve better performance for mission-critical applications by performing processing and analysis close to where the data originated. Edge computing enables 5G viable when working with millions of connected devices. The widespread adoption of 5G and MEC installation has brought about new difficulties. The expense and overhead of deployment have been raised by placing edge compute servers at large numbers of base station sites. As 5G and MEC-based applications proliferate and the security risk grows, protecting the privacy of the user is another issue. 5G will need to define these uncertainties to address the dangers to global security, such as those affecting trust, privacy, and cyber security. Deploying edge compute servers with multiple energy feeds aids in providing resilience if one feed fails. However, it is crucial to obtain it depending on the size and location of the edge servers [97].

(v) Researchers have identified the following additional challenges in the MEC environment. The deployment of MEC systems presents difficulties such as server capacity management, network infrastructure, and location identification for MEC servers [98]. Data caching has potential research directions, such as MEC data analytics and service cache for MEC resource allocation. Due to the requirement for mobility-aware offloading and mobility-aware server scheduling, mobility management for MEC is also a significant challenge. Deploying a greener MEC environment with MEC systems fuelled by sustainable power is yet to be solved.

Table 10: Comparison of the energy-efficient resource provisioning techniques.

References	Resource provisioning techniques used	Major contribution	Pros	Cons
Karamoozian et al. [84]	Learning automata (LA) to reward or penalize the VM	QoS-aware resource allocation for media services	Optimal resource selection based on response time and failure rate	SLA violations, system throughput, and energy are not considered to ensure QoS
Zhang et al. [93]	Energy-efficient joint resource management and allocation (ECM-RMA) policy	Reduced time-averaged energy consumption in a multi-user multi-task in MCC	Improved QoS performance	Effect of device mobility to ensure QoS was not considered
Singh and Chana [85]	Q-aware: QoS-based resource provisioning	Analyzed cloud workloads and clustered using workload patterns	Significant reduction in cost and time	Unable to characterize the mobile client requests
Hassan et al. [86]	Cost-effective provisioning scheme for the multimedia cloud environment	Resource allocation and management based on the Nash bargaining solution	Efficient system in terms of utilization and reduced migrations	Execution time of the incoming request to be considered
Dabbagh et al. [87]	Integrated energy-aware resource provisioning	A greedy heuristic approach for generating a near-optimal solution using a simulated annealing technique	Effective for intensive computing and streaming	Scalability issues
Mitra et al. [88]	Mobility management system (M^2C^2)	Probing mechanisms	Supports mobility efficiently	All QoS metrics to be considered
Zhang et al. [89]	The resource allocation model is based on an auction mechanism	Combinatorial auction mechanism with substitutable or complementary commodities	Individual rational and incentive compatible	Does not fit the cloudlet architecture
Sood and Sandhu [90]	Proactive resource provisioning model	Independent authority to predict the future required resources using an artificial neural network	Achieves mobile user details and precision	Independent authority's processing time and the communication cost are not considered
Din et al. [91]	Energy-efficient green solution	Hierarchical resource management based on novel 5G system architecture	Energy-efficient communication related to cost and time	Balancing the uneven energy consumption and traffic distribution required
Li and Xu [92]	Workflow can be executed by either the mobile device locally or the cloud server via computation offloading	Energy-efficient resource allocation algorithm (EERA)	Improved data communication time	Effects of dynamic bandwidth to be incorporated for a robust and adaptive system
Guo et al. [94]	Energy-efficient mobile edge computing systems	Computation-efficient models with a negligible and non-negligible base station	Optimally allocate the communication and computation resources	Cost ineffective for a large-scale geographic area
Avgeris et al. [95]	Efficient allocation of resources for the offloaded tasks from the mobile devices to the edge	Optimal resource allocation framework	Robust task offloading solution	Errors in dynamically estimated positions and end-user device numbers must be reduced to a minimum

Machine intelligence improves the efficiency of the mobile cloud system through workload prediction and resource allocation [99]. The resource provisioning techniques will be improved by incorporating artificial intelligence [100]. Federated learning equips edge devices with cutting-edge machine learning without centralizing data or jeopardizing privacy. Managing data transfer from devices to edges, edge resource provisioning, and federated learning between edges and the cloud is a critical task when operating federated learning optimally over distributed cloud-edge networks [101]. It is necessary to specifically design programs for MEC to operate at the edge. Applications for the edge must be built natively or migrated from cloud environments to the edge. The platform and its development suite combine third-party services for delivering apps through application programming interfaces (APIs). ClearBlade offers open-source software for edge computing for flexible IoT services. The Eclipse Foundation and IBM jointly created an open-source platform called Eclipse ioFog. Azure private MEC platform promises a combination of network operations, applications, and edge-optimized Azure services to enable high-performance solutions that satisfy enterprise customers' contemporary business needs.

6. Conclusions

This paper presented an overview of current resource provisioning approaches for mobile and cloud computing. Firstly, we begin with an introduction to the motivation for MCC and the taxonomy of MCC's resource provisioning

method presented based on five different perceptions: (1) deadline-based, (2) context-based, (3) cloudlet support, (4) resource management-based, and (5) energy-efficient resource provisioning techniques. Then, depending on distinct aspects of the resource provisioning mechanism, the taxonomy of each perspective is offered. Each category of resource provisioning techniques has been presented and their performances are compared and evaluated using vital parameters. Further, a survey on resource provisioning techniques is carried out and classification is mapped to the key characteristics. It is observed that very few resource provisioning algorithms concentrate on mobile client requests. It is evident from the literature that many factors are affecting the efficient provisioning of resources to mobile clients in the cloud environment. These factors are grouped according to the benefit level of the user and the server. They are the (1) mobile client side and (2) cloud provider side. Internet connection type, battery status, mobility, deadline, resource requirements such as RAM, CPU, storage, and network requirements, and other SLA requirements are important factors affecting the provision of resources on the part of the mobile client. The resource availability, load, utilization, and SLA requirements are important factors affecting the provision of resources on the part of the cloud provider. We summarized the research gaps in the existing literature and highlighted the future research directions.

Data Availability

No data were used to support this study.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the Japan Society for the Promotion of Science, Grants-in-Aid for Scientific Research [JP21H03496, JP22K12157]; the Japan Science and Technology Agency, Precursory Research for Embryonic Science and Technology [JPMJPR1934].

References

[1] S. Sundareswaran, A. Squicciarini, and D. Lin, "Ensuring distributed accountability for data sharing in the cloud," *IEEE Transactions on Dependable and Secure Computing*, vol. 9, no. 4, pp. 556–568, 2012.

[2] X. Lan, Y. Zhang, J. Ren, L. Cai, J. Ren, and L. Cai, "Efficient computing resource sharing for mobile edge-cloud computing networks," *IEEE/ACM Transactions on Networking*, vol. 28, no. 3, pp. 1227–1240, 2020.

[3] R. Y. Ma, W. Liang, J. Li, X. Jia, and S. Guo, *World Journal of Hepatology*, vol. 12, no. 1, pp. 1–5, 2020.

[4] S. Durga, S. Surya, and E. Daniel, "SmartMobiCam: towards a new paradigm for leveraging smartphone cameras and IaaS cloud for smart city video surveillance," in *Proceedings of the 2018 2nd International Conference on Trends in Electronics and Informatics (ICOEI)*, pp. 1035–1038, Tirunelveli, India, May 2018.

[5] S. Durga and S. Mohan, "Mobile cloud media computing applications: a survey," *Lect. Notes Electr. Eng.,* vol. 222, pp. 619–628, 2013.

[6] A. M. Rahmani, M. Mohammadi, A. H. Mohammed et al., "Towards data and computation offloading in mobile cloud computing: taxonomy, overview, and future directions," *Wireless Personal Communications*, vol. 119, no. 1, pp. 147–185, 2021.

[7] K. Muralidhar and K. Madhavi, "Setting up ad hoc computing as a service in mobile ad hoc cloud computing environment," *International Journal of Interdisciplinary Telecommunications and Networking*, vol. 13, no. 1, pp. 1–12, 2021.

[8] I. Yaqoob, E. Ahmed, A. Gani, S. Mokhtar, and M. Imran, "Heterogeneity-aware task allocation in mobile Ad Hoc cloud," *IEEE Access*, vol. 5, pp. 1779–1795, 2017.

[9] D. Martin-Sacristan, S. Roger, D. Garcia-Roger et al., "Low-latency infrastructure-based cellular V2V communications for multi-operator environments with regional split," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 2, pp. 1052–1067, Feb, 2021.

[10] A. Shahidinejad, M. Ghobaei-Arani, and L. Esmaeili, "An elastic controller using Colored Petri Nets in cloud computing environment," *Cluster Computing*, vol. 23, no. 2, pp. 1045–1071, 2020.

[11] M. Ghobaei-Arani and A. Shahidinejad, "An efficient resource provisioning approach for analyzing cloud workloads: a metaheuristic-based clustering approach," *The Journal of Supercomputing*, vol. 77, no. 1, pp. 711–750, 2021.

[12] A. Hussain and J. Chun, "Cloud service scrutinization and selection framework (C3SF): a novel unified approach to cloud service selection with consensus," *Information Sciences*, vol. 586, pp. 155–175, 2022.

[13] K. Akherfi, M. Gerndt, and H. Harroud, "Mobile cloud computing for computation offloading: issues and challenges," *Applied Computing and Informatics*, vol. 14, no. 1, pp. 1–16, 2018.

[14] A. Shakarami, H. Shakarami, M. Ghobaei-Arani, E. Nikougoftar, and M. Faraji-Mehmandar, "Resource provisioning in edge/fog computing: a Comprehensive and Systematic Review," *Journal of Systems Architecture*, vol. 122, Article ID 102362, 2022.

[15] O. Gireesha, N. Somu, and K. Krithivasan, "IIVIFS-WASPAS: an integrated Multi-Criteria Decision-Making perspective for cloud service provider selection," *Future Generation Computer Systems*, vol. 103, pp. 91–110, 2020.

[16] M. Etemadi, M. Ghobaei-Arani, and A. Shahidinejad, "A learning-based resource provisioning approach in the fog computing environment," *Journal of Experimental & Theoretical Artificial Intelligence*, vol. 33, no. 6, pp. 1033–1056, 2021.

[17] A. Shahidinejad, M. Ghobaei-Arani, and M. Masdari, "Resource provisioning using workload clustering in cloud computing environment: a hybrid approach," *Cluster Computing*, vol. 24, no. 1, pp. 319–342, 2021.

[18] M. Ghobaei-Arani, R. Khorsand, and M. Ramezanpour, "An autonomous resource provisioning framework for massively multiplayer online games in cloud environment," *Journal of Network and Computer Applications*, vol. 142, pp. 76–97, 2019.

[19] M. Etemadi, M. Ghobaei-Arani, and A. Shahidinejad, "Resource provisioning for IoT services in the fog computing environment: an autonomic approach," *Computer Communications*, vol. 161, pp. 109–131, 2020.

[20] M. Etemadi, M. Ghobaei-Arani, and A. Shahidinejad, "A cost-efficient auto-scaling mechanism for IoT applications in fog computing environment: a deep learning-based approach," *Cluster Computing*, vol. 24, no. 4, pp. 3277–3292, 2021.

[21] X. Ma, S. Wang, S. Zhang, P. Yang, C. Lin, and X. Shen, "Cost-efficient resource provisioning for dynamic requests in cloud assisted mobile edge computing," *IEEE Trans. Cloud Comput.*, vol. 9, no. 3, pp. 968–980, 2021.

[22] T. L. Duc, R. G. Leiva, P. Casari, and P. O. Östberg, "Machine learning methods for reliable resource provisioning in edge-cloud computing: a survey," *ACM Computing Surveys*, vol. 52, no. 5, pp. 1–39, 2020.

[23] A. Bozorgchenani, F. Mashhadi, D. Tarchi, and S. A. Salinas Monroy, "Multi-objective computation sharing in energy and delay constrained mobile edge computing environments," *IEEE Transactions on Mobile Computing*, vol. 20, no. 10, pp. 2992–3005, 2021.

[24] S. Member, "Mobile edge cloud system: architectures," *Challenges, and Approaches*, vol. 12, no. 3, pp. 2495–2508, 2018.

[25] G. Tang, D. Guo, K. Wu, F. Liu, and Y. Qin, "QoS guaranteed edge cloud resource provisioning for vehicle fleets," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 6, pp. 5889–5900, 2020.

[26] R. Zeng, S. Zhang, J. Wang, and X. Chu, "FMore: An Incentive Scheme of Multi-Dimensional Auction for Federated Learning in MEC," in *Proceedings of the 2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS)*, Singapore, Singapore, November 2020.

[27] T. Cao, Y. Jin, X. Hu et al., "Adaptive provisioning for mobile cloud gaming at edges," *Computer Networks*, vol. 205, Article ID 108704, 2022.

[28] E. Badidi, Y. Atif, Q. Z. Sheng, and M. Maheswaran, "On personalized cloud service provisioning for mobile users using adaptive and context-aware service composition," *Computing*, vol. 101, no. 4, pp. 291–318, 2019.

[29] M. Saaiq and B. Ashraf, "Modifying "Pico" question into "Picos" model for more robust and reproducible presentation of the methodology employed in a scientific study," *World Journal of Plastic Surgery*, vol. 6, no. 3, p. 390, 2017.

[30] S. C. Nayak and C. Tripathy, "Deadline based task scheduling using multi-criteria decision-making in cloud environment," *Ain Shams Engineering Journal*, vol. 9, no. 4, pp. 3315–3324, 2018.

[31] A. Shahidinejad and M. Ghobaei-arani, "Joint computation offloading and resource provisioning for edge-cloud computing environment," *A machine learning-based approach*, vol. 50, pp. 1–19, 2020.

[32] D. Li, C. Chen, J. Guan, Y. Zhang, J. Zhu, and R. Yu, "DCloud: deadline-aware resource allocation for cloud computing jobs," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 8, pp. 2248–2260, 2016.

[33] S. Durga, S. Mohan, and J. D. Peter, "A two-stage queue model for context-aware task scheduling in mobile multimedia cloud environments," *Advances in Big Data and Cloud Computing*, vol. 645, 2018.

[34] M. T. Islam, S. Karunasekera, and R. Buyya, "dSpark: deadline-based resource allocation for big data applications in Apache spark," in *Proceedings of the 2017 IEEE 13th International Conference on e-Science (e-Science)*, pp. 89–98, Auckland, New Zealand, October 2017.

[35] Y.-S. Chang, C.-T. Fan, R.-K. Sheu, S.-R. Jhu, and S.-M. Yuan, "An agent-based workflow scheduling mechanism with deadline constraint on hybrid cloud environment," *International Journal of Communication Systems*, vol. 31, no. 1, Article ID e3401, 2018.

[36] S. P. Praveen, U. Tulasi, and K. A. K. Teja, *A Cost Efficient Resource Provisioning Approach Using Virtual Machine Placement*, vol. 5, no. 2, pp. 2365–2368, 2014.

[37] S. Tuli, R. Sandhu, and R. Buyya, "Shared data-aware dynamic resource provisioning and task scheduling for data intensive applications on hybrid clouds using Aneka," *Future Generation Computer Systems*, vol. 106, pp. 595–606, 2020.

[38] S. Durga, S. Mohan, J. D. Peter, and S. Surya, "Context-aware adaptive resource provisioning for mobile clients in intra-cloud environment," *Cluster Comput*, vol. 22, pp. 9915–9928, 2019.

[39] D. Alsadie, E. J. Alzahrani, N. Sohrabi, Z. Tari, and A. Y. Zomaya, "DTFA: a dynamic threshold-based fuzzy approach for power-efficient VM consolidation," in *Proceedings of the 2018 IEEE 17th International Symposium on Network Computing and Applications (NCA)*, pp. 1–9, Cambridge, MA, USA, November 2018.

[40] Y. Lu, L. Liu, J. Gu, J. Panneerselvam, and B. Yuan, "EA-DFPSO: an intelligent energy-efficient scheduling algorithm for mobile edge networks," *Digital Communications and Networks*, vol. 8, no. 3, pp. 237–246, 2022.

[41] A. Verma and S. Kaushal, "Deadline constraint heuristic-based genetic algorithm for workflow scheduling in cloud," *International Journal of Grid and Utility Computing*, vol. 5, no. 2, pp. 96–106, 2014.

[42] M. Malawski, G. Juve, E. Deelman, and J. Nabrzyski, "Algorithms for cost-and deadline-constrained provisioning for scientific workflow ensembles in IaaS clouds," *Future Generation Computer Systems*, vol. 48, pp. 1–18, 2015.

[43] A. Nadjaran Toosi, R. O. Sinnott, and R. Buyya, "Resource provisioning for data-intensive applications with deadline constraints on hybrid clouds using Aneka," *Future Generation Computer Systems*, vol. 79, pp. 765–775, 2018.

[44] P. Lai, Q. He, X. Xia et al., "Dynamic user allocation in stochastic mobile edge computing systems," *IEEE Trans. Serv. Comput.*vol. 15, no. 5, pp. 2699–2712, 2022.

[45] E. Daniel, S. Durga, and M. Vijila, "A continuous sampling method for batch data auditing in cloud storage," *International Journal of Information Systems in the Service Sector*, vol. 13, no. 2, pp. 1–12, 2021.

[46] E. Daniel and N. A. Vasanthi, "An efficient continuous auditing methodology for outsourced data storage in cloud computing," *Advances in Intelligent Systems and Computing*, vol. 412, pp. 461–468, 2016.

[47] J. Andrew and G. J. W. Kathrine, "An intrusion detection system using correlation," *Prioritization and clustering techniques to mitigate false alerts*, vol. 645, 2018.

[48] J. Andrew, J. Karthikeyan, and J. Jebastin, "Privacy preserving big data publication on cloud using mondrian anonymization techniques and deep neural networks," *ICACCS*, in *Proceedings of the 2019 5th International Conference on Advanced Computing and Communication Systems*, pp. 722–727, Coimbatore, India, March 2019.

[49] Y. w. Zhang, W. m. Zhang, K. Peng, D. c. Yan, and Q. l. Wu, "A novel edge server selection method based on combined genetic algorithm and simulated annealing algorithm," *Automatika*, vol. 62, no. 1, pp. 32–43, 2021.

[50] S. T. Manukumar, "An efficient agent based," pp. 1–15, 2021.

[51] G. Lee, H. Park, S. Heo, K. A. Chang, H. Lee, and H. Kim, "Architecture-aware automatic computation offload for native applications," in *Proceedings of the 48th international*

symposium on microarchitecture, pp. 521–532, Dece, New York NY, 2015.

[52] O. Ascigil, A. G. Tasiopoulos, T. K. Phan, V. Sourlas, I. Psaras, and G. Pavlou, "Resource provisioning and allocation in function-as-a-service edge-clouds," *IEEE Trans. Serv. Comput.*, vol. 15, no. 4, pp. 2410–2424, 2022.

[53] P. Nawrocki, B. Sniezynski, J. Kolodziej, and P. Szynkiewicz, "Adaptive context-aware service optimization in mobile cloud computing accounting for security aspects," *Concurrency and Computation: Practice and Experience*, vol. 33, no. 18, pp. 1–13, 2021.

[54] F. Farahbakhsh, A. Shahidinejad, and M. Ghobaei-Arani, "Context-aware computation offloading for mobile edge computing," *Journal of Ambient Intelligence and Humanized Computing*, vol. 1, no. 1, pp. 1–13, 2021.

[55] J. Chase and D. Niyato, "Joint optimization of resource provisioning in cloud computing," *IEEE Trans. Serv. Comput.*vol. 10, no. 3, pp. 396–409, 2017.

[56] S. Mireslami, L. Rakai, B. H. Far, and M. Wang, "Simultaneous cost and QoS optimization for cloud resource allocation," *IEEE Transactions on Network and Service Management*, vol. 14, no. 3, pp. 676–689, 2017.

[57] S. Midya, A. Roy, K. Majumder, and S. Phadikar, "Multi-objective optimization technique for resource allocation and task scheduling in vehicular cloud architecture: a hybrid adaptive nature inspired approach," *Journal of Network and Computer Applications*, vol. 103, pp. 58–84, 2017.

[58] L. Chunlin and L. LaYuan, "Cost and energy aware service provisioning for mobile client in cloud computing environment," *The Journal of Supercomputing*, vol. 71, no. 4, pp. 1196–1223, 2015.

[59] H. Qian and D. Andresen, "Extending mobile device's battery life by offloading computation to cloud," in *Proceedings of the 2015 2nd ACM International Conference on Mobile Software Engineering and Systems*, pp. 150-151, Florence, Italy, May 2015.

[60] J. Niu, W. Song, and M. Atiquzzaman, "Bandwidth-adaptive partitioning for distributed execution optimization of mobile applications," *Journal of Network and Computer Applications*, vol. 37, no. 1, pp. 334–347, 2014.

[61] B. Zhou, A. V. Dastjerdi, R. N. Calheiros, S. N. Srirama, and R. Buyya, "MCloud: a context-aware offloading framework for heterogeneous mobile cloud," *IEEE Trans. Serv. Comput.*vol. 10, no. 5, pp. 797–810, 2017.

[62] S. Durga, S. Mohan, J. Dinesh, and A. Aneena, "Cuckoo based resource allocation for mobile cloud environments," *Advances in Intelligent Systems and Computing*, vol. 412, pp. 543–550, 2016.

[63] N. Z. Naqvi, K. Moens, A. Ramakrishnan, D. Preuveneers, D. Hughes, and Y. Berbers, "To cloud or not to cloud: a context-aware deployment perspective of augmented reality mobile applications," in *Proceedings of the ACM Symp. Appl. Comput*, pp. 555–562, Salamanca, Spain, April 2015.

[64] R. K. Naha, S. Garg, A. Chan, and S. K. Battula, "Deadline-based dynamic resource allocation and provisioning algorithms in Fog-Cloud environment," *Future Generation Computer Systems*, vol. 104, pp. 131–141, 2020.

[65] S. Durga, S. Mohan, and J. D. Peter, "Multi-context based optimal resource provisioning in mobile cloud environments," *Journal of Computational and Theoretical Nanoscience*, vol. 15, no. 5, pp. 1762–1768, 2018.

[66] D. Spatharakis, I. Dimolitsas, D. Dechouniotis et al., "A scalable edge computing architecture enabling smart offloading for location based services," *Pervasive and Mobile Computing*, vol. 67, Article ID 101217, 2020.

[67] M. Jia, W. Liang, Z. Xu, M. Huang, and Y. Ma, "Qos-aware cloudlet load balancing in wireless metropolitan area networks," *IEEE Trans. Cloud Comput.*, vol. 8, no. 2, pp. 623–634, 2020.

[68] S. David, J. Andrew, B. Xavier, I. Joel Raj, and R. Jennifer Eunice, "Improving scalability and balancing the network load using adaptive multiparent crossover method in wireless sensor networks," *Journal of Computational and Theoretical Nanoscience*, vol. 17, no. 5, pp. 2415–2420, 2020.

[69] H. Elazhary, "Internet of Things (IoT), mobile cloud, cloudlet, mobile IoT, IoT cloud, fog, mobile edge, and edge emerging computing paradigms: disambiguation and research directions," *Journal of Network and Computer Applications*, vol. 128, pp. 105–140, November 2018.

[70] Z. Xu, W. Gong, Q. Xia, W. Liang, O. F. Rana, and G. Wu, "NFV-enabled IoT service provisioning in mobile edge clouds," *IEEE Transactions on Mobile Computing*, vol. 20, no. 5, pp. 1892–1906, 2021.

[71] M. G. R. Alam, M. M. Hassan, M. Zi. Uddin, A. Almogren, and G. Fortino, "Autonomic computation offloading in mobile edge for IoT applications," *Future Generation Computer Systems*, vol. 90, pp. 149–157, 2019.

[72] D. Sivan and M. Sellappa, "Proximity-based Cloud Resource Provisioning for Deep Learning Applications in Smart Healthcare," *Expert Systems*, vol. 39, pp. 1–15, 2019.

[73] S. Echeverría, J. Root, B. Bradshaw, and G. Lewis, "On-demand VM provisioning for cloudlet-based cyber-foraging in resource-constrained environments," in *Proceedings of the 2014 6th Int. Conf. Mob. Comput. Appl. Serv*, pp. 116–124, Austin, TX, USA, November 2014.

[74] M. M. Islam, M. A. Razzaque, M. M. Hassan, W. N. Ismail, and B. Song, "Mobile cloud-based big healthcare data processing in smart cities," *IEEE Access*, vol. 5, pp. 11887–11899, 2017.

[75] X. Sun and N. Ansari, "Green cloudlet network: a distributed green mobile cloud network," *IEEE Network*, vol. 31, no. 1, pp. 64–70, 2017.

[76] L. Tawalbeh, M. A. Tawalbeh, and M. Aldwairi, "Improving the impact of power efficiency in mobile cloud applications using cloudlet model," *Concurrency and Computation: Practice and Experience*, vol. 32, no. 21, pp. 1–11, 2020.

[77] G. A. Lewis, S. Echeverr\'\ia, S. Simanta, J. Root, and B. Bradshaw, "Cloudlet-based cyber-foraging in resource-limited environments," *Emerging Research in Cloud Distributed Computing Systems*, pp. 92–121, 2015.

[78] Y. Jararweh, A. Doulat, O. Alqudah, E. Ahmed, M. Al-Ayyoub, and E. Benkhelifa, "The future of mobile cloud computing: integrating cloudlets and Mobile Edge Computing," in *Proceedings of the 23rd International Conference on Telecommunications (ICT)*, p. 2016, Thessaloniki, Greece, May 2016.

[79] J. Park, H. Kim, Y.-S. Jeong, and E. Lee, "Two-phase grouping-based resource management for big data processing in mobile cloud computing," *International Journal of Communication Systems*, vol. 27, no. 6, pp. 839–851, 2014.

[80] P. Si, Q. Zhang, F. R. Yu, and Y. Zhang, "QoS-aware dynamic resource management in heterogeneous mobile cloud computing networks," *China Commun*, vol. 11, no. 5, pp. 144–159, 2014.

[81] R. Yu, X. Huang, J. Kang et al., "Cooperative resource management in cloud-enabled vehicular networks," *IEEE*

Transactions on Industrial Electronics, vol. 62, no. 12, pp. 7938–7951, 2015.

[82] K. Liu, X. Xu, M. Chen, B. Liu, L. Wu, and V. C. S. Lee, "A Hierarchical architecture for the future internet of vehicles," *IEEE Communications Magazine*, vol. 57, no. 7, pp. 41–47, 2019.

[83] U. Tadakamalla and D. A. Menasce, "Autonomic resource management using analytic models for fog/cloud computing," in *Proceedings of the 2019 IEEE International Conference on Fog Computing (ICFC)*, no. 2, pp. 69–79, Prague, Czech, June 2019.

[84] A. Karamoozian, A. Hafid, M. Boushaba, and M. Afzali, "QoS-aware resource allocation for mobile media services in cloud environment," in *Proceedings of the 2016 13th IEEE Annual Consumer Communications & Networking Conference (CCNC)*, pp. 732–737, Las Vegas, NV, USA, January 2016.

[85] S. Singh and I. Chana, "Q-Aware: quality of service based cloud resource provisioning," *Computers & Electrical Engineering*, vol. 47, pp. 138–160, 2015.

[86] M. M. Hassan, "Cost-effective resource provisioning for multimedia cloud-based e-health systems," *Multimedia Tools and Applications*, vol. 74, no. 14, pp. 5225–5241, 2015.

[87] M. Dabbagh, B. Hamdaoui, M. Guizani, and A. Rayes, "Energy-efficient resource allocation and provisioning framework for cloud data centers," *IEEE Transactions on Network and Service Management*, vol. 12, no. 3, pp. 377–391, 2015.

[88] K. Mitra, S. Saguna, C. Åhlund, and D. G. Lulea, "A mobility management system for mobile cloud computing," in *Proceedings of the 2015 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1608–1613, New Orleans, LA, USA, March 2015.

[89] Y. Zhang, D. Niyato, and P. Wang, "An auction mechanism for resource allocation in mobile cloud computing systems," in *Proceedings of the International Conference on Wireless Algorithms, Systems, and Applications*, pp. 76–87, Berlin, Heidelberg, 2013.

[90] S. K. Sood and R. Sandhu, "Matrix based proactive resource provisioning in mobile cloud environment," *Simulation Modelling Practice and Theory*, vol. 50, pp. 83–95, 2015.

[91] S. Din, A. Paul, A. Ahmad, and G. Jeon, "Energy efficient green solution for hierarchical resource management for mobile cloud computing," in *Proceedings of the 2019 IEEE International Conference on Communications Workshops (ICC Workshops)*, pp. 1–6, Shanghai, China, May 2019.

[92] J. Li and X. Xu, "EERA: an energy-efficient resource allocation strategy for mobile cloud workflows," *IEEE Access*, vol. 8, Article ID 217008, 2020.

[93] H. Zhang, Z. Chen, J. Wu et al., "Energy-efficient online resource management and allocation optimization in multi-user multi-task mobile-edge computing systems with hybrid energy harvesting," *Sensors*, vol. 18, no. 9, pp. 3140–3223, 2018.

[94] J. Guo, Z. Song, Y. Cui, Z. Liu, and Y. Ji, "Energy-efficient resource allocation for multi-user mobile edge computing," in *Proceedings of the GLOBECOM 2017-2017 IEEE Global Communications Conference*, pp. 1–7, Singapore, 2017.

[95] M. Avgeris, D. Spatharakis, D. Dechouniotis, A. Leivadeas, V. Karyotis, and S. Papavassiliou, "ENERDGE: distributed energy-aware resource allocation at the edge," *Sensors*, vol. 22, no. 2, p. 660, 2022.

[96] Y. Bai and Y. Zhang, "StoArranger: enabling efficient usage of cloud storage services on mobile devices," in *Proceedings of the 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, pp. 1476–1487, Atlanta, GA, USA, June 2017.

[97] D. Dechouniotis, N. Athanasopoulos, A. Leivadeas, N. Mitton, R. Jungers, and S. Papavassiliou, "Edge computing resource allocation for dynamic networks: the DRUID-NET vision and perspective," *Sensors*, vol. 20, no. 8, p. 2191, 2020.

[98] T. Abdelzaher, Y. Hao, K. Jayarajah et al., "Five challenges in cloud-enabled intelligence and control," *ACM Transactions on Internet Technology*, vol. 20, no. 1, pp. 1–19, 2020.

[99] S. Durga, E. Daniel, and P. G. J. Leelipushpam, "A novel request state aware resource provisioning and intelligent resource capacity prediction in hybrid mobile cloud," *Journal of Ambient Intelligence and Humanized Computing*, vol. 13, no. 5, pp. 2637–2650, 2021.

[100] H. T. T. Binh, N. P. Le, N. B. Minh, T. T. Hai, and N. Q. Minh, "A reinforcement learning algorithm for resource provisioning in mobile edge computing network," in *Proceedings of the 2020 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–7, Glasgow, UK, July 2020.

[101] Y. Jin, L. Jiao, Z. Qian, S. Zhang, and S. Lu, "Learning for learning: predictive online control of federated learning with edge provisioning," in *Proceedings of the IEEE INFOCOM 2021 - IEEE Conference on Computer Communications*, Vancouver, BC, Canada, 2021 May.