Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 5549470, 30 pages
https://doi.org/10.1155/2021/5549470

Research Article

Mobile Intercloud System for Edge Cloud Computing

Yi Dou, Yik Him Ho, Yuxuan Deng, and Henry C. B. Chan

Department of Computing, The Hong Kong Polytechnic University, Hong Kong

Correspondence should be addressed to Yi Dou; yi.dou@connect.polyu.hk

Received 10 February 2021; Revised 16 July 2021; Accepted 26 July 2021; Published 3 September 2021

Academic Editor: Huaqun Wang

Recent years have seen considerable interest in mobile cloud computing and edge cloud computing. This paper presents a mobile Intercloud system for supporting mobile cloud computing in general and edge cloud computing in particular. In essence, a mobile user with a mobile terminal can set up a virtual mobile terminal with applications and data in a central/home cloud. The virtual mobile terminal can facilitate task and computation offloading and other functions. Moreover, when a mobile terminal joins an edge cloud, the virtual mobile terminal (including required applications and data) can be migrated to enhance system efficiency and the user experience (e.g., shorter access delays). An experimental prototype has been developed for evaluating certain basic object transfer functions. To support the application transfer function, we formulate both finite- and infinite-horizon Markov decision models to determine decision policies (i.e., should an application be transferred to an edge cloud). The transfer decision depends on various factors, including transfer cost, duration associated with the edge cloud, usage probability, and usage cost in the central cloud and edge cloud. Based on the models, we obtain closed-form solutions for the decision policies, which can be expressed in meaningful formulas to provide useful insights for edge cloud computing in general. To evaluate the mobile Intercloud system for edge cloud computing, we conducted extensive evaluations, including experimental evaluation for testing the basic functions and protocols, analytical evaluation for studying the analytical models, and simulation evaluation for analyzing performance in a multiuser and multicloud environment in particular. The experimental, simulation, and analytical results provide useful insights into the design and development of the mobile Intercloud system for edge cloud computing as well as decision policies for application transfer.

1. Introduction

In general, cloud computing enables computing resources such as storage and services to be shared dynamically and flexibly over the Internet [1, 2]. In particular, it enables using services/resources dynamically (i.e., based on user demand) and automatically, accessing services through different means, and providing services with a utility-based approach. According to [1, 2], cloud computing services are typically offered through software-as-a-service, platform-as-a-service, and infrastructure-as-a-service and there are four deployment methods: private clouds, community clouds, public clouds, and hybrid clouds. Based on the forecast in [3], the market for cloud computing services will continue to grow dramatically in the next few years. With the rapid growth of smartphones and other mobile terminals, there has been a strong demand for mobile cloud computing services (e.g., see the statistics/forecast on mobile cloud traffic in [4]).

As shown in Figure 1, research on cloud computing can in general be divided into four categories. Most research focuses on basic cloud computing for general computers (i.e., stationary terminals) typically for intraoperations (e.g., over homogeneous clouds). In general, mobile cloud computing seeks to extend cloud computing to mobile terminals. As mobile terminals are limited in computation power, a typical application of mobile cloud computing is to support computation offloading so that intensive tasks can be offloaded to a cloud for processing. Mobile cloud computing typically supports homogeneous cloud computing for mobile terminals (e.g., with a focus on intraoperations) [5–8]. On the other hand, inspired by the Internet, Intercloud aims to support interactions and operations over heterogeneous clouds

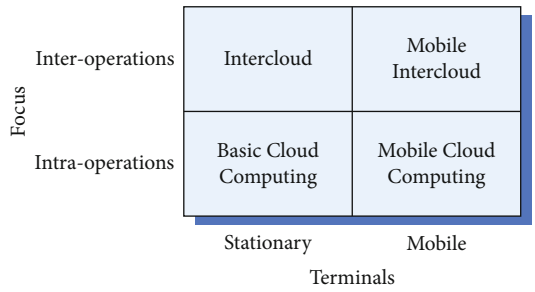

Figure 1: Different types of cloud computing research.

(i.e., running by different cloud providers or systems) [9, 10]. Note that an Intercloud system allows clouds to interact directly with each other without involving end users. As there are many new research challenges and problems, it is of great interest to extend Intercloud to a mobile environment. We refer to this as the "mobile Intercloud system." In other words, similar to the relationship between cloud computing and mobile cloud computing, mobile Intercloud involves the extension of the Intercloud environment to mobile terminals, allowing them to manage data and applications over heterogeneous clouds. In this paper, our focus is to investigate the use of the mobile Intercloud system to support edge cloud computing, allowing mobile users to access resources more effectively and efficiently. Edge cloud computing aims to bring computation resources closer to the network edge (i.e., closer to a mobile terminal). A typical application of edge cloud computing is to support the Internet of Things. As a summary, Table 1 gives an overview of Intercloud, mobile Intercloud, mobile cloud computing, and edge cloud computing.

Note that while there can also be multiple clouds involved in mobile cloud computing, a multicloud approach is often used (i.e., a mobile terminal uses multiple cloud services). To further explain the motivation of mobile Intercloud, Figure 2 provides an example. Suppose that there are three clouds A, B, and C, which are operated by three different cloud service providers. A mobile terminal can interact with the clouds using the multicloud approach. In this case, for example, if a mobile terminal wants to move an object (e.g., application or data) from cloud A and cloud B, it needs to download it from cloud A to the mobile terminal first and then upload it from the mobile terminal to cloud B. It may not be efficient because of the indirect transfer and the limited power of the mobile terminal, especially if the objects are large. If the clouds can interact with each other under an Intercloud framework and using an Intercloud protocol, the object can be moved directly from cloud A to cloud B as directed by the mobile terminal. In fact, a batch of objects can be transferred as well. This is the mobile Intercloud approach. Let us look at another scenario. Suppose that cloud A (e.g., edge cloud) and cloud B (e.g., central cloud) are closer and more remote to the mobile terminal. The mobile terminal can download an object from cloud B. Alternatively, the mobile terminal can direct the edge cloud (i.e., cloud A) to obtain the object from the central cloud first and then download it from the edge cloud (i.e., cloud A). Obliviously, this approach (i.e., transferring through Intercloud and then downloading from edge cloud) can be more efficient and effective. In summary, mobile Intercloud can be used to facilitate edge cloud computing. As shown by later experiments, the second approach (i.e., transferring through Intercloud and then downloading from edge cloud) can be more efficient and effective. The above example can be extended to supporting applications through computation offloading as well (see the later part of the paper).

In summary, with the mobile Intercloud system, the concerned clouds (e.g., central cloud and edge cloud) can interact directly with one another as controlled or managed by mobile terminals (e.g., service policies can be defined). In some cases, the Intercloud operation can even be transparent to the mobile terminals. The aim of this paper is to study mobile Intercloud for supporting edge cloud computing. In general, there are three research questions. First, what are the system and protocols for enabling the transfer of data/applications across different clouds (e.g., running by different cloud providers) in a mobile Intercloud system for edge cloud computing and the cooperation of heterogeneous clouds under a mobile environment? In this paper, we present a mobile Intercloud system, which is inspired by mobile cellular systems and mobile IP. Imagine that a physical mobile terminal can be associated with a virtual mobile terminal maintaining all data and applications in the cloud. When a mobile user (i.e., with his/her mobile terminal) enters a new region/area associated with an edge cloud (i.e., similar to roaming to another mobile network or joining an edge network), the virtual mobile terminal and certain data and applications can also be moved (i.e., transferred to an edge cloud and staying close to the physical mobile terminal). Second, what is the decision policy for transferring an application to an edge cloud? In this paper, we study this important issue using a Markov decision model, taking into consideration various factors. Third, what is the performance of the proposed mobile Intercloud system for edge cloud computing? As discussed later, extensive evaluations will be conducted, including simulations for multiple clouds and multiple users to illustrate the benefit of transferring applications in some cases.

In summary, the contributions of this paper are summarized as follows.

(i) We present a mobile Intercloud system and the associated protocols for supporting edge cloud computing. It facilitates the transfer of data and applications between heterogeneous clouds so that mobile users can use the resources more effectively and efficiently

(ii) We formulate a Markov decision-based model/problem to analyze an application migration problem for edge cloud computing (i.e., should an application be migrated to an edge cloud taking into consideration various factors). Both finite-horizon and infinite-horizon models are investigated

Table 1: Definitions and examples/scenarios.

Term	Definition	Example/scenario
Intercloud	Inspired by the Internet, Intercloud is aimed at facilitating the interactions and operations among heterogeneous clouds	Cloud service providers can interact with each other through the Intercloud protocols
Mobile Intercloud	As a subset of Intercloud, mobile Intercloud is the extension of Intercloud to a mobile environment, allowing mobile terminals to manage data/applications over heterogeneous clouds	Roaming cloud service can be provided, and edge cloud computing service can be supported
Mobile cloud computing	As a subset of cloud computing, mobile cloud computing extends cloud computing to a mobile environment	Computation offloading service can be provided
Edge Cloud Computing	Edge cloud computing can be viewed as a special case of mobile cloud computing. It aims to bring computation resources closer to the network edge	Internet of Things applications can be supported. Note that mobile Intercloud can be used to facilitate the implementation of edge cloud computing

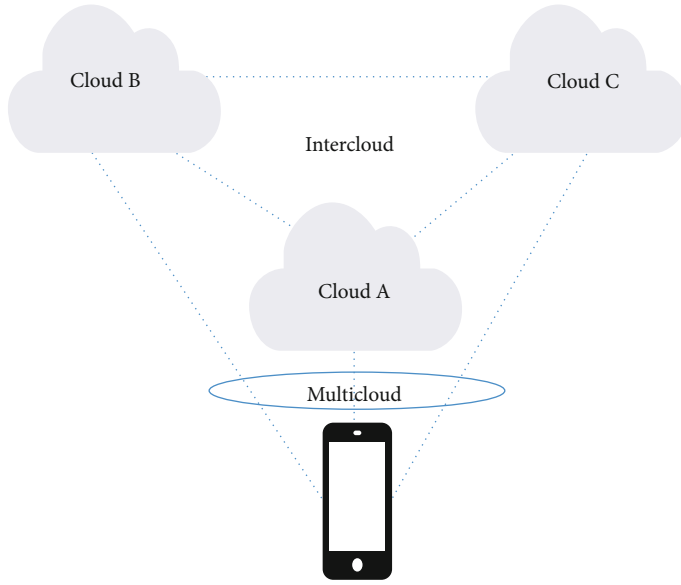

Figure 2: Intercloud and multicloud system.

(iii) By solving the aforementioned Markov decision-based problem and making certain assumptions, closed-form solutions, which provide valuable insights into the design of the decision policy and system, are obtained. The closed-form solutions should be useful (e.g., as a reference) for edge cloud computing in general

(iv) Extensive evaluations have been conducted, including experimental evaluation for proof-of-concept purposes, analytical evaluation for investigating the effect of the key system parameters, and simulation evaluation for studying the performance of multiple clouds and multiple users

The rest of the paper is organized as follows. Section 2 presents the related work. Section 3 presents the overall architecture of our system. Section 4 introduces the use of the Markov decision process to find optimal policies when deciding the migration of applications for edge cloud computing. Section 5 provides the experimental, analytical, and simulation evaluations. Section 6 concludes our work.

2. Related Work

2.1. Mobile Cloud Computing. With the growing demand for cloud computing and mobile terminals/devices (e.g., notebooks and smartphones), there has also been considerable interest in mobile cloud computing, which seeks to extend cloud computing to mobile terminals (e.g., see the survey at [5]). Based on the role/function of mobile terminals, [6] presents three service models for mobile cloud computing: obtaining a service from a cloud, offering a service (i.e., as a service provider), and functioning as a service broker. As discussed in [7], mobile cloud computing can be categorized according to four application models: performance-based (i.e., enhancing the performance by offloading tasks and applications, e.g., CloneCloud), energy-based (i.e., saving energy or power through the use of cloud resources, e.g., μ Cloud), constraint-based (i.e., carrying out different tasks on a mobile terminal and a cloud, e.g., eXCloud), and multi-objective-based (i.e., minimizing energy while fulfilling certain performance requirements, e.g., Cuckoo). [8] studies mobile cloud computing with a focus on augmentation

techniques (i.e., outsourcing computation and storage to enhance mobile terminal performance). Computation augmentation techniques can be categorized into four main types, namely, code offloading, service-oriented task delegation, parallel execution, and opportunistic mobile collaboration according to [8]. [11] gives an overview of decentralized cloud in general, including mobile cloud computing, mobile ad hoc computing, and edge computing.

2.2. General Computation Offloading for Mobile Cloud Computing. As illustrated by the above and other studies, computation offloading (or outsourcing computation to clouds in general) is a major area of mobile cloud computing. Here are examples of recent work related to this area. With the aim of minimizing the global task completion time, [12] studies an integer programming model for task offloading and scheduling for mobile clouds. To facilitate real-time implementation, the paper proposes an online code offloading and scheduling (OCOS) algorithm based on a rent/buy model. [13] presents an energy-efficient multisite offloading policy (EMOP) algorithm based on a Markov decision model. The objective is to minimize energy consumption through multisite offloading (i.e., among multiple clouds). Based on the Lyapunov optimization method, [14] investigates an offloading algorithm to optimize energy efficiency and throughput fairness. In addition to offloading, the paper also considers task arrival rates and local scheduling/processing in the optimization model. [15] investigates computation offloading for multiple mobile service workflows. Using a genetic algorithm, the offloading method seeks to optimize execution time and energy efficiency. Unlike other model-based approaches, [16] proposes an evidence-aware approach for computation offloading. Basically, the offloading policy is based on crowdsensing data, and an evidence-aware mobile computational offloading platform has been developed for implementation. By means of a mathematical graph model, [17] studies computation offloading for 3G and Wi-Fi networks. Based on the model, three heuristic algorithms are employed to minimize the financial cost while fulfilling the energy and delay requirements. With the goal of optimizing execution time as well as power consumption, [18] proposes a context-aware offloading scheme. It takes different contexts, such as network conditions and available cloud resources, into consideration, and a prototype system has been developed. [19] presents a cloud-assisted dynamic offloading scheme. With the assistance of the cloud, offloading decisions can be made based on resource information and network conditions, subject to energy constraints.

2.3. Markov Model-Based Computation Offloading for Mobile Cloud Computing. With the aim of enhancing experience quality and minimizing service migration cost, [20] employs a Markov decision model to determine a service migration policy based on geographical proximity and the workload of federated data centers (i.e., under a federated cloud environment). Based on a Markov decision process, [21] investigates service migration for mobile microcloud at the network edge. It seeks to minimize data transmission cost and migration cost. [22] proposes a Markov decision-based service placement algorithm for cloud-centric games under an edge cloud computing environment. The aim is to determine the optimal service placement taking into consideration both migration overhead and bandwidth. To minimize power consumption and request/response time, [23] presents a semi-Markovian process for application management for mobile terminals (i.e., computational offloading). [24] formulates a Markov decision model to determine an optimal offloading scheme for a cloudlet-based system with intermittent connectivity. It takes into consideration various factors, including workload of mobile terminals, availability of cloudlets, and user mobility pattern. [25] investigates a continuous time Markov decision process for mobile video offloading under a vehicular cloud environment. The aim is to maximize utilization of resources while fulfilling certain social restrictions. [26] studies a hidden Markov model autoscaling offloading (HMAO) mechanism to support the offloading decision (i.e., where to run a task: mobile, fog, or cloud) by considering energy consumption as well as execution time.

2.4. Computation Offloading for Edge Cloud Computing. With the advent of cloud computing, there are new and emerging cloud-based applications with low latency requirements (e.g., Internet of Things-based applications) and real-time performance (e.g., real-time games). By employing a relatively centralized system architecture, conventional cloud computing services cannot fulfill these requirements effectively and efficiently (e.g., response time cannot be guaranteed). As a complementary service, edge cloud computing has been proposed to extend computational resources as well as processing capability to the network edge (i.e., close to the users). In other words, users can access resources and applications through edge clouds so that better performance can be achieved (e.g., in terms of response time and latency). As edge clouds are close to users, they also facilitate computation offloading. However, note that edge clouds are usually less powerful than central clouds so there are also certain limitations. In [37, 38], an overview of computation offloading for mobile edge computing is given, such as types of offloading, architectures, offloading algorithms, resource management, and scheduling methods. With the aim of optimizing the computational overhead, [27] proposes a game-based approach using a distributed computational offloading algorithm with the Nash equilibrium for a wireless environment with multiple channels and interference. [28] studies the computation offloading problem from the energy consumption aspect. Based on the Lyapunov optimization, it proposes a greedy maximal task offloading scheduling algorithm for a multiuser and multitask scenario. To support vehicular edge cloud computing, [29] proposes an advanced deep learning-based computational offloading algorithm to enhance energy consumption and resource utilization. [30] focuses on reducing the total task delay by means of a computation offloading algorithm with artificial intelligence technology. To support Internet of Things, [31] studies a game theoretic approach using a multihop cooperative messaging mechanism to enhance computation time as well as energy consumption for an industrial Internet of Things edge cloud computing scenario. [32] considers a probabilistic offloading strategy

for edge cloud computing. By means of queuing theory and taking into consideration channel interference, the offloading probability of each terminal is computed based on a combinatorial optimization model. [33] analyzes the performance of a cloud-fog-edge computing system (i.e., a three-layer system) based on a Markov queuing system model as well as a computational resource allocation scheme. The aim is to optimize the social welfare metric subject to quality-of-service constraints. [34] investigates an image fusion method in conjunction with a data offloading scheme for 5G-based mobile terminals. A continuous time Markov model is employed to support the offloading over a cloud-based system. [35] studies an offloading method for a fog computing system (i.e., deciding if a task should be processed by a fog node through offloading). Two queuing models are considered for the model formulation, and an optimization problem is investigated with the aim of optimizing energy efficiency subject to some delay constraints. [36] investigates an offloading problem for vehicular edge computing. Queuing theory is used to derive the traffic/delay models, and an optimization problem is formulated by considering both energy consumption and delay. An energy-efficient resource allocation algorithm is also developed.

2.5. Summary Table for Computation Offloading. Tables 2–4 summarize the aforementioned related work. While various models have been studied for computation offloading, it is also of interest to consider a basic conceptual model, possibly with closed-form solutions. In this paper, we investigate a related problem but in a different context (i.e., from a new perspective). We consider that applications are offloaded or outsourced to clouds (i.e., to virtual terminals) under a mobile Intercloud system. Our aim is to determine when a virtual terminal should be transferred to an edge cloud to minimize costs (e.g., to facilitate computation offloading and edge cloud computing functions). Based on both finite-horizon and infinite-horizon Markov decision models, closed-form solutions are derived.

2.6. Intercloud. Another related and important development for cloud computing is Intercloud, which is aimed at facilitating the interconnection of clouds similar to the interconnection of networks (i.e., the Internet) [9, 10]. In general, an Intercloud system can be developed by using an Internet-like model, creating a new architecture or using an overlay approach. IEEE P2302 [39] employs the Internet-like approach. In other words, its aim is to enable the interconnection of clouds as inspired by the Internet model. In [40], the Intercloud Architecture Framework (i.e., a new framework or protocol model) is proposed for Intercloud systems. In [41], NGSON (i.e., a service overlay network) is presented to facilitate Intercloud operations. With the advent of Intercloud systems, mobile cloud computing can be extended to a new dimension, allowing data and applications to transfer across heterogeneous clouds under the management of mobile terminals. In this paper, we adopt the IEEE P2302 approach (i.e., the Internet-like approach). As shown in Figure 3, the basic IEEE P2302 Intercloud system comprises three main elements: Intercloud root, Intercloud exchanges, and Intercloud gateways. Resembling the Internet root for the domain name system, the Intercloud root (e.g., a cluster of root servers) provides directory-related services such as searching for clouds and supporting Intercloud authentication (e.g., serving as a certificate authority). The Intercloud root is linked with Intercloud exchanges, which function like Internet exchanges. In particular, Intercloud exchanges facilitate cloud-to-cloud as well as cloud-to-root interactions. Each cloud is linked to an Intercloud gateway. Intercloud gateways facilitate Intercloud communications through XML-based messages [42]. This paper can be viewed as an extension and enhancement of our previous/preliminary work in [43, 44]. The focus is on using the mobile Intercloud system to support edge cloud computing. Compared to the previous preliminary work, we not only include more comprehensive related work and provide a detailed design but also investigate both finite and infinite-horizon Markov decision models. In particular, as mentioned above, closed-form solutions for the decision policies have been developed, which provide a better understanding of the policies and valuable insights for supporting edge cloud computing in particular. Furthermore, extensive evaluations have been conducted, including experimental evaluation (i.e., using a prototype to evaluate the basic functions for proof-of-concept purposes), analytical evaluation (i.e., to evaluate the effect of the system parameters on the performance), and simulation evaluation, including multiple edge cloud and multiple user cases.

3. System and Protocol

3.1. Overview of Mobile Intercloud. As mentioned above, the aim of this paper is to present a mobile Intercloud system for supporting edge cloud computing in particular. Basically, in addition to a physical mobile terminal, a mobile user can also have a virtual mobile terminal. While there are handover and roaming processes for physical mobile terminals in the physical domain, similar processes can occur in the cyber domain as well (e.g., moving a virtual mobile terminal including data and applications from a central cloud to an edge cloud). A physical mobile terminal can offload data and applications to the corresponding virtual mobile terminal (i.e., to store data and run applications in the cloud). That means physical and virtual mobile terminals can work together to provide more effective and efficient service to the mobile user.

The mobile Intercloud system is closely related to mobile cloud computing, but it has differences as well. Obviously, virtual mobile terminals can provide an effective platform for supporting computation offloading, a key service of mobile cloud computing in general. Indeed, as each virtual mobile terminal is closely associated with a physical mobile terminal, personal data and applications can be stored or offloaded more effectively for the mobile user. In the mobile Intercloud system, due to the mobility of both the physical and virtual mobile terminals, they can work more closely with one another. This can greatly facilitate the offloading function and other edge computing services. Note that data and applications can be transferred as well. We shall study a related application transfer problem in the next section.

Table 2: General computation offloading for mobile cloud computing.

Work	Model/technique	Algorithm/scheme	Optimization goal	Scenario
[12]	Mixed integer nonlinear programming	Online code offloading and scheduling algorithm	Global task completion time	Wireless network connection and device mobility
[13]	Discrete time Markov chain and shortest path	Energy-efficient multisite offloading policy algorithm	Energy consumption	Multisite offloading
[14]	Lyapunov optimization	Energy-efficient computation offloading algorithm	Energy efficiency and throughput fairness	Mobile device resource management
[15]	Genetic algorithm	GA-based computation offloading algorithm	Executive time and energy efficiency	Mobile service work flow
[16]	Crowdsensing	Evidence-aware mobile computational offloading	Response time and energy footprint	Offloading based on crowdsensing datasets
[17]	Mathematical graph	Heuristic algorithm based on K-LARAC and M-LARAC	Financial cost subject to energy and delay	Wi-Fi and 3G networks
[18]	Optimization problem	Energy-efficient dynamic offloading and resource scheduling	Energy consumption with time constraint	Offloading and cooperative task scheduling
[19]	Basic mathematical model	Cloud-assisted computation offloading	Energy and latency	Selection of resource provider and execution plan

Table 3: Markov model-based computation offloading for mobile cloud computing.

Work	Model/technique	Algorithm/scheme	Optimization goal	Scenario
[20]	Markov decision process	Service migration policy model based on geographical proximity	Quality of experience and service migration cost	Federated cloud and distributed mobile network
[21]	Markov decision process	Mobility-based service migration policy model	Data transmission and migration cost	Mobile microcloud and wireless networks
[22]	Markov decision process	Efficient service placement algorithm	Migration overhead, latency, and bandwidth	Cloud-centric gaming and edge cloud computing
[23]	Semi-Markovian decision process	Application management for mobile devices	Request-response time and energy loss of battery	Computation offloading in mobile cloud computing
[24]	Markov decision process	Offloading scheme under a mobile environment	Computation and offloading costs	Intermittently connected cloudlet system
[25]	Continuous time Markov decision process	Social graph-based mobile video offloading	Resource utilization subject to social constraints	Mobile video offloading under a vehicular cloud system
[26]	Hidden Markov model	Autoscaling offloading	Energy consumption and execution time	Mobile fog computing

3.2. Key System Components. Figure 3 shows the key system components, extending/enhancing the previous work. Basically, each user has a physical mobile terminal in the physical domain and a virtual mobile terminal in the cyber domain. Essentially, the virtual mobile terminal functions like a software agent in the cloud domain. Equipped with data and applications, the physical and virtual mobile terminals can work together to provide better services (i.e., more effectively and efficiently). For example, some intensive tasks can be processed by the virtual mobile terminal, as the physical mobile terminal is relatively less powerful in terms of processing capability. In fact, the processing power of the virtual mobile terminal can also be flexibly increased whenever necessary. While a physical mobile terminal can move across areas, the corresponding virtual mobile terminal can also move across edge clouds (e.g., to remain close to the physical mobile terminal so that access delay can be minimized). Inspired by the mobile IP protocol, a mobile user is managed by a central or home cloud. The mobile user's virtual mobile terminal, including the user's data and applications, is hosted in the central cloud. That means the central cloud plays a primary role in supporting mobile Intercloud operations. As a mobile user enters a new area, region, or location, he/she can join or register with an edge cloud. The edge cloud can serve the user in collaboration with the central cloud. The interaction can be facilitated by the mobile Intercloud protocol. There can be different types or levels of edge clouds. For a city area, a more powerful edge cloud can be set up. For a small shop, a small edge cloud with basic functions should be more suitable. Within a certain local area or region (e.g., a shopping mall), users can be served by a medium edge cloud. There can also be other clouds in the system to provide various cloud computing services. That means the central cloud and edge cloud can also collaborate with other clouds to provide various services for a mobile user. Again, the mobile Intercloud protocol can facilitate the operations. To

Table 4: Computation offloading for edge cloud computing.

Work	Model/technique	Algorithm/scheme	Optimization goal	Scenario
[27]	Game theoretic approach	Distributed computation offloading algorithm with Nash equilibrium	Computational overhead	Multichannel wireless interference environment
[28]	Lyapunov optimization	Greedy maximal task offloading scheduling algorithms	Energy harvesting and overall system utility	Multiuser multitask computation offloading
[29]	Binary optimization problem	Advanced deep learning-based computational offloading algorithm	Energy consumption and resource utilization	Multilevel vehicular edge cloud computing
[30]	Artificial intelligence technology	Computation offloading algorithm based on task prediction	Total task delay	Mobile edge cloud computing
[31]	Game theoretic approach	Multihop cooperative messaging mechanism	Computation time and energy consumption	Industrial Internet of Things edge cloud computing
[32]	Queuing theory	Probabilistic offloading strategy	Offloading probability of each terminal	6G-enabled massive Industrial Internet of Things
[33]	Markov queueing system	Delay constrained data offloading	Social welfare subject to QoS constraints	Cloud-fog-edge computing system
[34]	Continuous time Markov model	Cloud-based data offloading	Multifocus and multiviews	Image fusion in mobile applications
[35]	Queuing models	Offloading method for a fog computing	Energy efficiency delay constraints	Fog computing system
[36]	Queuing theory	Workload offloading and power control	Energy consumption and delay	Vehicular edge computing

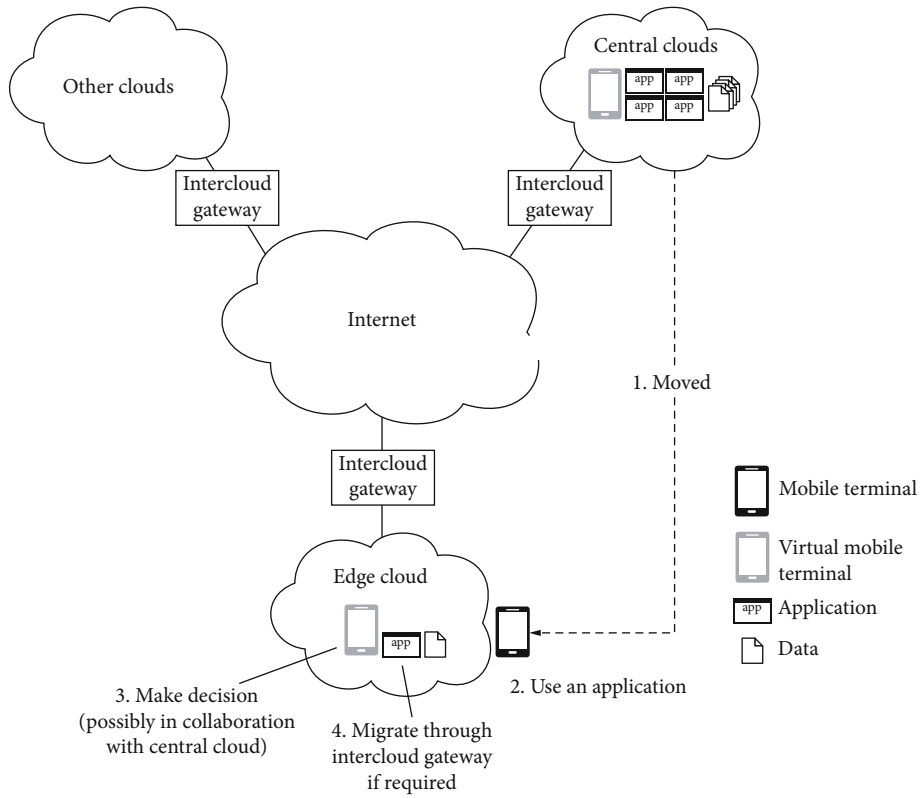

Figure 3: Basic mobile Intercloud system.

support Intercloud communications (e.g., between a central cloud and an edge cloud), Intercloud gateways are used, which communicate using the Intercloud Communications Protocol (ICCP) (i.e., by means of XML-based messages) (see [43]). That means each cloud is linked with an Intercloud gateway for Intercloud communications or

interactions. For example, a central cloud can send data or transfer a virtual mobile terminal (i.e., a virtual machine) to an edge cloud through ICCP.

3.3. System Protocols. Inspired by mobile IP [45] and extending the previous work, this section provides an overview of the system protocols to support the mobile Intercloud system for edge cloud computing. There are three major protocols, namely, registration protocol for registering with an edge cloud, service protocol for providing services, and deregistration protocol for disconnecting from an edge cloud.

3.3.1. Registration. As a mobile user enters a new location or area, he/she can first identify the available edge clouds (e.g., similar to the discovery of Wi-Fi networks by mobile terminals). Like mobile IP, a cloud discovery protocol can also be developed for this purpose. While the mobile user can choose to be served by the central cloud, in most cases, he/she should prefer registration with a nearby edge cloud. One advantage is to minimize access delay so as to provide better service. From another perspective, the registration process is similar to the roaming service provided by mobile network operators. In fact, cloud providers can form alliances with mobile network operators to provide an automatic roaming service. After joining an edge cloud, the edge cloud then registers the user (i.e., with the central cloud) based on a predefined authentication scheme. This procedure functions like a mobile IP registration protocol. After authentication, the corresponding virtual mobile terminal can be transferred to the edge cloud. Note that a virtual mobile terminal is essentially a virtual machine. Depending on the implementation requirement, the transfer can be implemented through direct transfer or generation of a virtual machine with the required configuration (e.g., same or simpler configuration depending on the edge cloud). The original virtual mobile terminal should still be maintained in the central cloud (i.e., serving as a master). The virtual mobile terminals in the edge cloud and central cloud can in fact be viewed as a single entity.

3.3.2. Services. After registering with the edge cloud, the mobile user can use various services through the physical mobile terminal in collaboration with the virtual mobile terminal (e.g., by running various applications). While the applications can be run in the central cloud, it is sometimes better to transfer an application to the edge cloud (i.e., running it by the virtual mobile terminal in the edge cloud). We shall study some decision algorithms for this application transfer problem in the next section. Basically, the decision depends on various factors, including the transfer cost, running cost in the central/edge cloud, expected duration for remaining in the edge cloud, and the usage probability. For instance, if an application is used frequently and the transfer cost is low, it is more justified to transfer the application to the edge cloud. However, if the application is not used frequently or the expected duration for staying in the edge cloud is short and the transfer cost is high, it is better not to transfer the application. The transfer decision can depend on the capabilities of the edge cloud (i.e., different for small, medium, and large edge clouds). There can be various implementation options. For example, the decision algorithms can be implemented in the virtual mobile terminals (e.g., see Figure 3). Whenever required, the virtual mobile terminals can also work with the central cloud to determine the decisions. Intercloud communications can be facilitated by the Intercloud Communications Protocol through the gateways. While we focus on transferring applications in this paper, data can be transferred from the central cloud to the edge cloud as well.

Note also that while a physical mobile terminal can offload tasks to the virtual mobile terminal, the virtual mobile terminal can also transfer data and applications to the physical mobile terminal as well (e.g., frequently used data). This allows mobile users to access them directly through the physical mobile terminal. That means, through the mobile Intercloud system, more comprehensive mobile cloud computing services can be provided through the interaction of the central cloud, edge cloud, physical mobile terminal, and virtual mobile terminal.

In summary, Figures 4–6 show three service scenarios for edge cloud computing through the mobile Intercloud system: full, minimal, and partial migrations. For full migration, basically all data and applications are transferred to the edge cloud from the central cloud. That means the edge cloud plays a primary role in providing services. In contrast, for minimal migration, only essential data are transferred to the edge cloud from the central cloud. In this case, the central cloud and edge cloud play a primary role and supporting role in providing the services, respectively. For partial migration, certain data and applications (e.g., frequently used data and applications) are transferred to the edge cloud from the central cloud. In the next section, a related application transfer problem will be studied. In all scenarios, the central cloud and/or edge cloud may also collaborate with other clouds to provide necessary services for the mobile user (i.e., through the Intercloud protocol).

3.3.3. Deregistration. Before leaving the area/region/location, the mobile user should perform deregistration with the edge cloud. The central cloud should be informed as well, and in most cases, data should be updated or synchronized so as to ensure data consistency.

3.4. A Prototype for Testing Certain Functions. For proof-of-concept purposes, we have developed a prototype to test some basic functions (i.e., object transfer functions). The first part of the prototype is a basic mobile app for managing the object transfer between two clouds through the Intercloud gateways. The objects can be files or applications. The mobile app seeks to test and demonstrate the functionality of object transfer based on ICCP. Figure 7 shows that there are two objects in cloud 1. After selecting an object and pressing the transfer button, the mobile app can then initiate the object transfer through the Intercloud gateway of cloud 1. Basically, an ICCP PUT message will be sent to the Intercloud gateway of cloud 2 to transfer the required object to cloud 2. Note that the transfer is conducted using an Intercloud approach rather than a multicloud approach. In other words, an object is transferred directly through two clouds instead of indirectly

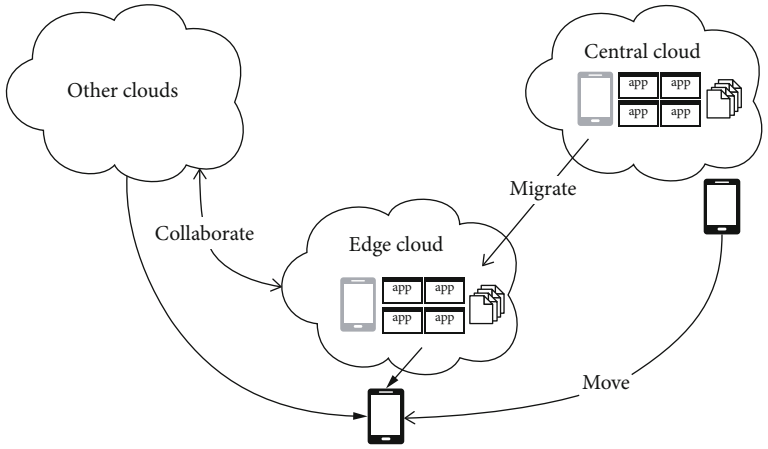

Figure 4: Full migration.

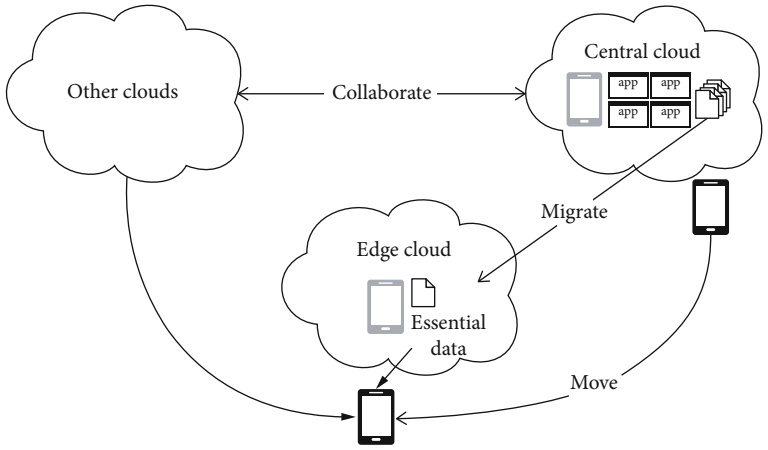

Figure 5: Minimal migration.

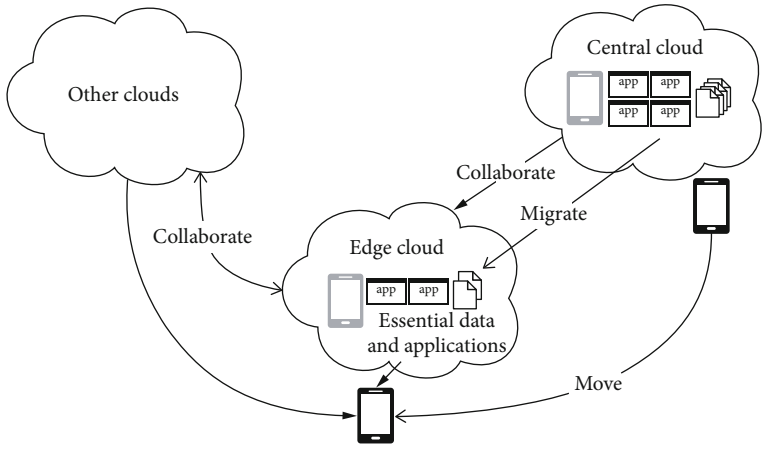

Figure 6: Partial migration.

through a mobile terminal (i.e., download the object from the originating cloud to the mobile terminal first and then transfer it to the destination cloud from the mobile terminal). The mobile app was developed based on MIT App Inventor blocks, including extension blocks for ICCP for object transfer, as shown in Figure 8. Note that this is an illustrative example. The extension blocks of ICCP can also be used for developing other mobile apps for the mobile Intercloud system. The second part of the prototype seeks to test and demonstrate how an Intercloud gateway can get or put

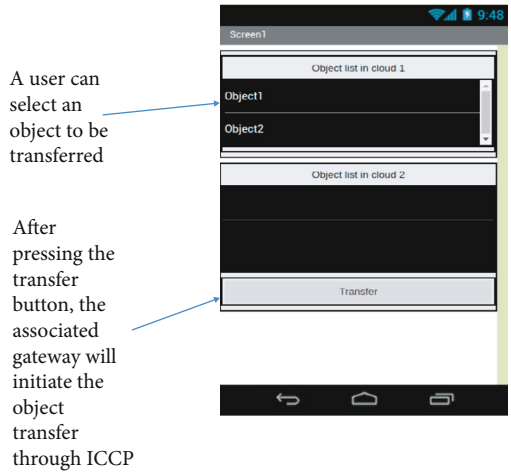

Figure 7: Object transfer through the Intercloud gateway.

applications from or to another cloud based on certain predefined requirements or algorithms. Figure 9 shows how a gateway can get an application from another cloud by issuing an ICCP GET command (e.g., as directed by an algorithm) through an ICCP request, which is defined based on XML. Similar to HTTP messages, ICCP messages allow two cloud gateways to communicate and interact with each other by means of predefined request/response messages. After receiving the command, the cloud then gets and transfers the application accordingly using ICCP (i.e., through an ICCP response). Similarly, Figure 10 shows how a gateway can put an application to another cloud by issuing an ICCP PUT command through an ICCP request (e.g., for certain predefined applications). Again, after receiving the applications, they can be uploaded to the cloud accordingly. In summary, cloud gateways can interact with each other through ICCP request/response XML-based messages similar to how clients and servers interact through HTTP request/response messages. Some experimental results will be presented in Section 5.1.

While the detailed implementation issues are outside the scope of this paper, it is expected that virtual mobile terminals for the mobile Intercloud system can be implemented based on similar computational offloading architectures for mobile cloud computing such as CloneCloud or Jade (see [46] for details). For example, CloneCloud-like virtual machine or Jade-like server with a runtime engine can be adapted or extended for the implementation.

4. Decision Models for Application Transfer

One of the fundamental research problems for this paper is to decide whether to move an application when a physical mobile terminal joins an edge cloud. The objective is to minimize the overall cost (see later definition). Note that while the subsequent usage cost is lower when an application is transferred from a central cloud to an edge cloud, there is a transfer cost. That means if a user only stays with an edge cloud for a short period of time or if the application usage is low, it may not be cost-effective to move the application. In this section, we investigate this application transfer problem using a Markov decision model. To facilitate the analysis, we consider a discrete time system. Like other discrete time systems, the time slot size can be set on a case-by-case basis (e.g., in the order of minutes or seconds depending on the cases). In each time slot, a user may or may not use the relevant application. If the user uses an application and the application is in the central cloud, a decision (i.e., to move or not to move) should be made at the start of each time slot. We assume that processing and transmission delays are negligible. Our focus is on the decision policy.

4.1. Finite-Horizon Markov Decision Process. First, we consider a finite-horizon Markov decision model, assuming that we know how long a user will stay with an edge cloud (e.g., leave the system at a certain time slot). We define the system state as a tuple of current application location and usage indication (e.g., state $s_{h,u}$ indicates that the application is at central/home and the user wants to use the application). We assume that the application can be either located in the central/home cloud or migrated to an edge cloud. Essentially, we have the following four states: $S = \{s_{h,u}, s_{h,i}, s_{f,u}, s_{f,i}\}$, where the subscripts h and f indicate that the application is located in the central cloud and edge cloud, respectively. Note that we use the subscript "f" (i.e., "foreign") to indicate a foreign area for an edge cloud in order not to use the confusing symbol e (2.728). Furthermore, the subscripts u and i indicate that the application is used and not used (i.e., idle) in the current time slot, respectively. In each time slot, the user may want to use the application with a certain probability. Let $P\{U\} = p$ and $P\{I\} = 1 - p$ be the probability that the user will use and not use the application, respectively, as shown in Figure 11. At the state $s_{h,u}$, the user can choose to either use the application directly from the central cloud at a cost of h or migrate the application to the edge cloud at a transfer cost of C so that it can be used through the edge cloud at a lower cost of f (i.e., for the current and subsequent time slots).

At state $s_{h,u}$, the possible actions are a_{move} (i.e., move the application from the central cloud to the edge cloud) and a_{stay} (i.e., retain the application in the central cloud). For other states, the actions can only be "stay." Note that if the application is already in the edge cloud or it is not required/used, no action is required. The cost of using the application depends on various factors such as bandwidth and data size.

The cost at time t is defined as follows: $\text{Cost}^t(s, a)$. That means the cost depends on the state and action. As shown in Figure 12, the cost comprises the following cost elements:

(i) h: cost of running an application in the central cloud, i.e., $s = s_{h,u}$

(ii) f: cost of running an application in the edge cloud, i.e., $s = s_{f,u}$

(iii) C: cost of transmitting an application from the central cloud to the edge cloud, i.e., $a = a_{\text{move}}$

Note that in this paper, the costs are defined in a general sense. That means they can be linked to response time, monetary cost, or any other costs to be optimized. This is similar

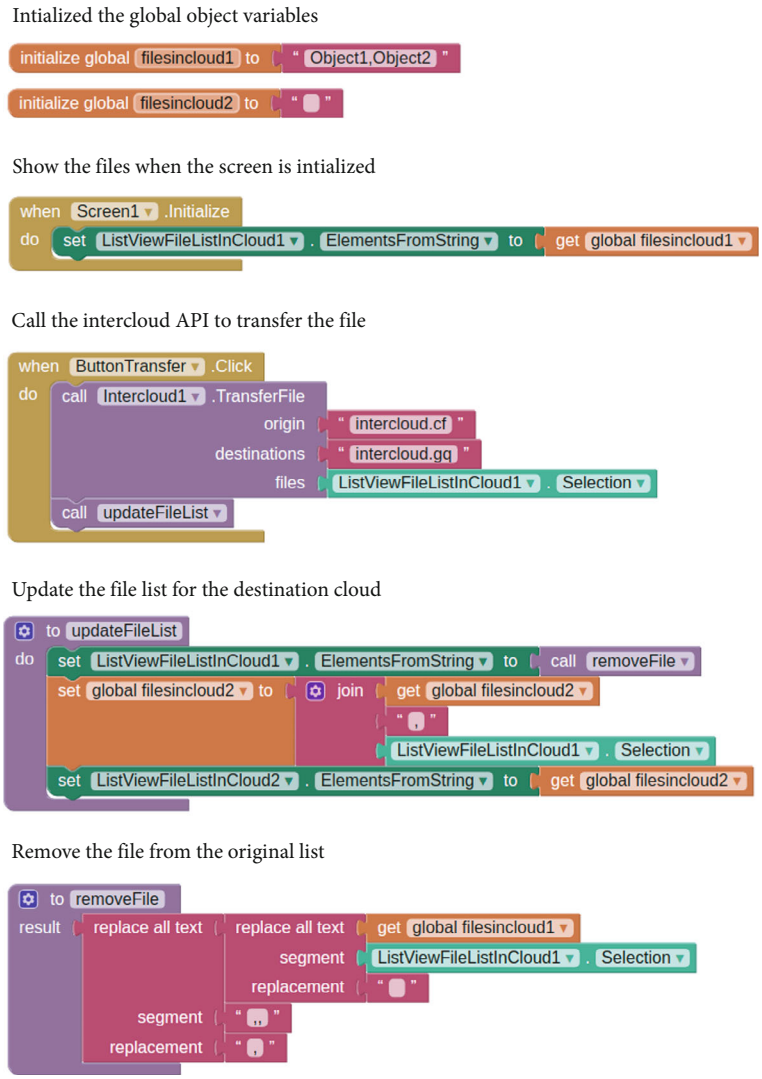

Figure 8: App inventor blocks for an ICCP-related mobile app.

to defining link costs in the case of a routing problem (e.g., for Dijkstra's algorithm). Furthermore, by using general costs, different users can define their own costs for their optimization problem. Basically, at state $s_{h,u}$, if the action is "stay," the cost is h (i.e., using the application in the central cloud). If the action is "move," the cost is $C + f$. Note that we have $h \geq f$, as it is obviously more efficient to run the application in the edge cloud when the user is in the edge cloud. However, since there is a transfer cost, it is not always desirable to move the application. The aim of the following analysis is to determine under what circumstances the application should be moved (i.e., to determine the decision policy).

4.1.1. Decision Policy 1. Under the finite-horizon Markov decision process, the optimal decision is to "move" if the duration for staying in the edge cloud is

$$\tau > \frac{C + f - h}{ph - pf} + 1. \tag{1}$$

Proof. For the finite-horizon Markov decision process, the backward induction algorithm can be used to determine the decision policy [47]. In other words, we can use the backward induction algorithm to determine whether an application should be moved if the duration of staying in the edge cloud is known. Here, our focus is to determine a closed-form solution. Let $\theta^t(s)$ be the minimum expected accumulated cost for state s at the decision time slot t. Following the notations for formulating a Markov decision model in [47], $\theta^t(s)$ and $\theta^{t+1}(s)$ are related by the following formula:

$$\theta^t(s) = \min \left\{ \begin{array}{l} \text{Cost}^t(s, a_{\text{move}}) + \sum_{s \in S} P\left(s^{t+1} \mid s, a_{\text{move}}\right) \cdot \theta^{t+1}\left(s^{t+1}\right), \\ \text{Cost}^t\left(s, a_{\text{stay}}\right) + \sum_{s \in S} P\left(s^{t+1} \mid s, a_{\text{stay}}\right) \cdot \theta^{t+1}\left(s^{t+1}\right) \end{array} \right\}. \tag{2}$$

According to [47], this kind of decision model should have a threshold-based decision policy. In our case, assuming

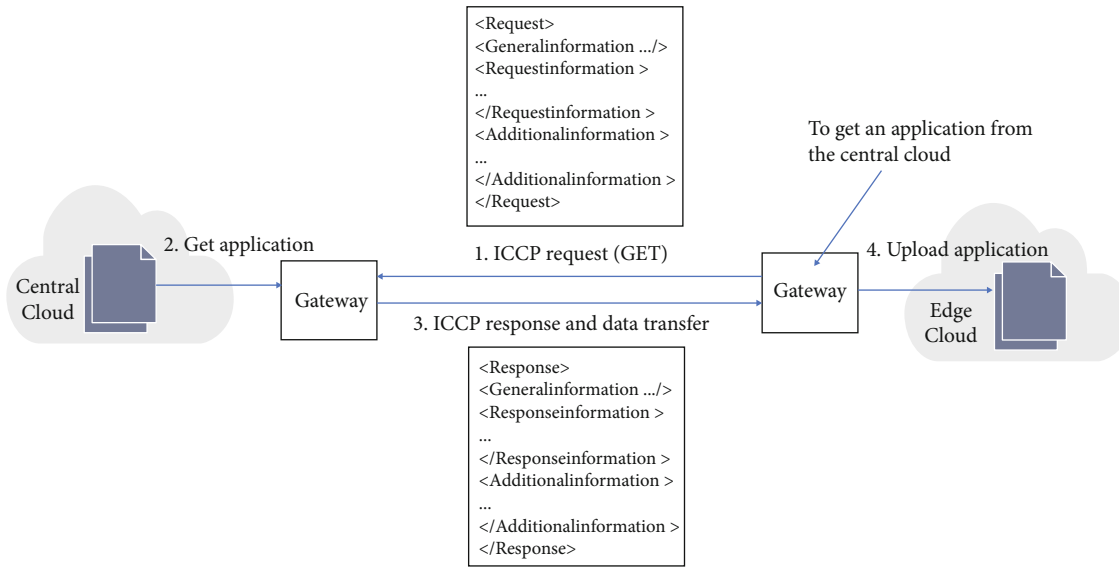

Figure 9: Gateway gets an application with the ICCP GET command.

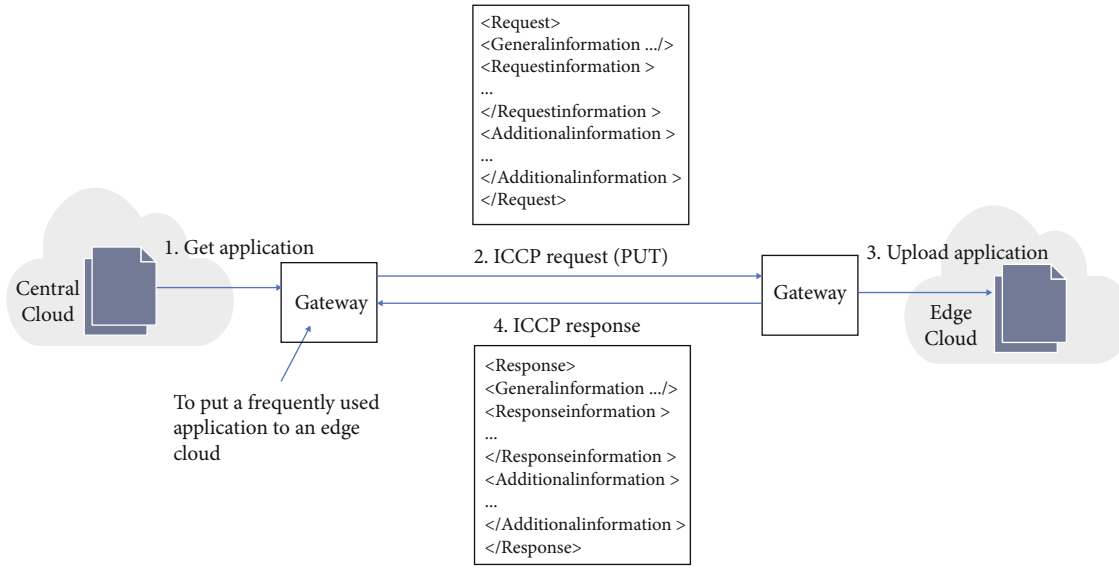

Figure 10: Gateway puts an application with the ICCP PUT command.

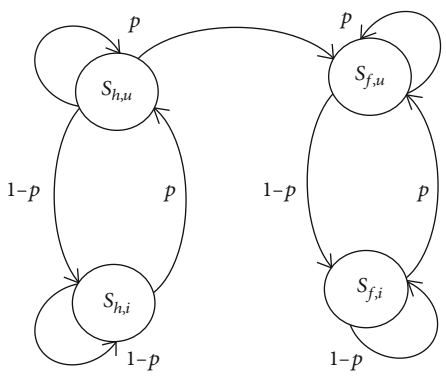

Figure 11: Finite-horizon Markov decision process states.

that the last time slot is N and working backward, the "stay" decision should be applied for N, $N-1$, and $N-\tau$. Note that working backward, the initial decision should obviously be "stay." The aim is to determine when the "stay" decision should end (i.e., at $N-\tau$).

When the decision is "stay" (i.e., do not move), the system will change from state $s_{h,u}$ to a state of $s_{h,u}$ or $s_{h,i}$. Then, the expected cost for state $s_{h,u}$ at time t can be computed as follows:

$$\theta^{t}_{\text{stay}}(s_{h,u}) = h + p\theta^{t+1}_{\text{stay}}(s_{h,u}) + (1-p)\theta^{t+1}_{\text{stay}}(s_{h,i}). \quad (3)$$

Note that in equation (3), we add the subscript "stay" to indicate the corresponding action (i.e., "stay"). For clarity, this notation will also be applied to equations (4)–(9).

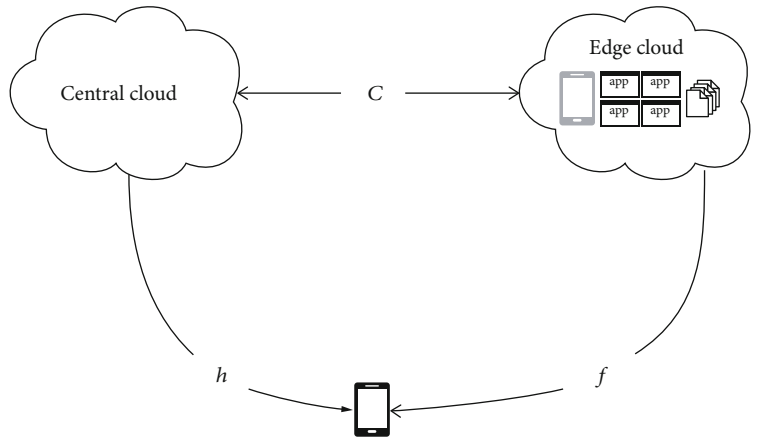

Figure 12: Costs and relationship for the Markov decision model.

Similarly, the expected cost for state $s_{h,i}$ at time t can be computed as follows:

$$\theta^{t}_{\text{stay}}(s_{h,i}) = p\theta^{t+1}_{\text{stay}}(s_{h,u}) + (1-p)\theta^{t+1}_{\text{stay}}(s_{h,i}). \tag{4}$$

Working backward, assuming that each decision is "stay" (i.e., between $N-\tau$ and N), the expected cost at each time slot can be calculated as shown in Table 5.

Hence, we have

$$\theta^{N-\tau}_{\text{stay}}(s_{h,u}) = h + (\tau-1)ph. \tag{5}$$

We assume that at time slot $N-\tau$, the optimal decision becomes "move." When the decision is "move," the system will change from state $s_{h,u}$ to state $s_{f,u}$ or $s_{f,i}$. In this case, we have the following relationships:

$$\begin{aligned}
\theta^{t}_{\text{move}}(s_{h,u}) &= C + f + p\theta^{t+1}_{\text{move}}(s_{f,u}) + (1-p)\theta^{t+1}_{\text{move}}(s_{f,i}),\\
\theta^{t}_{\text{move}}(s_{f,u}) &= f + p\theta^{t+1}_{\text{move}}(s_{f,u}) + (1-p)\theta^{t+1}_{\text{move}}(s_{f,i}),\\
\theta^{t}_{\text{move}}(s_{f,i}) &= p\theta^{t+1}_{\text{move}}(s_{f,u}) + (1-p)\theta^{t+1}_{\text{move}}(s_{f,i}).
\end{aligned} \tag{6}$$

As mentioned above, note that we add the subscript "move" to indicate the corresponding action (i.e., "move"). Using the backward induction, assuming that each decision is "move" (i.e., between $N-\tau$ and N), the expected cost at each time slot can be calculated as shown in Table 6.

Based on the above equations, if the optimal decision is to "move" at time slot $N-\tau$, we have

$$\theta^{N-\tau}_{\text{move}}(s_{h,u}) = C + f + (\tau-1)pf. \tag{7}$$

Note that once the application is moved, the subsequent actions or decisions are then always "stay." By comparing equations (5) and (7), it can be found that if the following condition is satisfied, the optimal decision is to "move" at time slot $N-\tau$.

$$\tau > \frac{C+f-h}{ph-pf} + 1. \tag{8}$$

That means if the duration τ for staying in the edge cloud is larger than $((C+f-h)/(ph-pf)) + 1$, the decision should be "move"; otherwise, the application should "stay" in the central cloud.

Table 5

Time slot	$\theta^{t}_{\text{stay}}(s_{h,u})$	$\theta^{t}_{\text{stay}}(s_{h,i})$
N	0	0
$N-1$	h	0
$N-2$	$h+ph$	ph
$N-3$	$h+2ph$	$2ph$
...	...	...
$N-\tau$	$\boldsymbol{h+(\tau-1)ph}$	$(\tau-1)ph$

Table 6

Time slot	$\theta^{t}_{\text{move}}(s_{h,u})$	$\theta^{t}_{\text{move}}(s_{f,u})$	$\theta^{t}_{\text{move}}(s_{f,i})$
N	0	0	0
$N-1$	C	f	0
$N-2$	$C+f+pf$	$f+pf$	pf
$N-3$	$C+f+2pf$	$f+2pf$	$2pf$
...	...	...	...
$N-\tau$	$\boldsymbol{C+f+(\tau-1)pf}$	$f+(\tau-1)pf$	$(\tau-1)pf$

4.1.2. Decision Policy 2. Under the finite-horizon Markov decision model, a "move" decision should be made with the following probability:

$$\rho = b^{((C+f-h)/(ph-pf))+1}. \tag{9}$$

where b is the probability that the mobile terminal will stay in the edge cloud.

Proof. In the above finite-horizon Markov decision model, the duration for staying in an edge cloud is considered to be a constant and is known in advance. In practice, a user may leave an edge cloud (e.g., go to the end state with probability $1 - b$ at each time slot). In this case, the duration time will no longer be a constant. The aforementioned decision policy can in fact be turned into a probabilistic policy, as explained below. Assuming a Bernoulli process (i.e., remain in and leave the edge cloud with probability b and $1 - b$, respectively), it can be shown that the duration for staying in the edge cloud follows a geometric distribution, as shown in Table 7. □

Based on the above closed-form solution, when the duration n is greater than $((C + f - h)/(ph - pf)) + 1$, the decision should be "move." Hence, we can calculate the probability of making a "move" decision by summing the probabilities that the duration of remaining in the edge cloud is larger than $((C + f - h)/(ph - pf)) + 1$, as shown below:

$$\rho = \sum_{n=((C+f-h)/(ph-pf))+2}^{n=\infty} b^{n-1}(1-b) = b^{((C+f-h)/(ph-pf))+1}. \tag{10}$$

Therefore, it becomes a probabilistic policy in which a "move" decision should be made with probability $b^{((C+f-h)/(ph-pf))+1}$. Algorithm 1 outlines the decision-making algorithm for the finite-horizon Markov decision process.

4.2. Infinite-Horizon Markov Decision Process. Next, we formulate an infinite-horizon Markov decision model. As shown in Figure 13, there are five states $\{s_{h,u}, s_{h,i}, s_{f,u}, s_{f,i}, s_{\text{end}}\}$ and two actions $\{a_{\text{stay}}, a_{\text{move}}\}$, where state s_{end} means that the session is terminated (i.e., leave the edge cloud). The user will remain in the edge cloud with probability b (i.e., leave with probability $1 - b$ or go to the end state).

4.2.1. Decision Policy 3. Under the infinite-horizon Markov decision model, a "move" decision should be made if $\psi(\text{move}) < \psi(\text{stay})$ where $\psi(\text{move}) = h + (bph/(1 - b))$ and $\psi(\text{stay}) = C + f + (bpf/(1 - b))$.

Proof. For the infinite-horizon Markov decision process, the optimal decision at each time slot is the same. If the decision is "move," there is a transfer cost of C and the application can be used at a cost of f. In the subsequent time slots (until the end), the application can be used at a cost of f. Hence, the expected cost for each time slot is pf. If the decision is "stay," the application is then used at a cost of h. Hence, the expected cost for each time slot is ph.

Assuming that the decision is "move," we can obtain the expected cost of different durations for staying in the edge cloud as shown in Table 8.

Table 7

Duration	Probability
1	$1 - b$
2	$b(1 - b)$
3	$b^2(1 - b)$
4	$b^3(1 - b)$
...	...
n	$b^{n-1}(1 - b)$
...	...

Therefore, if the decision is "move," the overall expected cost can be computed as follows:

$$\psi(\text{move}) = \sum_{n=1}^{\infty} b^{n-1}(1-b)[C + f + (n-1)pf] = C + f + \frac{bpf}{1-b}. \tag{11}$$

Similarly, if the decision is "stay," the expected cost of different durations can be computed as shown in Table 9.

Therefore, if the decision is "stay," the overall expected cost can be computed as follows:

$$\psi(\text{stay}) = \sum_{n=1}^{\infty} b^{n-1}(1-b)[h + (n-1)ph] = h + \frac{bph}{1-b}. \tag{12}$$

If the optimal decision is "move," the overall expected cost of moving the application should be less than that of not moving the application. Hence, we have

$$\psi(\text{move}) = C + f + \frac{bpf}{1-b} < \psi(\text{stay}) = h + \frac{bph}{1-b}. \tag{13}$$

In summary, according to equation (11), we can decide whether to move an application based on the following threshold conditions:

$$\begin{aligned} f &< h - \frac{C(1-b)}{1-b+bp}, \\ C &< \frac{(h-f)(1-b+bp)}{1-b}, \\ p &> \frac{(C+f-h)(1-b)}{b(h-f)}, \\ b &> \frac{C+f-h}{C+f-h+p(h-f)}. \end{aligned} \tag{14}$$

These closed-form threshold conditions are useful for making general migration decisions, which will be evaluated in the next section. In general, it is preferable to "move" the application when the usage probability p is high, the probability b is high, the transfer cost C is small, and the running cost in the edge cloud f is small. In summary, Algorithm 2 illustrates the decision-making mechanism for the IHMDP model.

Input: fh_state, C, f, h and p, b;
// fh_state is the current state
Output: fh_action; // action to be taken
1 $fh\_r \longleftarrow random.uniform\ (0,1)$; //random value
2 $\tau \longleftarrow (C + f - h)/(pxh - pxf) + 1$;
3 if $\tau < 1$ **then**
4 $\tau \longleftarrow 0$
5 else
6 $\tau \longleftarrow math.floor(\tau)$; //turn to discrete time
7 end
8 if $fh\_state \in \{s_{h,i}, s_{f,u}, s_{f,i}\}$ **then**
$fh\_action \longleftarrow$ "$stay$";
9 else if $fh\_state = s_{h,u}$ and
fh_is_$move = True$ the
10 $fh\_action \longleftarrow$ "$stay$"
11 else if $fh\_r < pow(b, \tau)$ then
12 $fh\_action \longleftarrow$ "$move$";
13 fh_is $move \longleftarrow$ True;
14 else
15 $fh\_action \longleftarrow$ "$stay$";
16 fh_is_$move \longleftarrow$ False;
17 end

Algorithm 1: Decision-making algorithm for the finite-horizon Markov decision process.

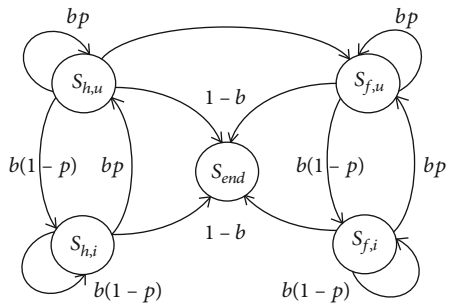

Figure 13: State transition diagram.

Table 8

Duration	Probability	Expected cost
1	$1 - b$	$C + f$
2	$b(1 - b)$	$C + f + pf$
3	$b^2(1 - b)$	$C + f + 2pf$
...	...	...
n	$b^{n-1}(1 - b)$	$C + f + (n - 1)pf$
...	...	...

Table 9

Duration	Probability	Expected cost
1	$1 - b$	h
2	$b(1 - b)$	$h + ph$
3	$b^2(1 - b)$	$h + 2ph$
...	...	...
n	$b^{t-1}(1 - b)$	$h + (t - 1)ph$
...	...	...

In general, the major contribution of this paper is to consider various Markov decision models to cater for different situations and to derive close-form solutions for the decision policies.

(i) FHMDP (deterministic scheme): this scheme can be used if the duration of staying with an edge cloud can be estimated. The advantage of this scheme is that the transfer decision can be determined when joining the edge cloud

(ii) FHMDP (probabilistic scheme): this scheme can be used if the duration of staying with an edge cloud cannot be estimated but the probability of leaving an edge cloud can be estimated. However, the decisions are probabilistic not deterministic

(iii) IHMDP: this can be viewed as the general scheme, which can be used if the probability of leaving an edge cloud can be estimated. This scheme can combine the advantages of the aforementioned schemes. Like FHMDP (deterministic scheme), the policy is deterministic. Like FHMDP (probabilistic scheme), there is no need to consider a fixed duration of staying with the edge cloud. Furthermore, the general IHMDP scheme is more flexible as it can be extended to cover other scenarios if appropriate Markov states can be defined

Input: fh_state, C, f, h and p, b;
// fh_state is the current state
Output: fh_action; // action to be taken
1 $\psi(move) \longleftarrow C + f + (bxpxf)/(1-b)$;
2 $\psi(stay) \longleftarrow C + f + (b \times p \times f)/(1-b)$;
//**expect cost**
3 if $fh\_state \in \{s_{h,i}, s_{f,u}, s_{f,i}\}$ **then**
$ih\_action \longleftarrow$ "$stay$";
4 else if $ih\_state = s_{h,u}$ **and** $ih\_is\_move = True$
then
5 $ih\_action \longleftarrow$ "$stay$"
6 else if $\psi(move) < \psi(stay)$ then
7 $ih\_action \longleftarrow$ "$move$";
8 $ih\_is\_move \longleftarrow$ True;
9 else
10 $ih\_action \longleftarrow$ "$stay$";
11 $ih\_is\_move \longleftarrow$ False;
12 end

Algorithm 2: Decision-making algorithm for the infinite-horizon Markov decision process.

5. Performance Analysis

In this section, we conduct extensive evaluation for the aforementioned methods/algorithms in three parts. In the first part, we present experimental results to evaluate the functions and basic performance through an experimental prototype for proof-of-concept purposes. In the second part, we present the analytical results with the aim of analyzing the aforementioned formulas (i.e., the threshold values of parameters for changing the decisions in different cases). In the third part, we present the simulation results to evaluate the performance of the finite-horizon Markov decision process (FHMDP) and infinite-horizon Markov decision process (IHMDP). Note that we have conducted extensive simulations for the performance analysis. In particular, simulation results for a multiuser and multicloud environment are presented. The following results are representative examples, which show the general trend and major findings.

5.1. Experimental Evaluation. First, we evaluate the basic concept of the mobile Intercloud system through an experimental prototype (i.e., for proof-of-concept purposes) as mentioned in Section 2. The focus is on the object transfer functions. In the first experiment, we evaluated the downloading/transferring of four objects (1 Mbyte, 10 Mbytes, 100 Mbytes, and 1000 Mbytes) to a mobile terminal at HK based on two methods (see Figure 2), assuming that there was a central cloud at U.S. (cloud B in Figure 2) and an edge cloud (cloud A in Figure 2) at HK. In the first method, the objects were downloaded/transferred directly to the mobile terminal from the U.S. cloud (central cloud). In the second method, the objects were first transferred to the HK cloud (edge cloud) and then downloaded/transferred to the mobile terminal from the HK cloud. In this case, the object transfer between the HK cloud and U.S. cloud (i.e., through two Intercloud gateways) was conducted using a mobile app developed by App Inventor with Intercloud extension blocks (see Figure A1). As shown in Table 10, for method 1, the objects were downloaded/transferred successfully with a mean download/transfer time of 2.75 s, 11.5 s, 128.25 s, and 1236.25 s, respectively. For method 2, the objects were downloaded/transferred successfully with a mean download/transfer time of 4.5 s, 6.75 s, 23.5 s, and 162 s, respectively. It can be seen that method 2 (i.e., mobile Intercloud approach) is more efficient for larger objects. For example, for the object of 1000 Mbytes, the mean download/transfer time can be reduced significantly by more than 87% (i.e., comparing between method 1 and method 2). This illustrates the benefit of using mobile Intercloud in certain cases. In addition, we have also conducted other test cases such as the second one and third one in Table 10 to test some basic functions of mobile Intercloud (i.e., object transfer between clouds for edge cloud computing). For example, the second test case seeks to demonstrate that an object transfer can be initiated based on a predefined policy or condition (e.g., based on equation (12)). Similarly, the third test case seeks to demonstrate that an object can be transferred based on its usage time.

Next, we evaluate the basic interactions and functions of physical and virtual mobile terminals through the second part of the experimental prototype. As a generic model (see Figure 14), to run an application, a physical mobile terminal interacts with a virtual mobile terminal (similar to a server) through i interactions over the Internet. In each interaction, d bytes of data are exchanged. At both the physical and virtual mobile terminals, m and s program loops are executed, respectively. In general, the physical mobile terminal performs light processing only. Intensive processing tasks are handled at the virtual mobile terminal. To simulate intensive processing at the virtual mobile terminal, a large file is encrypted at each program loop (i.e., as an example to simulate the application). The virtual mobile terminal can be situated in a U.S. (central) cloud or Hong Kong (edge) cloud.

Table 10: Summary of the test cases and results.

Test case	Result
Object 1 (1 Mbyte), object 2 (10 Mbytes), object 3 (100 Mbytes), and object 4 (1000 Mbytes) were downloaded by a mobile phone using the following methods (see Figure 2): Method 1: download directly from the central cloud (U.S.) Method 2: transfer from the central cloud (U.S.) to the edge cloud (HK) and then download from the edge cloud (HK)	For method 1, the mean download time for object 1, object 2, object 3, and object 4 was 2.75 s, 11.5 s, 128.25 s, and 1236.25 s, respectively For method 2, the mean transfer and download time for object 1, object 2, object 3, and object 4 was 4.5 s, 6.75 s, 23.5 s, and 162 s, respectively
Suppose that the system parameters are $h = 100$, $C = 200$, $b = 0.9$, and $p = 0.5$. $f = 50$, 60, and 70 for applications 1, 2, and 3, respectively	According to equation (12), applications 1 and 2 were transferred successfully through the gateways
Applications 1, 2, and 3 were lastly used 10, 4, and 2 days ago, respectively. The gateway for the U.S. cloud should transfer applications to the HK cloud (i.e., using the ICCP PUT command) if the applications were used within the last y days	For $y = 3$, application 3 was transferred successfully For $y = 5$, applications 2 and 3 were transferred successfully For $y = 10$, applications 1, 2, and 3 were transferred successfully

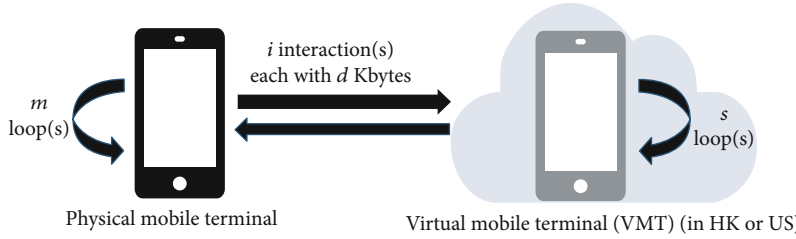

Figure 14: Generic model of the experiment.

That means the application can be run in the central cloud or edge cloud (i.e., by transferring the application from the U.S. cloud to HK cloud).

Figure 15 shows the mean processing time for $i = 1$ and $i = 10$ when $m = 1$ and $s = 1$. It indicates that the mean processing time can be kept within a low value (a few seconds) when $i = 1$ even if the application is run at the U.S. cloud (i.e., there is no need to transfer the application to the HK cloud). When $i = 10$ (i.e., more interactions between physical and virtual mobile terminals), it becomes more necessary to transfer the application to the HK cloud to maintain performance. Figure 16 shows the result when m and s are increased to 10. It can be seen that the pattern is similar to the previous one, indicating that the mean processing time is relatively insensitive to the change in m and s provided that there are sufficient internal resources (i.e., the mean processing time depends more heavily on i and d). Figure 17 shows the mean processing time for $s = 1$, $s = 10$, and $s = 20$ when $m = 1$ and $i = 1$. It indicates that the mean processing time is insensitive to the change in s when there are sufficient internal resources for running the application or program tasks at the virtual mobile terminal. When s is increased to 20, the mean processing time is no longer insensitive to the change in s. In other words, if a certain threshold is reached (i.e., the internal resources become limited), the mean processing time is increased more significantly. Note that as the virtual mobile terminal is cloud-based, resources can easily be dynamically adjusted. Figure 18 shows the mean processing time for $d = 100$ and $d = 1000$ when $m = 10$ and $i = 10$. To keep the mean processing time to a low level, it can be seen that it is better to transfer the application to the Hong Kong cloud, especially when the exchanged data size is large.

In summary, the mean processing time is mostly affected by the number of interactions and the exchanged data size between the physical and virtual mobile terminals. Therefore, to enhance the user experience, it is important to reduce the number of interactions and exchanged data size, such as by duplicating frequently used data and processes at both physical and virtual mobile terminals. If the virtual mobile terminal is provided with sufficient resources, the mean processing time is relatively insensitive to the change in s. However, when a certain threshold is reached, the mean processing time can be increased substantially, affecting the user experience. This reflects the importance of allocating sufficient resources at the virtual mobile terminal. The aforementioned analysis also indicates that in some cases, it is desirable to transfer the virtual mobile terminal to the local cloud to minimize mean processing time, so the user experience can be enhanced. Further analysis will be conducted in the following sections.

5.2. Analytical Evaluation. In this section, we evaluate the analytical formulas or closed-form solutions, particularly the effect of some key parameters.

5.2.1. Infinite-Horizon Markov Decision Model. Based on the closed-form solutions derived from the IHMDP model in equation (14), we analyze how the parameters influence the decision (i.e., "move" or "stay") and expected total cost in Figures 19–24, respectively.

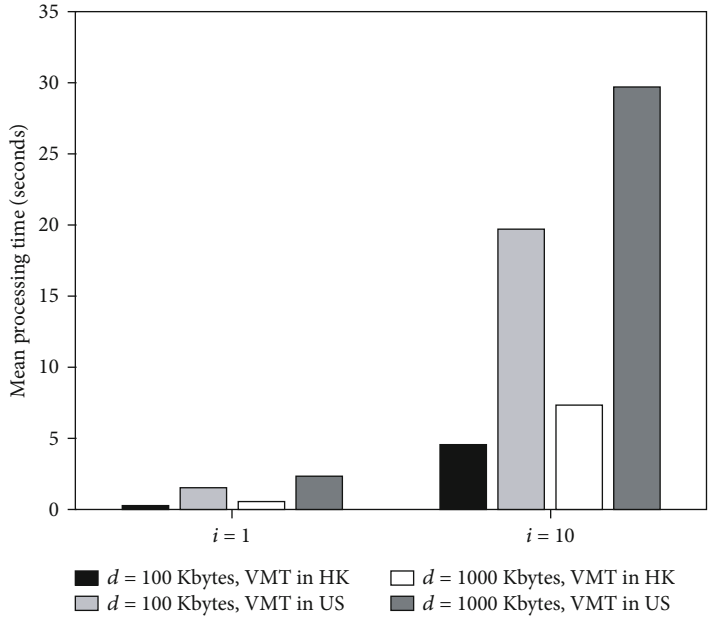

Figure 15: Mean processing time for $i = 1$ and $i = 10$ when $m = 1$ and $s = 1$.

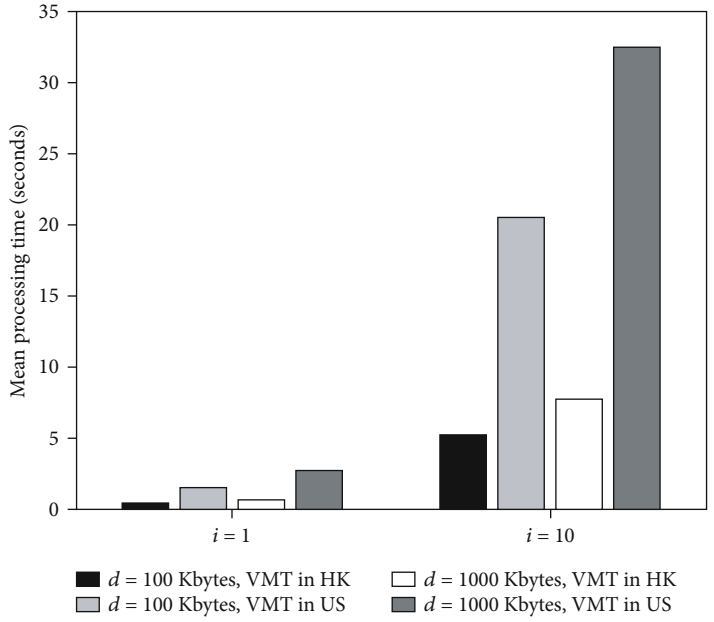

Figure 16: Mean processing time for $i = 1$ and $i = 10$ when $m = 10$ and $s = 10$.

Figures 19 and 20 show a similar trend in the threshold value of probability b. They indicate that the threshold value is relatively constant when the probability p is large. This is because as the usage probability is high, similar decisions should be made. Compared with C, the threshold value of b is more sensitive to the change of f. In other words, the change in the cost of using the application at the edge cloud (i.e., f) has a larger effect than the change in transfer cost (i.e., C). This is because C is a one-off transfer cost, but f is a recurrent usage cost, so its effect will last for the remainder of the duration of staying in the edge cloud, as determined by b.

Figures 21 and 22 show that the threshold values of C decrease linearly as f increases for different probabilities (i.e., b as well as p). Furthermore, as f increases, the threshold values of C tend to converge as shown in both figures. This is because when f is large, the decision should be "stay" irrespective of the usage probability or the expected duration of staying at the edge cloud (i.e., whenever the transfer cost is larger than the converged value as shown in the figure). Compared with the probability p, the threshold value of C is more sensitive to the change of b. It shows that b (i.e., expected duration for

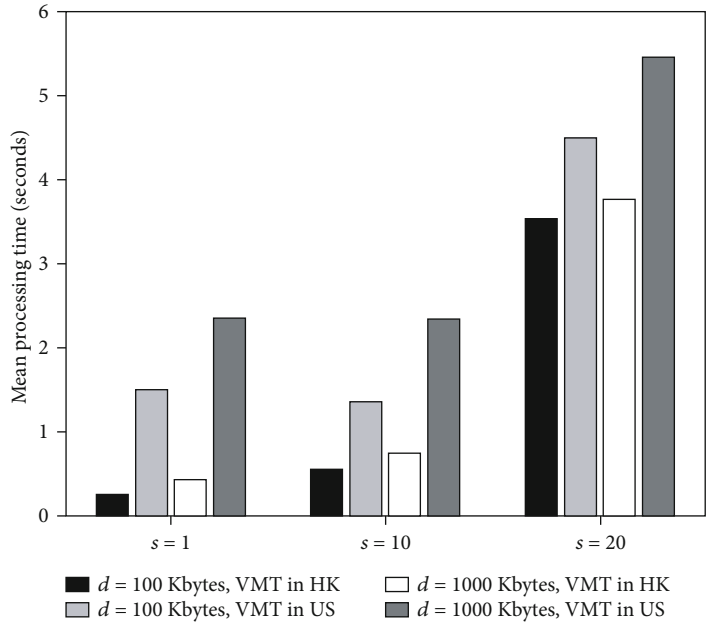

Figure 17: Mean processing time for $s = 1$, $s = 10$, and $s = 20$ when $m = 1$ and $i = 1$.

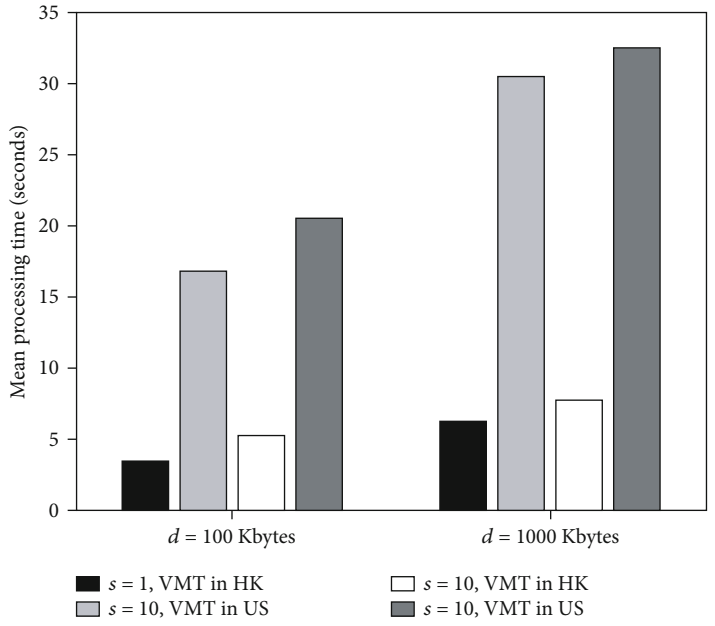

Figure 18: Mean processing time for $d = 100$ and $d = 1000$ when $m = 10$ and $i = 10$.

staying at the edge cloud) has a larger effect on the decision than p (i.e., the usage probability).

We then analyze how the parameters influence the expected total cost. Figure 23 illustrates that f has a greater influence than C on the expected total cost. The expected total cost increases linearly as f increases but flattens when f increases to a certain value. Eventually, all curves converge to a certain value because when f is large, it is better not to move. Hence, the cost becomes constant. That means if the cost of using the application at the edge cloud is large, the decision tends to be "stay." In this case, the expected total cost is not dependent on the transfer cost but is only dependent on the cost of using the application at the central cloud (i.e., a constant as shown in the figure). Figure 24 shows that the expected total cost increases exponentially as b as well as p increases toward one. However, all curves converge to the same low cost initially. This is because when the expected duration for staying at the edge cloud is low (i.e., small value of b), the preferred decision is "stay" irrespective of the usage probability.

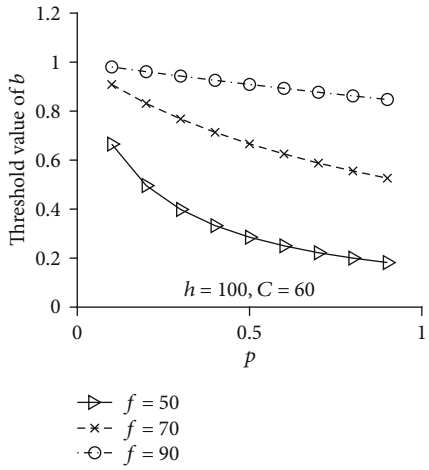

Figure 19: Influence of cost f on the threshold value of b in IHMDP.

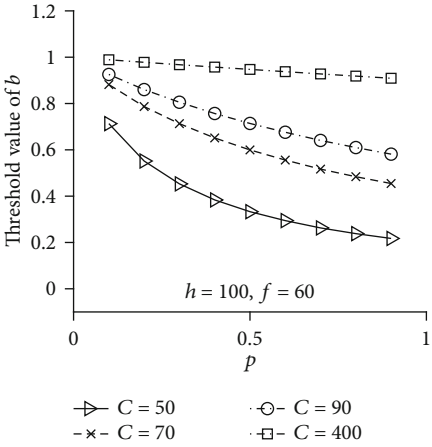

Figure 20: Influence of cost C on the threshold value of b in IHMDP.

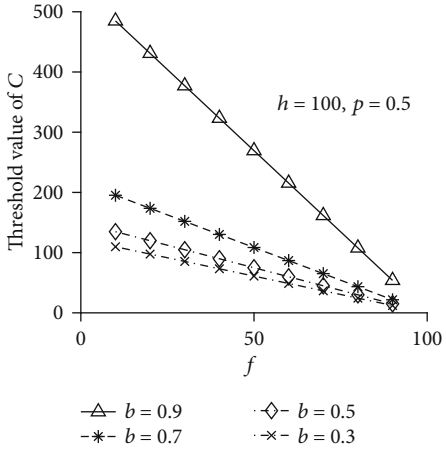

Figure 21: Influence of probability b on the threshold value of C in IHMDP.

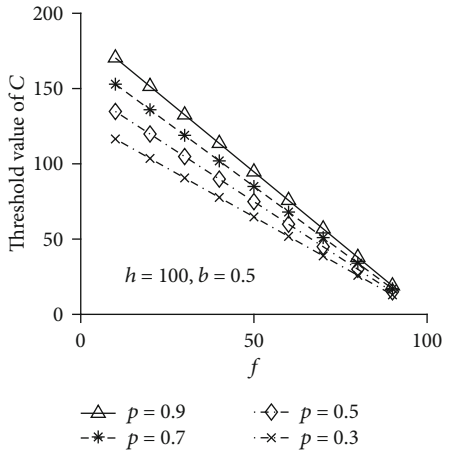

Figure 22: Influence of probability p_1 and p_2 on the threshold value C in IHMDP.

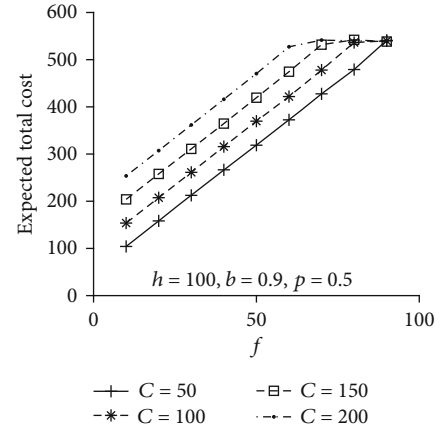

Figure 23: Influence comparison between cost C and f in IHMDP.

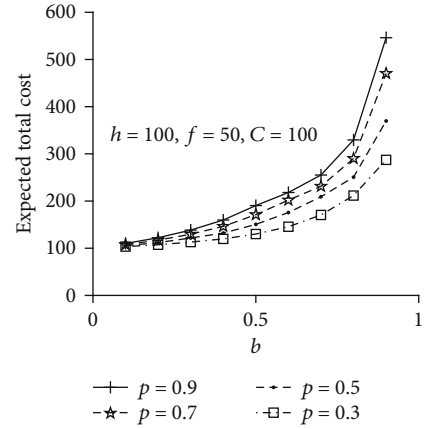

Figure 24: Influence comparison between probabilities p_1 and p_2 and b in IHMDP.

5.2.2. Finite-Horizon Markov Decision Model. Based on the closed form solutions derived from the FHMDP model in equations (8) and (10), we analyze the threshold time and the probability of moving the application to the edge cloud

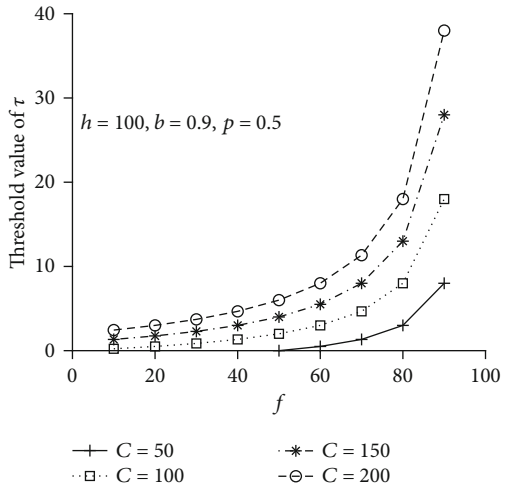

Figure 25: Influence of cost C on the threshold value τ in FHMDP.

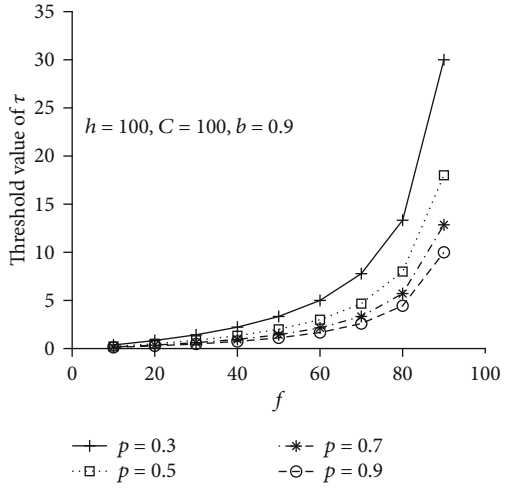

Figure 26: Influence of probability p on the threshold value τ in FHMDP.

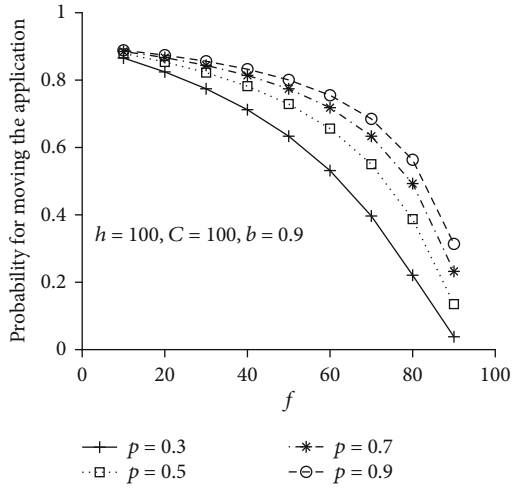

Figure 27: Influence of probability p on the probability of moving the application.

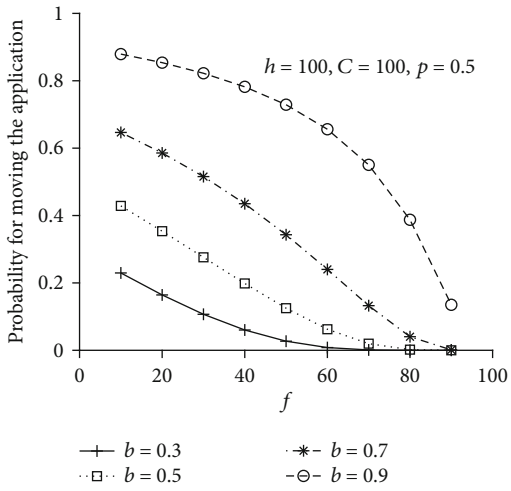

Figure 28: Influence of probability b on the probability of moving the application.

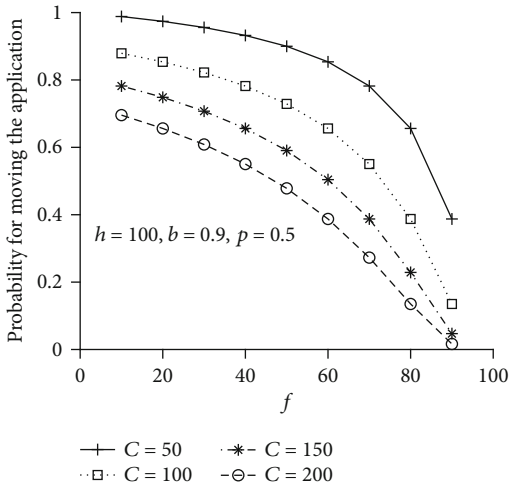

Figure 29: Influence of cost C on the probability of moving the application.

under different parameter settings in Figures 25–29, respectively.

Figure 25 shows that the threshold value of τ increases exponentially with the increase in f and C. This means that the application should stay at the central cloud longer (i.e., τ is larger), when the cost of transferring the application and running it at the edge cloud is large. On the contrary, Figure 26 shows that the threshold value of τ decreases with the increase in p. This indicates that when the application has a high usage probability (i.e., p close to one), it should be transferred earlier as expected.

As mentioned before, for the FHMDP model, a probabilistic policy can also be formulated by equation (10) (i.e., transferring the application with a certain probability). Figures 27–29 show that the probability of moving the application decreases when f increases for different values of C, b, and p. As shown in Figure 27, the probability decreases more dramatically when p is small. That means if the application is

Table 11: Influence of cost f on the decision.

$h = 100$, $C = 200$, $b = 0.9$, $p = 0.5$	FHMDP (moving probability)	IHMDP
$f = 10$	0.696	Move
$f = 20$	0.656	Move
$f = 30$	0.609	Move
$f = 40$	0.550	Move
$f = 50$	0.478	Move
$f = 60$	0.387	Move
$f = 70$	0.273	Stay
$f = 80$	0.135	Stay
$f = 90$	0.016	Stay

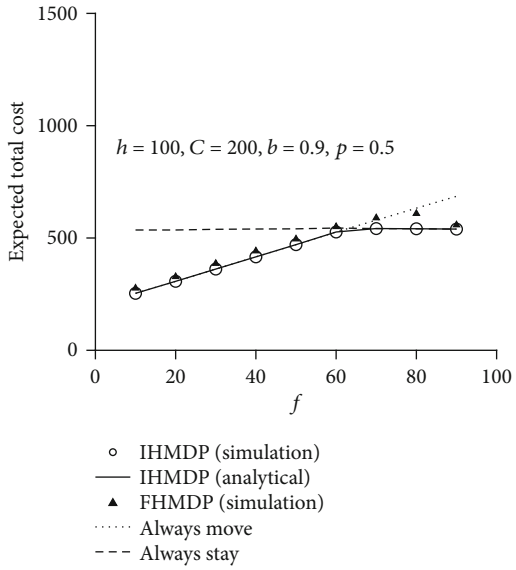

Figure 30: Influence of cost f on the expected total cost in IHMDP and FHMDP.

Table 12: Influence of cost C on the decision.

$f = 50$, $h = 100$, $b = 0.9$, $p = 0.5$	FHMDP (moving probability)	IHMDP
$C = 100$	0.729	Move
$C = 200$	0.478	Move
$C = 300$	0.314	Stay
$C = 400$	0.206	Stay
$C = 500$	0.135	Stay
$C = 600$	0.089	Stay
$C = 700$	0.058	Stay
$C = 800$	0.038	Stay
$C = 900$	0.025	Stay
$C = 1000$	0.016	Stay

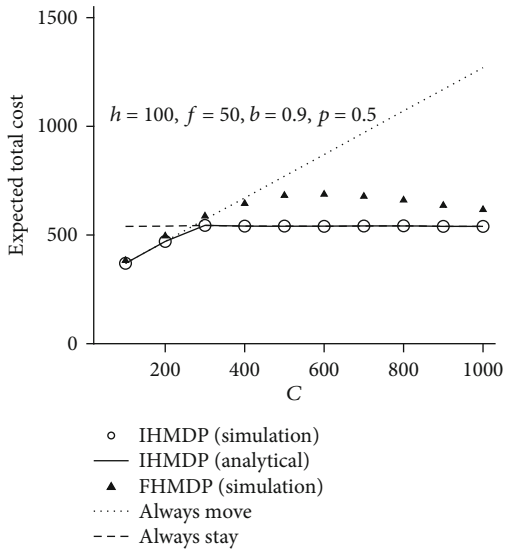

Figure 31: Influence of cost C on the expected total cost in IHMDP and FHMDP.

Table 13: Influence of probability b on the decision.

$f = 50$, $h = 100$, $C = 100$, $p = 0.5$	FHMDP (moving probability)	IHMDP
$b = 0.1$	0.001	Stay
$b = 0.2$	0.008	Stay
$b = 0.3$	0.027	Stay
$b = 0.4$	0.064	Stay
$b = 0.5$	0.125	Stay
$b = 0.6$	0.216	Stay
$b = 0.7$	0.343	Move
$b = 0.8$	0.512	Move
$b = 0.9$	0.729	Move

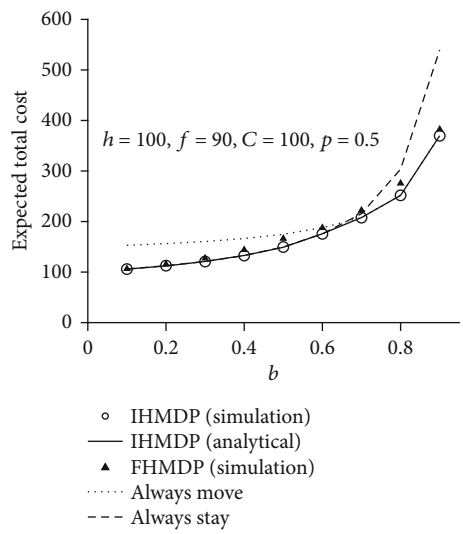

Figure 32: Influence of probability b on the expected total cost in IHMDP and FHMDP.

Table 14: Influence of probability p on the decision.

$f = 50$, $h = 100$, $C = 200$, $b = 0.9$	FHMDP (moving probability)	IHMDP
$p = 0.1$	0.038	Stay
$p = 0.2$	0.185	Stay
$p = 0.3$	0.314	Stay
$p = 0.4$	0.408	Move
$p = 0.5$	0.478	Move
$p = 0.6$	0.531	Move
$p = 0.7$	0.573	Move
$p = 0.8$	0.606	Move
$p = 0.9$	0.633	Move

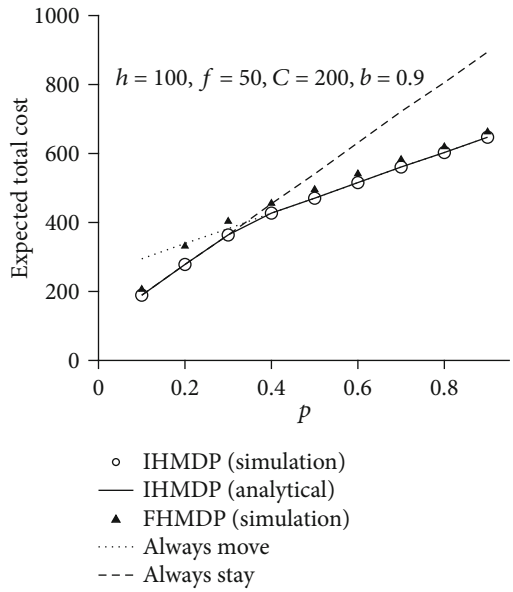

Figure 33: Influence of probability p on the expected total cost in IHMDP and FHMDP.

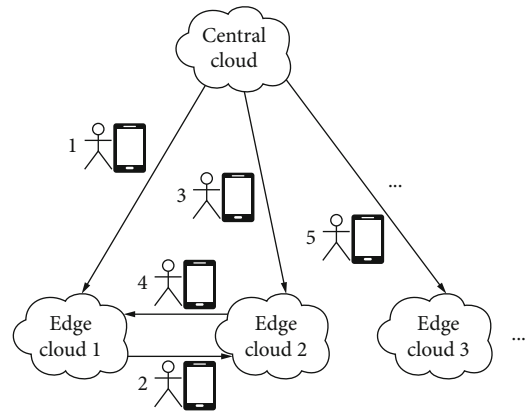

Figure 34: Simulation model.

Table 15: Simulation parameters.

Number of users	10,000
Number of edge clouds	40
Types of users	UT1—moderate usage, low mobility ($p = 0.5$, $b = 0.9$) UT2—high usage, low mobility ($p = 0.9$, $b = 0.9$) UT3—moderate usage, high mobility ($p = 0.5$, $b = 0.1$) UT4—high usage, high mobility ($p = 0.9$, $b = 0.1$)
Types of edge clouds	CT1—low usage cost, low transfer cost ($f = 10$, $C = 10$) CT2—high usage cost, low transfer cost ($f = 50$, $C = 10$) CT3—low usage cost, high transfer cost ($f = 10$, $C = 1000$) CT4—high usage cost, high transfer cost ($f = 50$, $C = 1000$)

not used frequently, we should transfer the application with a lower probability. Furthermore, Figure 28 indicates that the probability decreases more steeply when b is small, which shows that b also has a greater impact on the decision than p. In other words, even if the usage is high, it is still not preferable to transfer the application if the expected duration for staying at the edge cloud is short. Figure 29 shows that when C becomes large, the probability also decreases drastically, indicating that the application should not be transferred due to the high transfer cost.

5.3. Simulation Evaluation

5.3.1. Single Edge Cloud and Single User. As discussed above, the decision policy depends on three major factors, namely, (1) costs: C, f, and h; (2) usage probability p; and (3) the expected duration of remaining in the edge cloud, as determined by the probability b. We have conducted extensive simulations and evaluations to analyze the performance of the IHMDP and FHMDP models compared to the "ALWAYS MOVE" and "ALWAYS STAY" policies (see equations (9) and (10)). We also validated the closed-form analytical formulas for IHMDP through the simulation results. In this section, we present representative results to highlight the major findings.

Table 11 shows the decision made by models with the increasing value of f. The IHMDP model makes the "stay" decision when f is larger than 60. For the FHMDP model, it can be seen that the probability of moving the application decreases as f increases (i.e., tend to "stay" when f increases).

Figure 30 shows the expected total cost when f increases. For the IHMDP model, it can minimize the expected total cost by moving and not moving the application when f is below and above the threshold value of 60, respectively. By doing so, the expected total cost increases linearly from a low value when f is small. When f is above 60, the expected

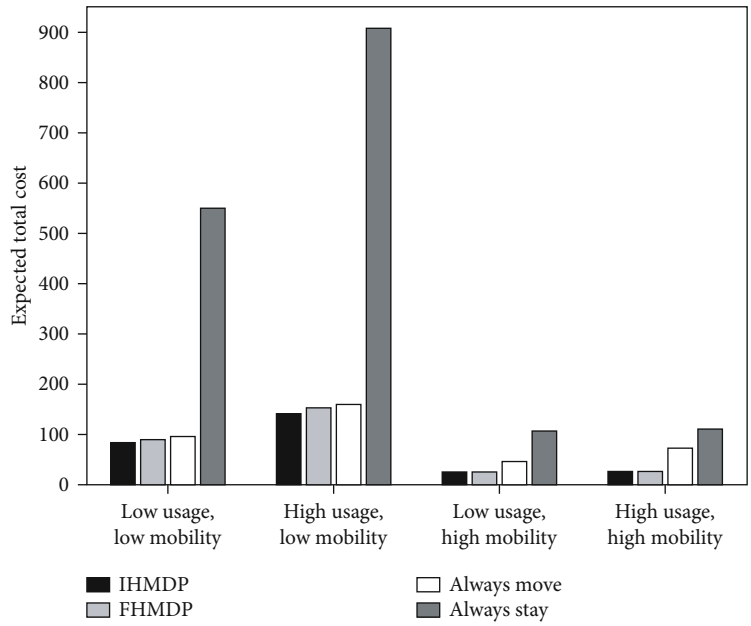

Figure 35: Expected total cost when most edge clouds are of type CT1.

total cost becomes constant because there is no need to move the application (i.e., the optimal policy is "stay"). For the FHMDP model, the expected total cost is in general consistent with that of the IHMDP model, except at $f = 70$ and 80. The larger discrepancy is due to the fact that there is still a moderate chance of moving the application. When f reaches 90, the moving probability becomes very small. Hence, the expected total cost matches with that of the IHMDP model.

That means the lowest expected total cost can always be achieved in the IHMDP model. When f is above 60, the expected total cost becomes constant because there is no need to move the application (i.e., the optimal policy is "stay").

For the FHMDP model, the expected total cost is in general consistent with that of the IHMDP model, except at $f = 70$ and 80. The larger discrepancy is due to the fact that there is still a moderate chance of moving the application. When f reaches 90, the moving probability becomes very small. Hence, the expected total cost matches that of the IHMDP model.

Table 12 shows the influence of C on the decision policy. For the IHMDP model, it can be seen that when C is low, the decision should be "move." When C increases to a certain value, the decision should be "stay." For the FHMDP model, the probability of moving the application decreases as C increases (i.e., tend to "move" and "stay" for small and large C, respectively).

For the IHMDP model, Figure 31 shows that the expected total cost increases linearly between $C = 100$ and $C = 300$. Beyond $C = 300$, the expected total cost becomes constant because there is no need to move the application. Similar to the above, it can be seen that the IHMDP model can minimize the expected total cost by moving and not moving the application when C is below and above the threshold value of 300, respectively. For the FHMDP model, the expected total cost is higher, especially when C is equal to or larger than 300, because it still has a high chance of moving the application (i.e., due to its probabilistic nature).

Table 13 shows the influence of b on the decision. For the IHMDP model, when b is low, the decision should be "stay." When b is increased to a threshold value (i.e., $b = 0.7$), the decision is changed to "move." For the FHMDP model, it can be seen that when b is small, the probability of moving the application is almost zero, indicating a strong "stay" decision, which is consistent with the other two models.

Figure 32 shows that the expected total cost increases more dramatically as b is close to one. In general, the expected total costs for all methods are consistent, although the expected total cost for the FHMDP model is slightly higher because of the probabilistic nature of its decision policy (e.g., it has half a chance of moving and staying at $b = 0.8$). For the IHMDP model, again it can minimize the expected total cost by not moving and moving the application when b is below and above the threshold value of 0.7, respectively.

Table 14 shows the influence of p on the decision. For the IHMDP model, when p is low, the decision should be "stay." As p increases to 0.4, the decision becomes "move." For the FHMDP model, the probability of moving the application increases as p increases.

Figure 33 shows that in general, the expected total costs for all methods are consistent, except that in some cases, the costs for the FHMDP model are higher due to the probabilistic nature of its decision policy. Similar to the above, the IHMDP model can minimize the expected total cost by not moving and moving the application when p is below and above the threshold value of about 0.35.

Last but not least, in all of the above simulations, the expected total costs of the IHMDP, as determined by the analytical model, agree closely with the simulation results, thus validating the analytical model.

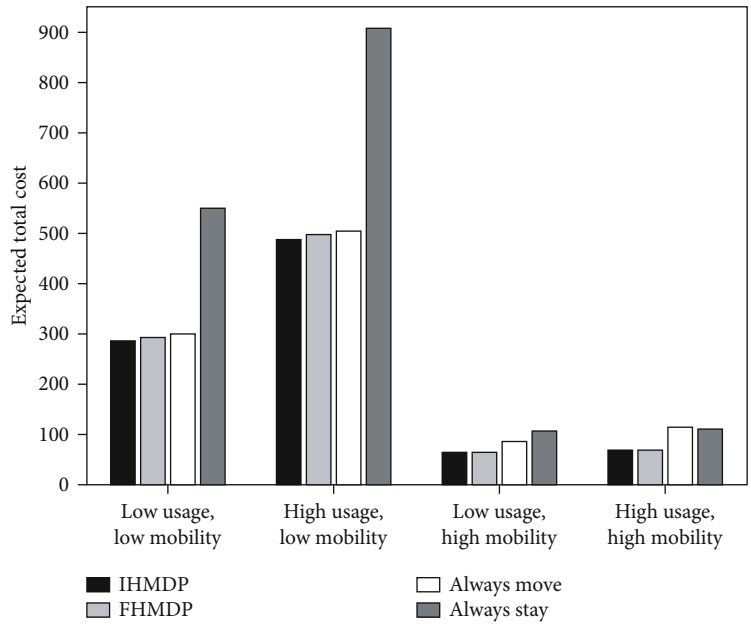

Figure 36: Expected total cost when most edge clouds are of type CT2.

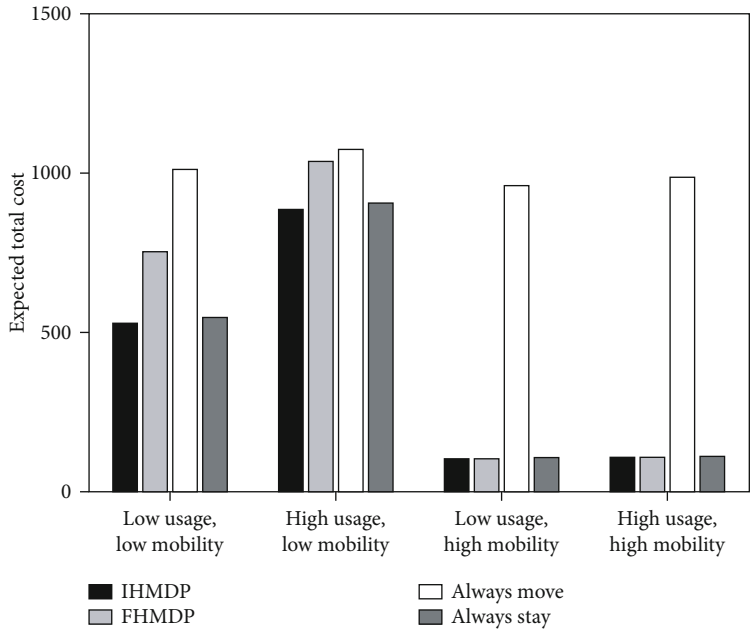

Figure 37: Expected total cost when most edge clouds are of type CT3.

5.3.2. Multiple Edge Clouds and Multiple Users. In the last section, we mainly discussed the influence of different parameters on the expected total cost of an individual user. In this section, we conduct simulations for a multiuser and multicloud environment as shown in Figure 34. For simulation purposes, we assume that all users belong to the same central cloud (i.e., with the same cost of running an application in the central cloud). Whenever a user visits a new area/region/location, he/she can decide whether to move the application to the corresponding edge cloud or use the application remotely from the central cloud based on the aforementioned models. After leaving the current edge cloud, the user may join another edge cloud. The decision process will then repeat.

The simulation model and simulation parameters are shown in Table 15. We assume there are 10,000 users and 40 edge clouds. There are four types of users (i.e., with different duration probabilities b and usage probabilities p). For example, UT1 users have moderate usage ($p = 0.5$) and low mobility ($b = 0.9$). Note that we did not consider low usage, because the cost will be too low to provide a meaningful comparison. The edge clouds can be categorized into four types

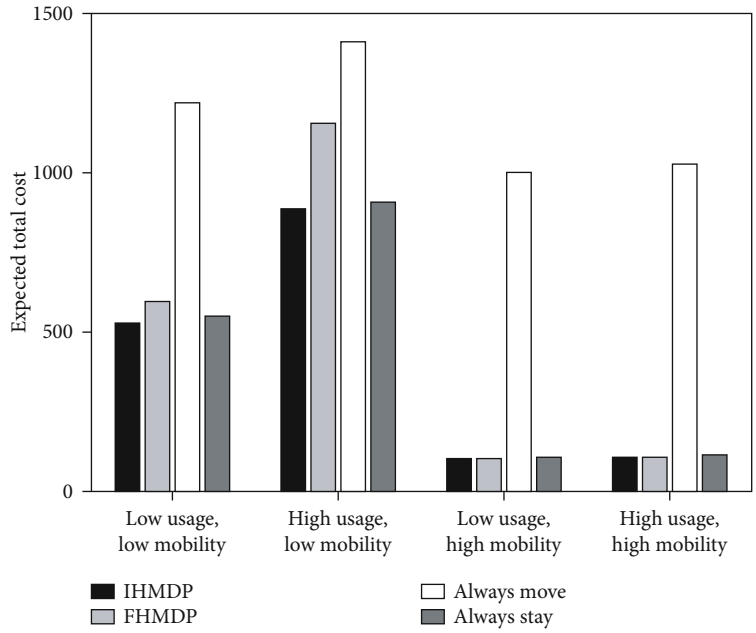

Figure 38: Expected total cost when most edge clouds are of type CT4.

Table 16: Summary of decision policies.

Decision model	Decision policy
FHMDP model (deterministic) (i.e., move the application if the duration τ is greater than a certain value)	$\tau > (1/p)\left(C'/\left(1-f'\right)-1\right)+1$
FHMDP model (probabilistic) (i.e., move the application with probability ρ)	$\rho = (1/p)\left(\left(C'/\left(1-f'\right)\right)-1\right)+1$
IHMDP model (i.e., move the application if a condition is fulfilled)	$f' < 1 - C'/(1+(bp/(1-b)))$ $C' < \left(1-f'\right)/(1-bp/(1-b))$ $p > \left(\left(C'-1\right)/\left(1-f'\right)\right)(1/b-1)$ $b > 1/\left(1+\left(p\left(1-f'\right)/\left(C'-\left(1-f'\right)\right)\right)\right)$

depending on the transfer cost C and usage cost f at the edge cloud. For example, CT1 clouds have low usage cost and low transfer cost (e.g., closer to the central cloud). To facilitate the simulations, we assume that the number of clouds and the number of users are uniformly distributed (i.e., each with 25% and a user may visit a certain type of cloud with the same probability). In the following simulations, we evaluate the performance of different combinations of cloud types. In each case, the major cloud type will account for 94% of the clouds and the remaining ones will account for 6% (i.e., 2% for each type). Figures 35–38 show the evaluation results when the majority of edge clouds are of types CT1, CT2, CT3, and CT4, respectively. In each figure, the expected total cost of each type of user is shown. Note that similar to the previous simulation results, the expected total cost is found based on the mean cost of all of the visited clouds during the simulation period (i.e., the expected total cost is cloud-based). For each type of user, the expected total costs for the IHMDP model and FHMDP model are compared with the results for "ALWAYS STAY" and "ALWAYS MOVE."

Figure 35 shows the expected total cost when the majority of the edge clouds are of type CT1 (i.e., low usage cost and low transfer cost). The results show that the expected total costs for the IHMDP model, FHMDP model, and "ALWAYS MOVE" can maintain at a low level, while the expected total costs for "ALWAYS STAY" are much higher, especially for low-mobility users. This is because as the transfer cost is low, it is more desirable to transfer the application to achieve better performance (i.e., a lower cost). In the cases of the IHMDP model and FHMDP model, the "move" decision can be determined effectively by the algorithms. Figure 36 shows the expected total cost when the majority of edge clouds are of type CT2 (i.e., moderate usage cost and low transfer cost). In general, the trend is similar to that of Figure 35 although the costs are higher because of the higher usage cost. Again, the IHMDP model and FHMDP model can perform well to maintain a lower expected total cost for all types of users. Figure 37 shows the expected total cost when the majority of the edge clouds are of type CT3 (i.e., low usage cost and high transfer cost). It can be seen that

Input: *state*, *action*, C, f, h and p, b
Output: *next_state*, *cost*; // decide next state and compute immediate cost
1 Initialize: is_*move* ⟵ False;
//application has not yet moved
2 if is_{move} = False and *action* = "*move*" **then**
3 *cost* = $C + f$, is_*move* = True
4 if $random.uniform(0, 1) > b$ **then**
5 $next\_state = s_{end}$
6 **else**
7 $next\_state = s_{f,u}$
8 **end**
9 end
10 if is_*move* = *False* and *action* = "*stay*" **then**
11 *cost* = h, is_{move} = True
12 **if** random.uniform(0, 1) > b **then**
13 $next\_state = s_{end}$
14 **else**
15 $next\_state = s_{h,u}$
16 **end**
17 end
/∗application has already moved∗/
18 if is_*move* = *True* **and** *action* = "*stay*" **then**
19 **if** $random.uniform(0, 1) \leq p$ **then**
20 **if** $state = s_{h,u}$ or $state = s_{h,i}$ **then**
21 $cost = h$, $next\_state = s_{h,u}$
22 **end**
23 **if** $state = s_{f,u}$ or $state = s_{f,i}$ **then**
24 $cost = f$, $next\_state = s_{f,u}$
25 **end**
26 **if** $random.uniform(0, 1) \geq b$ **then**
27 $next\_state = s_{end}$
28 **end**
29 **end**
//compute the cost when the application is used
30 **if** random.uniform(0,1)>p then
31 **if** $state = s_{h,u}$ or $state = s_{h,i}$ then
32 $cost = 0$, $next\_state = s_{h,i}$
33 **end**
34 **if** $state = s_{f,u}$ or $state = s_{f,i}$ then
35 $cost = 0$, $next\_state = s_{f,i}$
36 **end**
37 **if** $random.uniform(0, 1) \geq b$ **then**
38 $next\_state = s_{end}$
39 **end**
40 **end**
//computer the cost when the application is idle
41 end

Algorithm 3: **STATE_TRANSITION**() for the system behaviour.

the result includes a number of differences compared to the previous ones. In this case, "ALWAYS MOVE" now has a higher cost than "ALWAYS STAY" as the transfer cost is higher, especially for high-mobility users. Furthermore, the IHMDP model can outperform the FHMDP model, especially for low-mobility users. Compared with FHMDP, IHMDP can save up to 42% extra cost. The IHMDP model cost is at most 5.6 times lower compared to the "ALWAYS STAY" decision and at most 8.7 times lower compared to the "ALWAYS MOVE" decision. Figure 38 shows a similar trend (i.e., similar to Figure 37) with higher expected total cost in general due to the higher usage cost. Again, the IHMDP model can outperform the FHMDP model and "ALWAYS MOVE" has the worst performance because of the high transfer cost. In summary, the simulation results show that the IHMDP model and FHMDP model work well in the multicloud and multiuser environment and the IHMDP model can always provide the best performance.

As mentioned, we have conducted extensive simulations to analyze the performance of the algorithms. The above results are representative examples to highlight the system behavior/performance and major findings. Other results show a similar trend. In summary, based on these extensive experiments and simulations, the major findings are given as follows:

(i) The basic concept of the mobile Intercloud system for edge cloud computing has been demonstrated by an experimental prototype. Basically, objects and applications can be transferred through Intercloud gateways based on ICCP in accordance with predefined requirements (e.g., algorithms and programs). For the interactions between a physical mobile terminal and a virtual mobile terminal, performance is mostly affected by the number of interactions and exchanged data size. Hence, it is important to minimize the number of interactions and exchanged data size by duplicating certain data and predefining certain frequently used tasks

(ii) The IHMDP model can always provide the best decision policy and achieve the lowest expected cost. Furthermore, as confirmed by the simulation results, the closed-form solutions should be correct. That means if the system parameters can be estimated based on the analytical model, the formulas are useful for determining the basic decision policy. In particular, it is deterministic and traceable

(iii) The FHMDP model with a probabilistic policy can provide similar results to the IHMDP model in general. Again, as confirmed by the simulation results, the analytical solution should be correct, providing a useful basis for formulating the decision policy. However, because of its probabilistic nature, decisions may sometimes be made incorrectly (i.e., not optimally)

(iv) As illustrated by the multiuser and multicloud simulations, the decision policy (i.e., move or stay) depends on the user type (i.e., p and b) and cloud type (i.e., C, f, and h). For example, when there are many edge clouds with low usage cost and low transfer cost, it is better to move an application, especially for low-mobility users. On the other hand, if there are many edge clouds with high transfer cost, it is better not to move the application (i.e., use the "stay" policy), especially for high-mobility users. In general, the optimal decision policy can be determined effectively by the IHMDP model.

Table 16 summarizes the decision policies for the Markov decision models with the costs normalized with respect to h (i.e., assuming $h = 1$). That means $C' = C/h$ and $f' = f/h$. Note that the equations are rearranged in more meaningful forms to provide better insights. For example, for the FHMDP model, if $C' = 10$, $f' = 0.5$, and $p = 0.5$, the application should be moved if the duration for remaining in the edge cloud is greater than 39 time units. Using the same parameters, if $b = 0.5$, we should move the application with a very small probability (i.e., for the probabilistic decision policy for the FHMDP model). For the IHMDP model, if we assume $C' = 1.5$, $p = 0.5$, and $b = 0.5$, we should move the application if f' is less than one (i.e., "ALWAYS MOVE"). Other interesting relationships can be derived in similar manners for edge cloud computing.

6. Conclusion

In conclusion, we have presented a mobile Intercloud system for supporting edge cloud computing. In addition to task or computation offloading, virtual mobile terminals as well as data/files and applications can be transferred among clouds (e.g., from a central cloud to an edge cloud) to better support physical mobile terminals. The main advantages of the mobile Intercloud system are explained as follows. First, the central cloud and edge cloud can interact directly with each other under the Intercloud framework. For example, objects (e.g., data and application) can be moved directly from one cloud to another cloud as controlled by a mobile terminal. Second, based on different decision policies and user requirements, a user can decide how and when to transfer the objects effectively, efficiently, and flexibly. Third, due to the direct transfer between clouds, the Intercloud operations can be transparent to end users (e.g., a cloud service provider can obtain certain objects for a mobile user based on predefined policies, bringing convenience to the mobile users). To support application transfer over the mobile Intercloud system for edge cloud computing, both FHMDP and IHMDP models have been formulated to determine the decision policies. Based on the models, closed-form solutions have been obtained for the decision policy. Extensive experimental, analytical, and simulation evaluations have been conducted to analyze the system, performance of the models, and decision policies. The evaluation results provide valuable insights into the design of the mobile Intercloud system for edge cloud computing. In particular, a summary table of the decision policies (i.e., Table 16) has been developed to show meaningful closed-form expressions for edge cloud computing. These expressions should also be useful for similar scenarios (e.g., for mobile cloud computing in general). For future work, the decision model can be enhanced (e.g., by using more sophisticated machine learning mechanisms), implementation issues can be studied in detail, and various edge cloud applications can be developed for demonstration and evaluation purposes. The framework and models presented in this paper should provide a foundation for the future research work on the mobile Intercloud system for edge cloud computing.

Data Availability

Certain data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by The Hong Kong Polytechnic University under account YBAJ.

References

[1] R. Buyya, J. Broberg, and A. M. Goscinski, *Cloud Computing: Principles and Paradigms*, vol. 87, John Wiley & Sons, 2010.

[2] "NIST cloud computing program - NCCP," February 2021, https://www.nist.gov/programs-projects/nist-cloud-computing-program-nccp.

[3] "Cloud computing - statistics & facts," February 2021, https://www.statista.com/topics/1695/cloud-computing/.

[4] "Distribution of global cloud and non-cloud mobile data traffic from 2014 to 2019," February 2021, https://www.statista.com/statistics/292840/distribution-global-cloud-and-non-cloud-traffic/.

[5] H. T. Dinh, C. Lee, D. Niyato, and P. Wang, "A survey of mobile cloud computing: architecture, applications, and approaches," *Wireless Communications and Mobile Computing*, vol. 13, no. 18, p. 1611, 2013.

[6] D. Huang, T. Xing, and H. Wu, "Mobile cloud computing service models: a user-centric approach," *IEEE Network*, vol. 27, no. 5, pp. 6–11, 2013.

[7] A. ur Rehman Khan, M. Othman, S. A. Madani, and S. U. Khan, "A survey of mobile cloud computing application models," *IEEE Communications Surveys and Tutorials*, vol. 16, no. 1, pp. 393–413, 2014.

[8] B. Zhou and R. Buyya, "Augmentation techniques for mobile cloud computing: a taxonomy, survey, and future directions," *ACM Computing Surveys*, vol. 51, no. 1, pp. 13:1–13:38, 2018.

[9] A. N. Toosi, R. N. Calheiros, and R. Buyya, "Interconnected cloud computing environments: challenges, taxonomy, and survey," *ACM Computing Surveys*, vol. 47, no. 1, pp. 7:1–7:47, 2014.

[10] N. Grozev and R. Buyya, "Inter-cloud architectures and application brokering: taxonomy and survey," *Software: Practice and Experience*, vol. 44, no. 3, pp. 369–390, 2014.

[11] A. J. Ferrer, J. M. Marquès, and J. Jorba, "Towards the decentralised cloud: survey on approaches and challenges for mobile, ad hoc, and edge computing," *ACM Computing Surveys*, vol. 51, no. 6, pp. 111: 1–111: 36, 2019.

[12] B. Zhou, A. V. Dastjerdi, R. N. Calheiros, and R. Buyya, "An online algorithm for task offloading in hetero-geneous mobile clouds," *ACM Transactions on Internet Technology*, vol. 18, no. 2, pp. 23:1–23: 25, 2018.

[13] M. B. Terefe, H. Lee, N. Heo, G. C. Fox, and S. Oh, "Energy-efficient multisite offloading policy using Markov decision process for mobile cloud computing," *Pervasive and Mobile Computing*, vol. 27, pp. 75–89, 2016.

[14] Y. Kim, H. Lee, and S. Chong, "Mobile computation offloading for application throughput fairness and energy efficiency," *IEEE Transactions on Wireless Communications*, vol. 18, no. 1, pp. 3–19, 2019.

[15] S. Deng, L. Huang, J. Taheri, and A. Y. Zomaya, "Computation offloading for service workflow in mobile cloud computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 12, pp. 3317–3329, 2015.

[16] H. Flores, P. Hui, P. Nurmi et al., "Evidence-aware mobile computational offloading," *IEEE Transactions on Mobile Computing*, vol. 17, no. 8, pp. 1834–1850, 2018.

[17] V. Haghighi and N. S. Moayedian, "An offloading strategy in mobile cloud computing considering energy and delay constraints," *IEEE Access*, vol. 6, pp. 11849–11861, 2018.

[18] S. Guo, J. Liu, Y. Yang, B. Xiao, and Z. Li, "Energy-efficient dynamic computation offloading and cooperative task scheduling in mobile cloud computing," *IEEE Transactions on Mobile Computing*, vol. 18, no. 2, pp. 319–333, 2019.

[19] K. Elgazzar, P. Martin, and H. S. Hassanein, "Cloud-assisted computation offloading to support mobile services," *IEEE Transactions on Cloud Computing*, vol. 4, no. 3, pp. 279–292, 2016.

[20] A. Ksentini, T. Taleb, and M. Chen, "A Markov decision process-based service migration procedure for follow me cloud," in *2014 IEEE International Conference on Communications (ICC)*, pp. 1350–1354, Sydney, Australia, 2014.

[21] S. Wang, R. Urgaonkar, T. He, M. Zafer, K. S. Chan, and K. K. Leung, "Mobility-induced service migration in mobile micro-clouds," in *2014 IEEE Military Communications Conference*, pp. 835–840, Baltimore, MD, USA, 2014.

[22] W. Zhang, J. Chen, Y. Zhang, and D. Raychaudhuri, "Towards efficient edge cloud augmentation for virtual reality MMOGs," in *Proceedings of the Second ACM/IEEE Symposium on Edge Computing, San Jose/Silicon Valley, SEC 2017*, pp. 8:1–8: 14, CA, USA, 2017.

[23] S. Chen, Y. Wang, and M. Pedram, "A semi-Markovian decision process based control method for offloading tasks from mobile devices to the cloud," in *2013 IEEE Global Communications Conference (GLOBECOM)*, pp. 2885–2890, Atlanta, GA, USA, 2013.

[24] Y. Zhang, D. Niyato, and P. Wang, "Offloading in mobile cloudlet systems with intermittent connectivity," *IEEE Transactions on Mobile Computing*, vol. 14, no. 12, pp. 2516–2529, 2015.

[25] L. Hou, "A continuous-time Markov decision process-based resource allocation scheme in vehicular cloud for mobile video services," *Computer Communications*, vol. 118, pp. 140–147, 2018.

[26] F. Jazayeri, A. Shahidinejad, and M. Ghobaei-Arani, "A latency-aware and energy-efficient computation offloading in mobile fog computing: a hidden Markov model-based approach," *The Journal of Supercomputing*, vol. 77, no. 5, pp. 4887–4916, 2021.

[27] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Transactions on Networking*, vol. 24, no. 5, pp. 2795–2808, 2016.

[28] W. Chen, D. Wang, and K. Li, "Multi-user multi-task computation offloading in green mobile edge cloud computing," *IEEE Transactions on Services Computing*, vol. 12, no. 5, pp. 726–738, 2019.

[29] M. Khayyat, I. A. Elgendy, A. Muthanna, A. S. Alshahrani, S. Alharbi, and A. Koucheryavy, "Advanced deep learning-based computational offloading for multilevel vehicular edge-cloud computing networks," *IEEE Access*, vol. 8, pp. 137052–137062, 2020.

[30] Y. Miao, G. Wu, M. Li, A. Ghoneim, M. al-Rakhami, and M. S. Hossain, "Intelligent task prediction and computation offloading based on mobile-edge cloud computing," *Future Generation Computer Systems*, vol. 102, pp. 925–931, 2020.

[31] Z. Hong, W. Chen, H. Huang, S. Guo, and Z. Zheng, "Multi-hop cooperative computation offloading for industrial IoT-edge-cloud computing environments," *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 12, pp. 2759–2774, 2019.

[32] Z. Liao, J. Peng, J. Huang et al., "Distributed probabilistic offloading in edge computing for 6G-enabled massive Internet of things," *IEEE Internet of Things Journal*, vol. 8, no. 7, pp. 5298–5308, 2021.

[33] R. Fantacci and B. Picano, "Performance analysis of a delay constrained data offloading scheme in an integrated cloud-fog-edge computing system," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 10, pp. 12004–12014, 2020.

[34] Y. Shi, J. Sun, D. Liu et al., "Cloud-based data offloading for multi-focus and multi-views image fusion in mobile applications," *Mobile Networks and Applications*, vol. 26, no. 2, pp. 830–841, 2021.

[35] Z. Chang, Z. Zhou, T. Ristaniemi, and Z. Niu, "Energy efficient optimization for computation offloading in fog computing system," in *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*, pp. 1–6, Singapore, 2017.

[36] Z. Zhou, P. Liu, Z. Chang, C. Xu, and Y. Zhang, "Energy-efficient workload offloading and power control in vehicular edge computing," in *2018 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, pp. 191–196, Barcelona, Spain, April 2018.

[37] P. Mach and Z. Becvar, "Mobile edge computing: a survey on architecture and computation offloading," *IEEECommunications Surveys & Tutorials*, vol. 19, no. 3, pp. 1628–1656, 2017.

[38] A. Shakarami, M. Ghobaei-Arani, M. Masdari, and M. Hosseinzadeh, "A survey on the computation offloading approaches in mobile edge/cloud computing environment: a stochastic-based perspective," *Journal of Grid Computing*, vol. 18, no. 4, pp. 639–671, 2020.

[39] D. Bernstein and D. Vij, "IEEE PROJECT 2302-Standard for Intercloud Interoperability and Federation (SIIF)," 2012, https://standards.ieee.org/develop/project/2302.html.

[40] Y. Demchenko, M. X. Makkes, R. J. Strijkers, and C. de Laat, "Intercloud architecture for interoperability and integration," in *4th IEEE International Conference on Cloud Computing Technology and Science Proceedings*, pp. 666–674, Taipei, Taiwan, December 2012.

[41] C. Shan, C. Heng, and Z. Xianjun, "Inter-cloud operations via NGSON," *IEEE Communications Magazine*, vol. 50, no. 1, pp. 82–89, 2012.

[42] C. T. Yu, H. C. B. Chan, and D. W. K. Kwong, "Discovering resources in an intercloud environment," in *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*, pp. 1–6, Singapore, December 2017.

[43] Y. H. Ho, P. M. F. Ho, and H. C. B. Chan, "Mobile intercloud system and objects transfer mechanism," in *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*, pp. 1–6, Singapore, December 2017.

[44] Y. H. Ho, Z. Cheng, P. M. F. Ho, and H. C. Chan, "Mobile intercloud system with blockchain," in *Proceedings of the international MultiConference of engineers and computer scientists*, vol. 1, p. 2018.

[45] C. E. Perkins, "Mobile IP," *IEEE Communications Magazine*, vol. 35, no. 5, pp. 84–99, 1997.

[46] 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.

[47] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*, John Wiley & Sons, 2014.