Wireless Communications and Mobile Computing

Service Migration in Mobile Edge Computing

Lead Guest Editor: Shangguang Wang
Guest Editors: Wu Chou, Kok-Seng Wong, Ao Zhou, and Victor C. Leung

Service Migration in Mobile Edge Computing

Wireless Communications and Mobile Computing

Service Migration in Mobile Edge Computing

Lead Guest Editor: Shangguang Wang
Guest Editors: Wu Chou, Kok-Seng Wong, Ao Zhou, and Victor C. Leung

Editorial Board

Contents

Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 3823721, 2 pages
https://doi.org/10.1155/2018/3823721

Editorial

Service Migration in Mobile Edge Computing

Shangguang Wang,[1] **Wu Chou**,[2] **Kok-Seng Wong**,[3] **Ao Zhou**,[1] **and Victor C. Leung**[4]

[1]*State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, No. 10, Xitucheng Rd., Beijing, China*
[2]*Huawei Shannon IT Lab, Bridgewater, NJ, USA*
[3]*Soongsil University, Seoul, Republic of Korea*
[4]*The University of British Columbia, BC, Canada*

Correspondence should be addressed to Shangguang Wang; sgwang@bupt.edu.cn

Received 4 November 2018; Accepted 4 November 2018; Published 13 November 2018

With the rapid proliferation of intelligent mobile devices, mobile edge computing (MEC) is proposed to relieve the bottleneck of mobile devices' limited capacity and low latency for services from the core network. Service migration is a significant issue in MEC. The contradiction between the limited covering area of a single edge server and the mobility of users (e.g., intelligent vehicles and smart devices) will result in significant network degradation if the distance between the user and his serving edge server is beyond a threshold, which can further lead to dramatic drop of QoS/QoE and even interruption of ongoing services. At this moment, service migration bears the responsibility to migrate the user's ongoing service from the source (current) edge server to the destination edge server near the mobile user.

Although there have been some research efforts relevant to service migration in MEC, many challenges should be addressed. On one hand, the characteristics of service migration for MEC are quite different from the traditional handover process in cellular network and live migration in data center of cloud computing diagram for emphasis on data transferring, different evaluation metrics, more complex network conditions, and so forth. On the other hand, the service migration for MEC should deal with the nontrivial problems in MEC, such as mobility management, cache-enabled, deployment of MEC systems, and resource management. Moreover, energy-saving service migration in real time, architecture design of MEC systems, and the privacy and security mechanisms for service migration are also considered.

The main focus of this special issue is on the proposal of techniques for service migration for MEC and wireless mobile network. This special issue summarizes the most recent developments in the field, which aims to foster the dissemination of high quality research in new ideas, methods, theories, techniques, and applications of evaluation and management for improving mobile services, especially, in areas of

(i) QoS-aware edge server selection algorithm in MEC systems

(ii) Selection algorithm of migration path with both of latency and cost in MEC

(iii) Mobility management for service migration in MEC

(iv) Virtual resource allocation strategy on edge servers

(v) Architecture design of MEC systems for efficient service migration

(vi) Data compression algorithm and transferring optimization in service migration

(vii) Network control protocols and algorithms for service migration in MEC

(viii) Security and privacy issues for service migration in MEC

Twenty-five papers were submitted for this special issue. Our distinguished reviewers from respective research fields narrowed the field to fourteen papers which were finally

accepted. The following is a short summary of the findings of each of these papers.

L. Rao et al. designed an authenticated data structure called bv23Tree for Remote Storage Security, which enables client to batch-verify the indices and values of any number of appointed leaves all at once for efficiency. By utilizing bv23Tree and a hierarchical storage structure, they presented the solution to support dynamic updates of the outsourced data.

Q. Wei et al. proposed a flexible, secure, and reliable data sharing scheme based on collaboration in multicloud environment. For securely and instantly providing data sharing service even if the owner is offline and without trusted third party, they distributed all encrypted split data/key blocks together to multiple cloud service providers, respectively.

B. Cao et al. proposed a flexible dynamic pricing model for cloud service which takes into account occupying time, resource consumption, and maximal concurrency. In the pricing model, the fee of cloud service for the user is mainly composed of three parts: the monthly rental, the fee of his maximal concurrency, and the fee of his using time and resource consumption.

S. Yang et al. proposed a flexible framework which could support three kinds of selection schemes with respect of different service requirements for mobile crowdsensing service. The framework consists of two modules. For collaboration motivating module, they introduced two motivating methods including centralized decision-making and distributed decision-making. For message relaying module, they introduced two methods: contention based relaying and clustered based relaying.

Q. Wang et al. proposed an efficient forwarding capability evaluation method for opportunistic offloading in mobile edge computing. The first step of the method is to judge the possibility of a device contacting the destination within the time constraint of the data using the proposed transient cluster detection method. The second step of the method is to calculate the device's probability of encountering destination within the time constraint of the data as evaluation metric.

D. Yan et al. took the restaurant recommendation as an example and proposed a personalized POI recommender system integrating the user profile, restaurant characteristics, users' historical behavior features, and subway network features. Then the subway network features such as the number of passing stations, waiting time, and transfer times are extracted and they employed a recurrent neural network model to model user behaviors.

H. Wu et al. proposed a cross-edge model for better personalization service, and they revealed how famous topics in one resource edge server can emerge on several other destination edge servers in mobile edge environment. The prototype of the model consists of the interaction between the users and diverse items, the personalization of new items, and privacy collection and distribution strategies.

W. Liang et al. proposed a weighted hidden Markov model to predict the security situation of the mobile network. In the model, the multiscale entropy is used to address the low speed of data training in mobile network, and the autocorrelation coefficient can reasonably use the association between the characteristics of the historical data to predict future security situation.

X. Ma et al. proposed an image restoration method in edge computing environment which achieves good image real-time restoration results. For this method, the 10 classical functions are used to determine the population size and maximum iteration times of traction fruit fly optimization algorithm, and then the algorithm is used to optimize the optimal parameters of least squares support vector regression kernel function, and the error function of image restoration is taken as an adaptive function of the algorithm.

W. Ding and Z. Zhao proposed a data harmonization service for spatiotemporal data stream in the edge computing environment. Through the online cleaning and complementing steps of the hierarchical service instances, the records' validity and continuity can be guaranteed in an efficient way.

K. Peng et al. proposed a comprehensive survey of mobile edge computing (MEC) from the perspective of service adoption and provision. They firstly described the overview of MEC and reviewed the existing MUs-oriented service adoption of MEC. Finally, open issues are investigated.

K. Niu et al. proposed a flexible algorithm for bed-time prediction, which is designed for predicting wake time and bedtime by analyzing screen status of smartphone. The first one is to detect get up and go to bed incidents. The second one is to predict the next wake time and bedtime.

Y. Chen et al. proposed a dynamic service request scheduling algorithm, which makes request scheduling decisions to optimize scheduling cost while providing performance guarantees. The algorithm can be implemented in an online and distributed way and can achieve arbitrary tradeoff between scheduling cost and performance.

J. Huang et al. proposed a simulation-based approach of QoS-aware dynamic service selection for mobile edge computing systems. Stochastic system models were presented and mathematical analyses were provided. Based on the analytical results, the QoS-aware service selection problem was formulated by a dynamic optimization problem.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this article.

Shangguang Wang
Wu Chou
Kok-Seng Wong
Ao Zhou
Victor C. Leung

Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 5485461, 10 pages
https://doi.org/10.1155/2018/5485461

Research Article

A Simulation-Based Approach of QoS-Aware Service Selection in Mobile Edge Computing

Jiwei Huang[1,2] **Yihan Lan**[3] **and Minfeng Xu**[3]

[1]*Department of Computer Science and Technology, China University of Petroleum - Beijing, Beijing 102249, China*
[2]*Beijing Key Laboratory of Petroleum Data Mining, Beijing 102249, China*
[3]*International School, Beijing University of Posts and Telecommunications, Beijing 100876, China*

Correspondence should be addressed to Jiwei Huang; huangjw05@gmail.com

Received 20 April 2018; Accepted 19 July 2018; Published 1 November 2018

Academic Editor: Kok-Seng Wong

Edge computing is an emerging computational model that enables efficient offloading of service requests to edge servers. By leveraging the well-developed technologies of cloud computing, the computing capabilities of mobile devices can be significantly enhanced in edge computing paradigm. However, upon the arrival of user requests, whether to dispatch them to the edge servers or cloud servers in order to guarantee the quality of service (QoS), i.e., the QoS-aware service selection problem, still remains an open problem. Due to the dynamic mobility of users and the variation of task arrivals and service processes, it is extremely costly to obtain the global optimal solution by both mathematical approaches and simulation-based schemes. To attack this challenge, this paper proposes a simulation-based approach of QoS-aware dynamic service selection for mobile edge computing systems. Stochastic system models are presented and mathematical analyses are provided. Based on the analytical results, the QoS-aware service selection problem is formulated by a dynamic optimization problem. Goal softening is applied to the original problem, and service selection algorithms are designed using ordinal optimization techniques. Simulation experiments are conducted to validate the efficacy of the approach presented in this paper.

1. Introduction

Edge computing is an emerging technique of optimizing cloud computing systems by performing data processing at the edge of the network near the source of the original data [1]. It pushes applications, data, and services away from centralized points (i.e., the cloud) to the logical extremes of a network. Consequently, the communications bandwidth needed between sensors and the central data center by performing analytics and knowledge generation can be significantly reduced. With the rapid development of mobile communications and mobile services, mobile edge computing (MEC) as a special type of edge computing has emerged. It is a network architecture that enables cloud computing capabilities and an IT service environment at the edge of the cellular network [2]. Due to its high performance and strong support for new personalized services for specific customers, MEC has become a hot topic in both industry and academia.

In an MEC system, cellular operators can efficiently deploy their services on the edge nodes which are usually cellular base stations. Since the cellular base stations combine elements of information technology and telecommunications networking in the same equipment, their performance has become a critical issue especially for some special applications such as Internet of Things (IoT). Comparing to the cloud data center which can be commonly regarded as a high-performance computing (HPC) system, the capacity of the base station usually meets some bottlenecks in reality. When the workload of a cellular base station gets heavy, the user requests have to be transferred to the cloud site in order to guarantee the quality of service (QoS) for both the services requested by users and the services already running on the edge node. Therefore, how to determine whether edge servers or cloud servers handle the user requests upon their arrival in order to meet the QoS requirement, i.e., the QoS-aware service selection problem, is critical for the performance of MEC systems.

Due to the high dynamic characteristics of both workload arrival and MEC systems, service selection can be theoretically formulated as a dynamic optimization problem. There have been several mathematical approaches for solving dynamic optimization problems. Nevertheless, most of them meet challenges in large-scale MEC systems. On one hand, dynamic programming based approaches such as Markov decision process (MDP) face a state explosion issue when the scale of MEC system grows reasonably large, resulting in the impossibility of solving the problems in acceptable time. On the other hand, queueing-based approaches, like Lyapunov optimization which is very popular in dynamic optimization nowadays, can only give a lower bound of the optimization algorithms. Furthermore, some of the approaches have to make assumptions to facilitate their optimization procedures, needing additional prior knowledge on the mathematical distributions of the dynamic processes, or resulting in an inaccuracy of performance evaluation. To attack these challenges, another type of approaches has emerged, whose basic idea is to design simulation models and conduct simulation experiments to obtain the optimal service selection policy [3]. However for the QoS-aware service selection problems, the performance of the simulations is a critical issue. Using the simulation to evaluate the output variables for a given setting of the input variables is already computationally expensive not even mention the search of the best policy provided that the input-variable space is huge, and furthermore variability is an integral part of the problem making the simulations more complex [4]. Thus, the simulation approaches sometimes face space explosion problem for both state space and action space. Therefore, how to design and implement efficient approaches for solving the QoS-aware service selection problem remains an open problem.

In this work, we make an attempt at filling this gap and propose an efficient scheme of simulation-based QoS-aware service selection for MEC systems. Simulation models capturing the dynamics and characteristics of MEC systems are carefully designed, and their corresponding mathematical analyses are provided. A framework of event-driven simulation is given, and algorithms of simulation-based service selection are proposed. For facilitating the long-term dynamic optimization, we slightly sacrifice some part of the optimality by softening our goal from "finding the best" to "selecting good enough solutions." Mathematical formulations are presented and quantitative analyses of both performance bound and convergence rate are conducted. By applying ordinal optimization (OO) techniques, we propose an efficient scheme of simulation-based optimization for QoS-aware service selection in MEC systems. Finally, we conduct empirical experiment based on real-life data to validate the efficacy of our approach.

The remainder of this paper is organized as follows. In Section 2, we discuss the related work most pertinent to this paper. In Section 3, we present detailed models for dynamic simulations and conduct quantitative analyses of the models. In Section 4, we design basic framework of the simulation-based approach of MEC service selection and propose an efficient scheme by applying OO techniques. In Section 5, we conduct real-life data based experiments to validate the efficacy of our scheme. Finally, we conclude the paper in Section 6.

2. Related Work

Service selection has been a hot topic in services computing community. The classical service selection was often a question of retrieving functional descriptions from service repositories and the ensuring that the described and required interfaces match a technical level [6]. Solutions regarding the functional aspects of service selection usually studied this problem from modeling or semantics angle [7, 8]. With the rapidly growing number of functional similar services being available on the Internet, the research on service selection has focused on the QoS issues. The very foundation of QoS-aware service selection is to evaluate the QoS of the services. The most straightforward type of the approaches is to design and implement some hardware equipment or computer programs and deploy them in the real-life running systems or emulators to obtain QoS metrics. Existing approaches took full advantage of measurement and feedback techniques for improving the accuracy of the QoS evaluation [9]. Another type of existing approaches is prediction-based scheme, which uses the historical QoS data from previous users of invoked services to predict the QoS metrics by a current user on the particular service before the invoking. Since the prediction-based approaches allow for data missing, they appear powerful strength in QoS evaluation especially in large-scale services computing systems. Several theories and techniques have been applied for improving the predictive quality of the QoS values, and examples include Group Decision Theory [10], collaborative filtering [11], and neural network [12]. The third type of QoS evaluation is model-based method, which is to build a mathematical model for formulating the dynamics of the services and conduct quantitative QoS analysis according to the model parameters. Services can be modeled by Markov chains [13] or queueing models [14] and QoS attributes can be calculated even before services being implemented and deployed, according to which optimal service selection policies can be obtained. Finally, some of the researchers studied this problem from a simulation point of view. System dynamics were analyzed, based on which simulation experiments were designed and conducted for achieving the performance evaluation [15].

With the analytical results, service selection can be studied by solving its correlated optimization problem. Several aspects could be paid attention to while selecting optimal service for users. For example, the diversity of user demands and preferences should be carefully considered especially in the formulation of optimization objectives and constraints [16]. Trustworthiness should be formulated and optimized for better meeting the QoS requirements of the users [17]. Sometimes, the service selection can be formulated as a multidimensional, multiobjective, and multichoice problem, which is so complex to be NP-Hard and can only be solved approximately by heuristic algorithms [18].

Service selection in mobile edge computing may face new characteristics and challenges. Due to the emergence of the edge layer, several additional aspects may affect the

performance of the system, bringing in more uncertain factors to end-to-end QoS. For instance, task arrivals to the MEC system [19], data processing rates in both edge layer and cloud site [20], and service scheduling strategies [21, 22] would all affect the dynamics of the systems, making service selection problem more complex. Therefore, how to design and implement efficient and practical solutions of QoS-aware service selection capturing the dynamics and characteristics of the MEC systems remains largely unexplored. Our simulation-based approach, to be described next, is designed to fill this gap.

3. System Models

In order to solve the service selection problem in mobile edge computing, we have to firstly formulate the problem theoretically. To do so, we study the dynamic behaviours of both mobile users and servers. Mathematical models that capture the dynamics of the systems are presented, and then quantitative analyses are provided. With the analytical results, the problem is hence formulated by an optimization model.

3.1. User Mobility Model. In a mobile edge computing system, we suppose that there are N users, each of which requests for certain services that are deployed on either an edge server or the cloud server. For one of the users $i \in \{1, 2, \ldots, N\}$, let $p_i(t)$ represent the geographical position of the user at the time point t. Since the physical location of a user in an MEC system is usually changing with time, $p_i(t)$ is defined as a stochastic variable, whose distribution varies for different users in different systems.

In an MEC system, a user connects to the system via a cellular base station, which is also an edge server. We suppose that there are M edge servers in the mobile edge computing system. Each of the servers runs several services fulfilling requests from its accessible users. Actually, we usually do not care about the physical location of a user. Instead, we concern which edge server the user connects. Therefore, the value of the stochastic variable $p_i(t)$ can be set by the ID of the accessed edge server, i.e., $p_i(t) \in \{1, 2, \ldots, M\}$.

Computer systems always operate in discrete-time fashion, since each of them is equipped with an internal discrete-time clock. Therefore, the time line can be thought of as a sequence of intervals defined by a sequence of points $t_0 < t_1 < \ldots < t_k < \ldots$. Thus, the collection of $p_i(t)$ is a stochastic process representing the mobility of a user in an MEC system. This can be regarded as a general user mobility model, which can be applied in most types of MEC systems in reality.

3.2. Queueing Model for Edge Servers. The dynamics of an edge server include the following three fundamental parts. Firstly, requests arrive at the server for completing certain tasks according to their requirements. Secondly, since the computational/storage resources of the server are not unlimited, the requests sometimes have to wait in queues until the service is available. Otherwise, arriving requests immediately proceed to the service without queueing. Thirdly, the requests get served and finally depart from the server. Such dynamics can be well captured by a queueing model, which can be represented graphically by Figure 1. The circle represents an edge server, and an open box represents a buffer (queue) preceding this server, where the slots in the queue are meant to indicate waiting requests. The requests are thought of as arriving at the queue and departing from the server, and it is assumed that the service process normally takes a strictly positive amount of time.

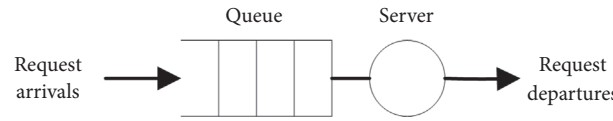


Figure 1: Queueing model of edge server.

The queueing model can be formulated by a discrete event system (DES), where its "events" consist of a sequence of task arrivals and departures. Specifically, let $w_j(t)$ denote the workload (number of requests in the queue) of server $j \in \{1, 2, \ldots, M\}$ at time t, and thus $w_j(t)$ is a stochastic variable which can be defined as the state of the queueing model. In addition, we define $a_j(t, t+1)$ to represent the number of arrival requests at server j in the time interval from t to $t+1$, and $d_j(t, t+1)$ to express the number of departures in the same time interval. Suppose that the edge server has a limited buffer and its buffer size is denoted by B_j. Therefore, the dynamics of the states of the queueing model can be captured by the following expression:

$$w_j(t+1) = \min\left\{B_j, \max\left\{0, w_j(t) + a_j(t, t+1) - d_j(t, t+1)\right\}\right\} \quad (1)$$

The average queue length of the edge server j can be obtained by calculating the mean of $w_j(t)$, as follows.

$$q_j \equiv \mathbf{E}\left[W_j\right] = \lim_{T \to \infty} \frac{1}{T} \sum_{t=0}^{T} w_j(t) \quad (2)$$

With *Little's law* which provides an all-purpose steady-state performance analysis of queueing systems with any stochastic clock structure, the average response time is proportional to the mean queue length with $1/\lambda_j$ being the constant of proportionality where λ_j is the average arrival rate of requests at edge server j.

$$T_j = \frac{1}{\lambda_j} \mathbf{E}\left[W_j\right] = \frac{q}{\lambda_j} \quad (3)$$

$$\lambda_j = \lim_{T \to \infty} \frac{1}{T} \sum_{t=0}^{T} a_j(t, t+1) \quad (4)$$

For each of the user requests handled by an edge server, the response time, also known as the sojourn time, can be calculated by the total time that the request spends in the queueing system, which is the summation of its waiting time and its service time. Suppose the arrival time of a request $k \in \{1, 2, \ldots\}$ is denoted by $t(A_k)$ while its departure time

is expressed as $t(D_k)$, and thus the response time can be obtained by (5). We should note that although intuitive, this expression is with value especially for simulation-based schemes, referring readers to Section 4 for details.

$$t_k = t\left(D_k\right) - t\left(A_k\right) \tag{5}$$

3.3. Service Selection Model. The service selection is to determine either the local service located on an edge server or the remote one deployed on cloud site will handle the user requests. Since an edge server can be regarded as an intermediate layer between users and cloud that is able to process in part workload and services locally [23], it usually brings in a performance enhancement especially for some lightweight realtime tasks. However, an edge server is commonly a virtualized lightweight cloud server deployed in a base station, and thus its capacity is much lower than a cloud server. If the workload of an edge server exceeds a certain level resulting in a heavy workload or request congestion, the end-to-end performance decreases dramatically. As a result, the service selection strategy has to be dynamic according to the workload status of the edge servers.

We let a stochastic variable $x_i(t)$ denote the decision variable of service selection in the time epoch of t. We suppose that $x_i(t) = 1$ indicates that the requests submitted by user i will be handled by the accessed edge server $p_i(t)$, while $x_i(t) = 0$ represents that the requests in this time epoch will be submitted to the cloud site. With such definitions, one can obtain that if $x_i(t) = 0$ then there will be nothing submitted to the edge server during the time epoch, and hence (1) can be expressed more specifically by

$$w_j\left(t+1\right) = \begin{cases} \min\left\{B_j, \max\left\{0, w_j\left(t\right) + \sum\limits_{p_i(t)=j} a_i\left(t, t+1\right) - d_j\left(t, t+1\right)\right\}\right\}, & x_i\left(t\right) = 1; \\ \max\left\{0, w_j\left(t\right) - d_j\left(t, t+1\right)\right\}, & x_i\left(t\right) = 0. \end{cases} \tag{6}$$

Considering that the services deployed on the cloud usually have guaranteed QoS via SLA, we assume that the response time of a cloud service to a user request is nearly deterministic. However, as cloud services are often deployed on a remote cloud site, there should be an additional part of the end-to-end response time for a cloud service, i.e., the network communication delay. Comparing with the response time of the services on the edge server, this delay is usually longer. We use (7) to express the end-to-end response time of cloud service for request submitted by user i, where T_i^C is the response time at the cloud site and T_i^N is the communication delay.

$$T_i = T_i^C + T_i^N \tag{7}$$

The service selection scheme is to minimize the response time of all the requests submitted by users, which is one of the most popular approaches for guaranteeing the end-to-end QoS. The decision of service selection is made periodically, and during each time epoch the service selection policy remains the same. We should note that such scheme is time-driven; i.e., the decisions are made at certain time points. One may also design and implement the service selection scheme as an event-driven one; i.e., a decision has to be made upon the occurrence of any event (e.g., arrival and departure). This is a special case of time-driven scheme when we set the time points of making decision to the exact time when events occur. However, the event-driven approaches are usually too costly that the dispatchers have to obtain the optimal solution upon every event resulting in significant overhead. Thus, in the following parts of this paper, we focus on the time-driven schemes which are much more practical in reality.

In a steady-state point of view, the service selection scheme should minimize the average response time of all the user requests. Since the states of edge servers (numbers of requests in the queues) vary with time, the scheme should also be dynamic. Therefore, the dynamic service selection is formulated by a dynamic optimization problem as follows.

Decision Epoch. A decision epoch is indexed by $n \in \{0, 1, \ldots\}$ and correspondingly a decision is executed at time $t = t_0, t_1, \ldots$. If each decision epoch takes equivalent time unites (i.e., τ), then the decisions are made at time $t \in \{0, \tau, 2\tau, \ldots\}$.

States and State Space. The state $S(n)$ at the beginning of epoch n is defined by the state of the system. Since the system consists of M edge servers, we define

$$\begin{aligned} S\left(n\right) &= \left[w_1\left(t_n\right), w_2\left(t_n\right), \ldots, w_M\left(t_n\right)\right] \\ &\in S = B_1 \times B_2 \times \ldots \times B_M \end{aligned} \tag{8}$$

Actions and Action Space. The action of service selection at time epoch n is expressed by

$$a_n = \left[x_1\left(t_n\right), x_2\left(t_n\right), \ldots, x_N\left(t_n\right)\right] \in A = \{0, 1\}^N \tag{9}$$

Optimization Objective. The objective of QoS-aware service selection is to minimize the average response time of all the user requests submitted by users to the system, by tuning the decision variables $x_i(t)$. Mathematically, the objective function is expressed as follows:

$$\underset{a \in A}{\text{maximize}} \quad \mathbf{E}\left[t_k\right] = \lim_{K \longrightarrow \infty} \frac{1}{K} \sum_{k=1}^{K} t_k. \tag{10}$$

We should note that the response time t_k of request k is closely related to the decision made upon its arrival. If the service on the edge server is selected, t_k should be calculated by (5). Otherwise if being served by cloud servers, the request will be responded in T_i time expressed by (7).

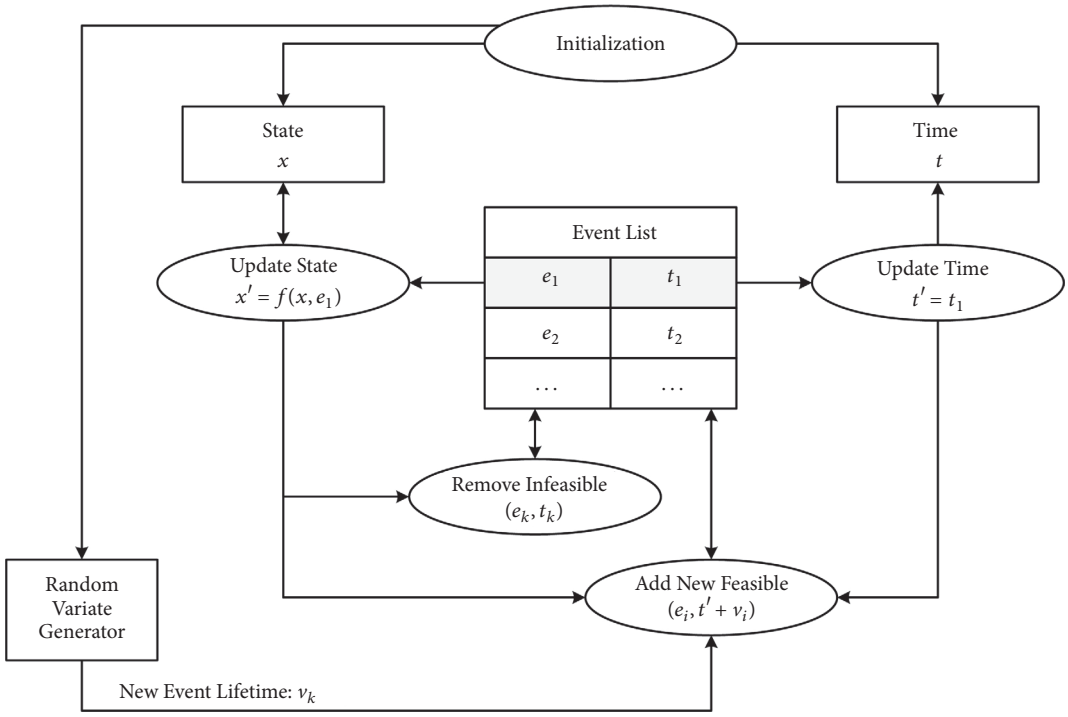


Figure 2: Framework of event-driven simulation.

4. Approach

4.1. Simulation-Based Optimization. With the models presented in the previous section, the service selection problem has been well formulated and can be furthermore solved by optimization theories and techniques. The optimization problem defined by (10) is similar to a Markov Decision Process (MDP) which is a well-known dynamic optimization problem being studied for decades. However, there is a significant difference. The MDP always assigns a reward to each of the states, and the optimization objective is to maximize or minimize the expected value of the reward, either discounted or undiscounted. But for our problem shown in (10), the objective is to minimize the average of response time which is a transient time-varying variable. The transient response time is correlated to both of the status of queues and policies after decisions having been made. Therefore, the Markovian (memoryless) property sometimes may not hold, resulting in significant difficulty in optimization procedures. Furthermore, real-world systems do not conform to some assumptions (e.g., Poisson arrivals, exponential service rates) we make in order to simplify a model, and they are too complex to yield analytical solutions. Since both analytical and algorithmic solutions may fail for solving this problem, we design a simulation-based approach.

The basic idea of simulation is to conduct a series of experimental processes through which the system models are evaluated numerically, and the data from the processes are used to estimate various quantities of our interests. The states of the physical system are represented by state variables, and the simulation program modifies them to reproduce the evolution of the physical systems over time. With different policies, the performances of the system are evaluated and compared, based on which the optimal solution will finally surface. The most appealing advantage of simulation-based optimization is its relative simplicity and wide applicability, especially comparing to old-fashioned laboratory experiments in which the real physical systems have to be implemented. The only hardware involved is a computer, and instead of physical servers connected with each other we have software programs capturing all such interactions and activities. Randomness is also fully considered, and replaced by appropriate software driven by random variate generator.

Although the service selection scheme is time-driven, we implement our simulation using an event-driven fashion, in order to capture all the dynamics of the whole MEC system. The overall framework of the simulation is shown by Figure 2. The basic procedure of a simulation experiment is to continuously repeat the following five steps. (1) The first entry (e_1, t_1) from the *event list* is selected and removed. (2) We update the simulation *time* by advancing it to the new event time t_1. (3) The *state* of the system x is updated according to the state transition function $x' = f(x, e_1)$. More specifically, if e_1 is an arrival event, we have $x' = f(x, e_1) = x + 1$; otherwise if e_1 is a departure, and thus we obtain $x' = f(x, e_1) = x - 1$. (4) If there exist any infeasible events in the new state, we remove their corresponding entries from the event list. (5) The new feasible events triggered by e_1 will be added into the event list. The scheduled event time for event e_i is given by $t_1 + v_i$, where v_i is a lifetime obtained from the *random variate generator*.

The simulation-based approach of QoS-aware service selection in MEC systems is designed and implemented as

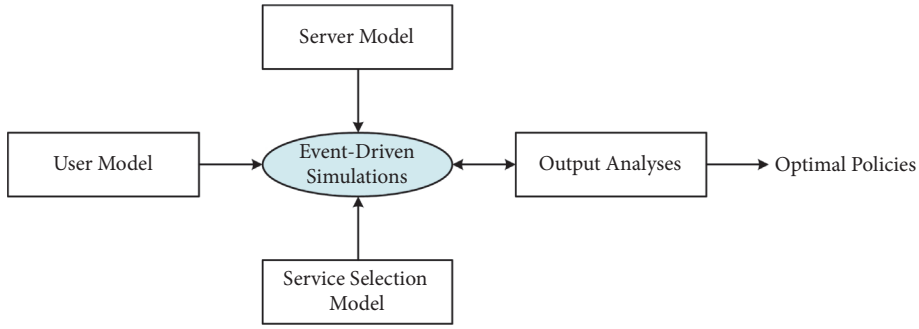


Figure 3: Framework of simulation-based service selection.

Figure 3. Based on the user mobility model, server queueing model, and service selection model presented in the previous section, we conduct simulation experiments for certain parameter settings and analyze the output (e.g., average response time of user requests). With different policies, the output is analyzed and compared, among which the optimal solution can be obtained.

4.2. Goal Softening. The conventional simulation-based optimization is to experimentally evaluate all the candidate policies and select the one with the optimal objective. However, such type of approach meets challenges especially in the long-term optimization of the service selection problem discussed in this paper.

First and foremost, the search space is extremely large. From the discussions in Section 3.3, we obtain that the number of feasible policies in only one decision epoch is 2^N. Furthermore, in one of the simulation experiments, we have to make decisions in a series of decision epoches, resulting in a further exponential increase of the search space. The number will become extremely large especially when the scale of the MEC systems grows up. Consequently, there is a critical limit of the scalability of the conventional simulation-based optimization approaches.

Secondly, the number of random variables is also extremely large. For each of the events in the simulations, i.e., task arrivals and service processes, the random variate generator has to generate a new random variable in order to obtain new feasible events. In each of the decision interval, there might be a number of event having occurred, and thus more events are generated. In an MEC system with multiple edge nodes, the event table shown in Figure 2 grows very fast. Since plenty of correlated data has to be recorded and calculated, both the overall performance and the memory space will face critical challenges in the simulation programming.

Last but not least, since the workload in reality is usually dynamic and varies with time, we have to simulate quite a long time period in order to capture the characteristics of the workload for guaranteeing the accuracy of the simulation results. However, such is quite time-consuming and resource-consuming. For some large-scale MEC system, it is impossible for us to conduct such long-term simulations for every feasible policy with existing computer systems.

Therefore, in order to attack these challenges, we have to sacrifice some part of the optimality for solving the optimization problem in reasonable time. Here, we borrow the idea from ordinal optimization (OO) which was firstly proposed by Ho et al. [24] for solving extremely complex search-based optimization problems. The basic idea of OO is "soft optimization for hard problems," which means that one can solve the hard problem within an acceptable time after softening the optimization objective.

Since we have known that finding the global optimal solution is practically infeasible, we switch our goal to a reasonable one which is to find a good enough solution with high probability. Mathematically, the goal is expressed as

$$\Pr\left(|G \cap S| \geq k\right) \geq \alpha \tag{11}$$

Here, G is the actual good enough set which is usually the top-g feasible solutions for the optimization problem, where $g = |G|$. S is the set of selected solutions by OO which is usually the estimated top-s solutions selected by simulation experiments, where $s = |S|$. $\Pr(|G \cap S| \geq k)$ is called *alignment probability* which indicates the probability that there are actually k truly good enough solutions in S, and k is called the alignment level. We should note that, in most cases where the users usually select the best one solution in S, k can be set to 1.

The first basic idea of OO is that ordinal optimization is much easier than cardinal optimization. A traditional cardinal problem usually asks us to estimate the difference in performance between two policies. In simulation-based optimizations, since all the simulation experiments are identical and independent, the overall performance (e.g., average response time) is calculated using a mean estimator by averaging all the experimental results obtained from the simulations. We suppose that there are l identical and independent simulation experiments obtaining a sequence of i.i.d. samples of the estimate observations expressed by $X(i)$ $(i = 1, 2, \ldots, l)$. Consequently, the estimator is expressed as $\widetilde{X} = (1/l)\sum_{i=1}^{l} X(i)$. We let X denote the actual value of the overall performance. With *Strong Law of Large Numbers*, we have $\lim_{l\longrightarrow\infty}\widetilde{X} = \lim_{l\longrightarrow\infty}(1/l)\sum_{i=1}^{l} X(i) = X$, indicating that the estimator is unbiased. Furthermore, we can conclude that the standard deviation of the mean estimator can be calculated by $\sigma_{\widetilde{X}} = (1/\sqrt{l})\sigma_X$, which shows that such estimate's convergence rate of no faster than $l^{-1/2}$ is unsatisfactorily slow. On the other hand, the *Central Limit Theorem* illustrates that the sample mean $\widetilde{X}$ converges to

1: Use a crude and computationally fast model to estimate the performance of all feasible policies.
2: Estimate the Ordered Performance Curve (OPC) class of the problem and the noise level of the crude model.
3: Use the table presented in [5] to calculate the size of selected set s.
4: Select the estimated top-s policies by the crude model as the selected set S.
5: Simulate each policy in the selection set with a precise model.
6: Apply the best policy from the simulation results in step 5.

Algorithm 1: OO-Based Service Selection Scheme for MEC.

Gaussian distribution when l is large enough. Therefore, if we only want to determine whether a policy is better than another one by comparing their simulation output $\widetilde{X}_1$ and $\widetilde{X}_2$ following OO spirit, we find that the differential $\widetilde{X}_2 - \widetilde{X}_1$ also conforms to Gaussian distribution. Suppose that the actual values have the relationship as $\Delta X = X_2 - X_1 > 0$, and thus we have $\Pr(\widetilde{X}_2 - \widetilde{X}_1 > 0) = \Phi(\sqrt{l}\Delta X) = 1 - O(l^{-1/2} \cdot e^{-\Delta X^2 l})$, showing the order between estimates of the two results agreeing the true order converges exponentially to 1 with rate no slower than ΔX^2. In conclusion, with much higher convergence rate, OO is easier than conventional cardinal optimization.

The second idea of OO is that the optimization with a softer goal of being good enough is much easier than trying to find the exact one best solution. This relaxing of the goal can buy us a lot in the easing of the computational and memory burden, meanwhile guaranteeing the optimality in an acceptable level. We discuss a blind picking scheme which is able to provide a lower bound analysis for the alignment probability. Such scheme is to just randomly pick s policies from the search space to obtain the selected set S, and thus no prior knowledge is used in the selection. Therefore, we have the alignment probability when $k = 1$ given by the following expression, where Θ represents the overall search space.

$$\begin{aligned} \Pr(|G \cap S| \geq 1) &= 1 - \frac{\binom{|\Theta|-g}{s}}{\binom{|\Theta|}{s}} \geq 1 - \left(1 - \frac{g}{|\Theta|}\right)^s \\ &\geq 1 - e^{-gs/|\Theta|} \end{aligned} \tag{12}$$

Therefore, the alignment probability converges exponentially with respect to the size of the set G and S. Furthermore, the lower bound of the alignment probability has a general form as follows:

$$\Pr(|G \cap S| \geq k) \geq \sum_{i=k}^{\min(g,s)} \frac{\binom{g}{i}\binom{|\Theta|-g}{s-i}}{\binom{|\Theta|}{s}} \tag{13}$$

4.3. Service Selection Scheme. We take advantage of OO techniques to solve the dynamic service selection problem. After goal softening, an efficient simulation-based scheme for dynamic service selection in MEC systems is carefully designed. The basic procedures of our approach are shown by Algorithm 1.

We firstly use a rough model to get the top-s candidate policies. The crude model will provide a rough estimation of the performance of each policy, but it is quite efficient in both computational operations and memory space. In the optimization problem defined by service selection in MEC presented in Section 3, simulation parameters are carefully controlled. Since the resource consumption is closely correlated to the simulation time, we implement our crude model by estimating the performance of the feasible policies using simulation experiments for only a relatively small time period. In other words, although the parameter K in (10) should be large enough to obtain an accurate estimate of the average response time, we set it a very small number in our crude model, which makes the simulations complete in a short time resulting in small resource consumption. Consequently, step 1 can be completed in a reasonable time.

In the next step we determine the parameters of ordinal optimization. The user specifies the size of good enough set g and the required alignment level k. With well-developed theoretical and practical results, after estimating the Ordered Performance Curve (OPC) class of the problem and the noise level of the crude model by our approach, we are able to obtain the appropriate size of selected set s by looking up a precalculated table [5].

After the top-s feasible policies have been selected by the crude model, we apply the precise model simulation on the s candidate policies to find the optimal one. We notice that the precise model simulations are much more expensive than the previous crude version, which takes much more time and computational resources. However, we only need to run the precise simulations on the s policies, and thus comparing to the original search space Θ the overhead is quite acceptable. Also, we will illustrate such issue in the next section by the experimental results obtained from simulation experiments we conduct in reality.

5. Evaluation

5.1. Experimental Setup. We conduct experiments based on real-life data to validate the efficacy of our approach. The QoS-aware service selection scheme introduced in the previous sections is implemented in simulation experiments, where workload is generated according to real-world trace data and OO theory is applied to find a good enough solution in a reasonable time.

We apply a real-world data set released by Microsoft Research, namely, "T-Drive" to generate the workload in our experiments. It contains the GPS trajectories of 10,357 taxis collected by GPS loggers and GPS phones within the city of Beijing during a period of one week in the year of 2008 [25, 26]. The exact time of each piece of data being submitted to

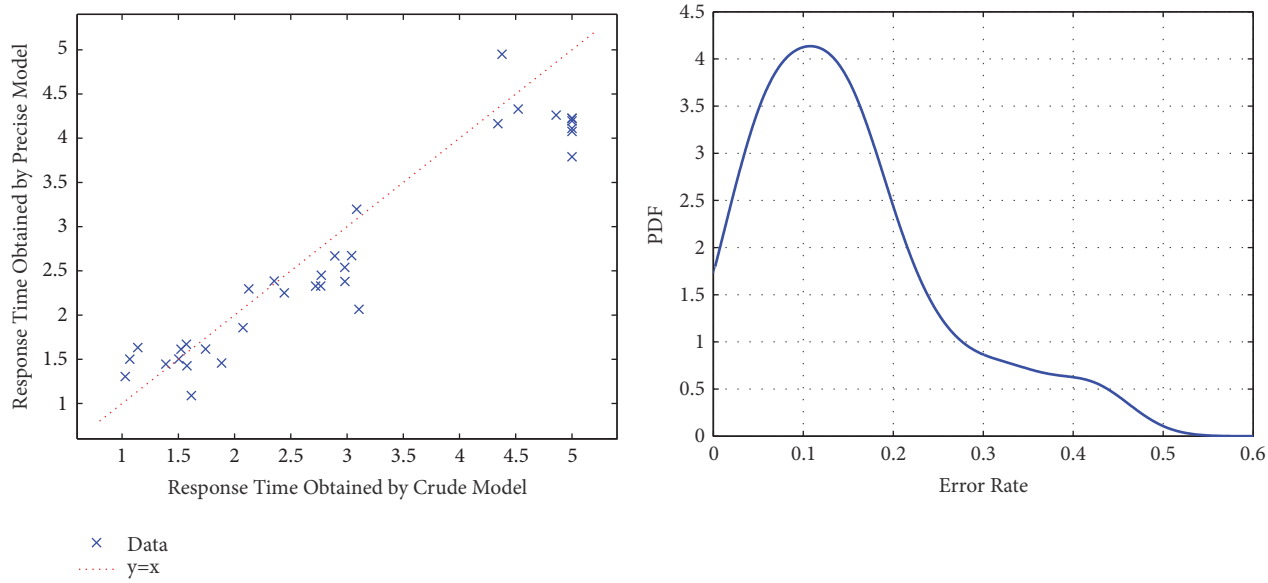


Figure 4: Estimation error of crude model from precise model.

the system has been recorded, which is of valuable reference of the task arrivals from different users. In our experiments, the time when the users initiate a request is followed by the timestamp of each piece of taxi GPS data. We assume that the users in our MEC system basically follow a random walk mobility model within different coverage areas of different edge servers, and we implement an exponentially distributed random variable generator for the service processing events in the edge servers. The service selection scheme is designed and implemented in a time-driven fashion, following the procedures presented in Section 4.

An MEC system is simulated on a PC environment with an Intel quad-core CPU and 8GB memory. A simulator is designed and implemented capturing the basic dynamics of the system behaviors, including request arrivals, task scheduling, and service procedures. With our approach, experimental data is collected and analyzed, which will validate the effectiveness and efficiency of our approach. We defer readers to the following subsection for details.

5.2. Experimental Results. We conduct the simulation experiments several times by tuning the parameters such as the number of decision epoches that we consider during the long-term optimization (i.e., K) and the number of edge servers in the MEC system (i.e., M). Also, both crude model and precise model are implemented. After running the simulation experiments, we illustrate the experimental results. We should note that, during our experiments, we find that the conventional simulation-based optimization approach takes so long time that we are not able to complete all the experiments in a reasonable time. Therefore, we illustrate the comparison between these two approaches within the limited cases.

Firstly, we evaluate the effectiveness of our approach. The response times obtained by the crude model and the precise model are shown by the first subfigure of Figure 4. The x-axis indicates the response time obtained using the crude model in our simulations, while y-axis illustrates the response time calculated by the simulation results using the precise model. We also draw a dot line of $y = x$ for clear demonstration. The closer the data points are to the dot line, the less estimating error the crude model has. We obtain from Figure 4 that most of the data points are close to the line. The second subfigure further analyzes the distribution of the error rate by illustrating its probability density function (PDF). It has been shown that the majority of the data is within the error rate below 20%. Quantitatively, we calculate the mean error rate of the crude model, showing that such value is around 16.3%. Considering the significant reward in reducing the search space of the crude model, such error rate is quite satisfactory.

Secondly, we tune the experimental parameters to discuss the size of search space with the increase of the scale of the optimization problem. The experimental results are shown by Figure 5. On one hand, we find that the search space of the precise model increases exponentially with the increase of the decision period time, indicating that the conventional simulation-based approaches have to spend plenty of time and resources for obtaining a long-term optimality. However, the search space of the crude model basically remains the same, since we only conduct a fixed short-term dynamic programming optimization for obtaining the selected set. Consequently, using crude model is robust to the increase of the long-term optimization. On the other hand, we find from the second subfigure that the search spaces of both crude model and precise model grow exponentially with the increase number of the edge servers. The reason is that the action space increases exponentially in this case, and both of the two models have to look for the optimal solutions in a much wider scope. However, it is clear that there is a significant difference between their increase rates. The size of search space in the crude model grows slower than precise

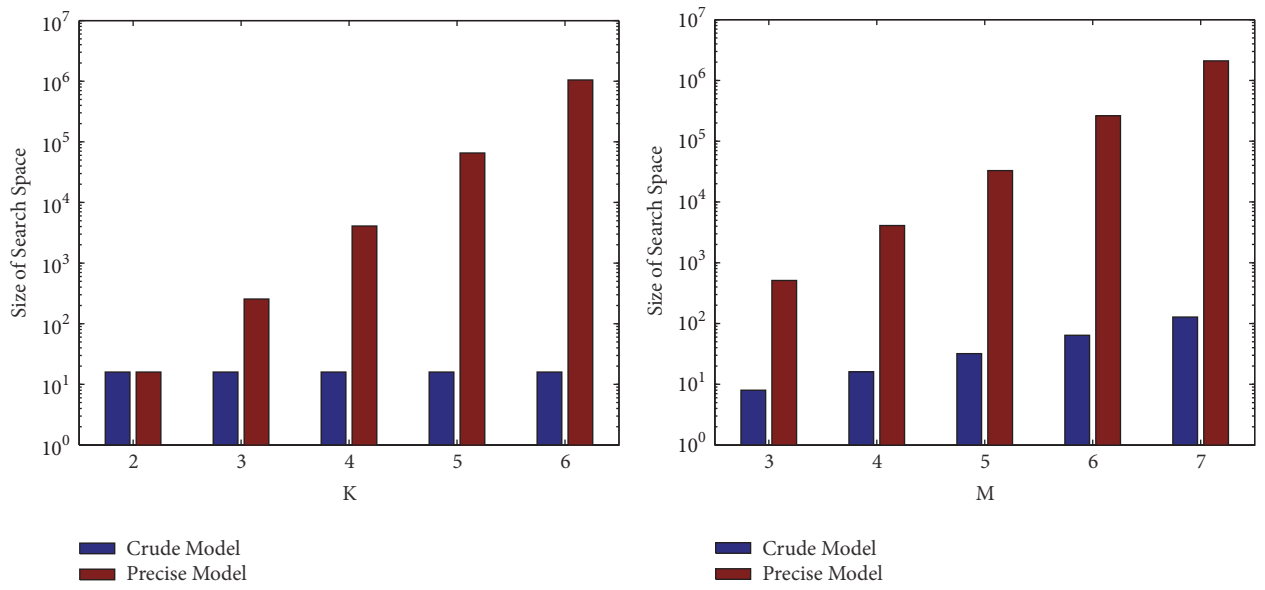


Figure 5: Size of search space of the two models.

Table 1: Experimental results of running time.

Scheme		Running Time
Traditional Approach		161min36s
OO	Crude Model	3s
	Precise Model	20min12s
	Total	20min15s

model. In summary, the effectiveness of our approach in reducing the search space can be validated.

Finally, we evaluate the running time of our approach comparing with the traditional simulation-based optimization. We show a group of experimental results for a small-sized MEC system, since the traditional approach is not able to complete the task in acceptable time if the search space grows larger. Shown as Table 1, one can obtain that the computational complexity of the crude model is significantly low. Applying ordinal optimization in search-based service selection optimization is able to nearly reduce the overhead by one order of magnitude, which validates the efficiency of our approach.

6. Conclusion

Service selection is an important and open problem in mobile edge computing for guaranteeing the quality of service. This paper proposes an efficient simulation-based service selection approach for MEC systems, tackling the challenges of state explosion and high variability in simulation-based optimization. Frameworks, models, analyses, and algorithms are discussed in detail, and empirical validation is conducted based on trace data obtained from reality. It has been proved both theoretically and experimentally that the performance can be significantly improved by slightly softening the optimization objective with a sacrifice of global optimality of the obtained solutions. This work is expected to bring a new idea for solving the optimization problems in large-scale MEC systems and provide with a practically efficient solution for optimal QoS-aware service selection.

Data Availability

Previously reported T-Drive dataset was used to support this study and is available at https://www.microsoft.com/en-us/research/publication/t-drive-driving-directions-based-on-taxi-trajectories/. This prior dataset is cited at relevant places within the text as [25, 26]. Most of the simulation experimental data used to support the findings of this study are included within the article. Further additional data are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work is supported by Beijing Natural Science Foundation (no. 4162042), National Natural Science Foundation of China (no. 61502043), and National Key Research and Development Plan (no. 2016YFC0303700).

References

[1] P. G. Lopez, A. Montresor, D. Epema et al., "Edge-centric computing: vision and challenges," *Computer Communication Review*, vol. 45, no. 5, pp. 37–42, 2015.

[2] A. Ahmed and E. Ahmed, "A survey on mobile edge computing," in *Proceedings of the 10th International Conference on Intelligent Systems and Control, ISCO 2016*, India, January 2016.

[3] F. Patrizi, "An Introduction to Simulation-Based Techniques for Automated Service Composition," *Electronic Proceedings in Theoretical Computer Science*, vol. 2, pp. 37–49, 2009.

[4] S.-C. Horng and S.-S. Lin, "An ordinal optimization theory-based algorithm for a class of simulation optimization problems and application," *Expert Systems with Applications*, vol. 36, no. 5, pp. 9340–9349, 2009.

[5] T. W. E. Lau and Y.-C. Ho, "Universal alignment probabilities and subset selection for ordinal optimization," *Journal of Optimization Theory and Applications*, vol. 93, no. 3, pp. 455–489, 1997.

[6] S. Reiff-Marganiec and M. Tilly, "Non-functional property based service selection: A survey and classification of approaches," in *Proceedings of the 2008 Non Functional Properties and Service Level Agreements in Service Oriented Computing Workshop*, 2008.

[7] P. C. Xiong, C. Pu, and M. C. Zhou, "Protocol-level service composition mismatches: A petri net siphon based solution," *International Journal of Web Services Research*, vol. 7, no. 4, pp. 1–20, 2010.

[8] M. Rouached, W. Fdhila, and C. Godart, "Web services compositions modelling and choreographies analysis," *International Journal of Web Services Research*, vol. 7, no. 2, pp. 87–110, 2010.

[9] S. Wang, A. Zhou, W. Lei, Z. Yu, C. Hsu, and F. Yang, "Enhanced User Context-Aware Reputation Measurement of Multimedia Service," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 12, no. 4s, pp. 1–18, 2016.

[10] Y. Ma, S. Wang, P. C. Hung, C. H. Hsu, Q. Sun, and F. Yang, "A highly accurate prediction algorithm for unknown web service QoS value," *IEEE Transactions on Services Computing*, vol. 9, no. 4, pp. 511–523, 2016.

[11] M. Tang, Z. Zheng, G. Kang, J. Liu, Y. Yang, and T. Zhang, "Collaborative web service quality prediction via exploiting matrix factorization and network map," *IEEE Transactions on Network and Service Management*, vol. 13, no. 1, pp. 126–137, 2016.

[12] W. Xiong, Z. Wu, B. Li, and Q. Gu, "A Learning Approach to QoS Prediction via Multi-Dimensional Context," in *Proceedings of the 24th IEEE International Conference on Web Services, ICWS 2017*, pp. 164–171, USA, June 2017.

[13] R. S. Matos, P. R. M. Maciel, and R. M. A. Silva, "QoS-driven optimisation of composite web services: An approach based on GRASP and analytical models," *International Journal of Web and Grid Services*, vol. 9, no. 3, pp. 304–321, 2013.

[14] L. Li, S. Li, and S. Zhao, "QoS-Aware scheduling of services-oriented internet of things," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 2, pp. 1497–1507, 2014.

[15] A. P. T. Hsu, W. T. Lee, A. J. C. Trappey, C. V. Trappey, and A.-C. Chang, "Using System Dynamics Analysis for Performance Evaluation of IoT Enabled One-Stop Logistic Services," in *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics, SMC 2015*, pp. 1291–1296, Hong Kong, October 2015.

[16] L. Yang, L. Liu, and Q. Fan, "A Survey of User Preferences Oriented Service Selection and Deployment in Multi-Cloud Environment," in *Proceedings of the 2017 18th International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT)*, pp. 354–359, Taipei, December 2017.

[17] H. Ma, Z. Hu, and M. Cai, "Trustworthy service selection integrating cloud model and possibility degree ranking of interval numbers," *Journal of Electronics*, vol. 26, no. 6, pp. 1177–1183, 2017.

[18] M. C. Jaeger, G. Mühl, and S. Golze, "QoS-aware composition of web services: A look at selection algorithms," in *Proceedings of the 2005 IEEE International Conference on Web Services, ICWS 2005*, pp. 807–810, USA, July 2005.

[19] Z. Wang, Q. Zhao, F. Xu, H. Dai, and Y. Zhang, "Detection performance of packet arrival under downclocking for mobile edge computing," *Wireless Communications and Mobile Computing*, vol. 2018, Article ID 9641712, 7 pages, 2018.

[20] G. Li, J. Wang, J. Wu, and J. Song, "Data Processing Delay Optimization in Mobile Edge Computing," *Wireless Communications and Mobile Computing*, vol. 2018, 2018.

[21] X. Lyu, H. Tian, L. Jiang et al., "Selective Offloading in Mobile Edge Computing for the Green Internet of Things," *IEEE Network*, vol. 32, no. 1, pp. 54–60, 2018.

[22] Li. W, Y. Xia, M. Zhou, X. Sun, and Q. Zhu, "Fluctuation-aware and predictive workflow scheduling in cost-effective Infrastructure-as-a-Service clouds," *IEEE Access*, 2018.

[23] R. Deng, R. Lu, C. Lai, T. H. Luan, and H. Liang, "Optimal Workload Allocation in Fog-Cloud Computing Toward Balanced Delay and Power Consumption," *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 1171–1181, 2016.

[24] Y. C. Ho, R. S. Sreenivas, and P. Vakili, "Ordinal optimization of DEDS," *Discrete Event Dynamic Systems*, vol. 2, no. 1, pp. 61–88, 1992.

[25] J. Yuan, Y. Zheng, C. Zhang et al., "T-drive: driving directions based on taxi trajectories," in *Proceedings of the 18th International Conference on Advances in Geographic Information Systems ACM SIGSPATIAL (GIS '10)*, pp. 99–108, November 2010.

[26] J. Yuan, Y. Zheng, X. Xie, and G. Sun, "Driving with knowledge from the physical world," in *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '11)*, pp. 316–324, August 2011.

Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 7619102, 11 pages
https://doi.org/10.1155/2018/7619102

Research Article

BTP: A Bedtime Predicting Algorithm via Smartphone Screen Status

Kun Niu,[1] **Shubo Zhang**,[2] **Haizhen Jiao**,[1] **Cheng Cheng**,[1] **and Chao Wang**[1]

[1]*School of Software Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China*
[2]*Software Engineering Architecture Design Department of Consumer Business Group, Huawei Technologies Co., Ltd., Beijing 100095, China*

Correspondence should be addressed to Kun Niu; niukun@bupt.edu.cn

Received 19 April 2018; Revised 17 August 2018; Accepted 8 October 2018; Published 22 October 2018

Guest Editor: Kok-Seng Wong

For smartphone service providers, it is of vital importance to recognize characteristics of customers. The process of recognizing these characteristics is generally referred to as user profile, which provides knowledge basis for business decisions, enables intelligent services, and brings unique competitiveness. As a basic component of user profile, bedtime could reflect lifestyle, health condition, and occupation of people. This paper presents a flexible algorithm named BTP (Bedtime Prediction), which is designed for predicting wake time and bedtime by analysing screen status of smartphone. BTP first collects screen status log data of user's smartphone and conducts preprocessing with a series of auxiliary user profiles. Then, it detects and records users' wake time and bedtime of one day by searching and combining major screen extinguish periods in the past 24 hours. Finally, BTP predicts future bedtime by matching current screen status sequence with all historical records. By applying BTP, most of night and morning scenario-based applications could provide more considerate services, rather than following fixed execution time like alarm clock. Experiments on practical applications prove that BTP can effectively predict wake time and bedtime without applying complicated machine learning algorithms or uploading data to server.

1. Introduction

From eras of keeping records by tying knots to big data, data analysis technique has been developing throughout the history of human civilization. With tide of mobile Internet, enterprises, and governments, even individuals have begun to exploit their own business to portable devices especially smartphones. At present, mobile Internet has already exceeded personal computer on number of users. To serve customers better, it is very important to identify their preferences and characteristics. Because of different life background, Internet business must take care of personalized needs. This way of identifying user characteristics is generally called user profile.

Accurate user behaviour prediction could help mobile Internet service providers optimize their business on extensive scenarios. For instance, system may preload application which has the highest probability of being used next moment to shorten users' waiting time. Getting up and going to bed are two critical incidents that normally indicate the start and end of human schedule and intelligent service as well. Besides, sleep pattern can reflect the health condition of a person [1, 2]. It is possible to unite sleep pattern with other characteristics such as age, gender, and working time to conclude subhealth. Furthermore, applications could give customized suggestions based on these conclusions. "Pzizz" [3] helps people fall asleep and "Sleep Cycle" [4] offers waking up service. Both these "lullaby" and alarm clock apps strictly depend on manual activation and detailed settings. It is meaningful to liberate users from repeated mechanical operations and make services more natural and flexible.

Smartphone, integrated with multiple sensors and chips, becomes a primary user data acquisition platform owing to tight connection with people. Many researchers have made researches and applications based on usage records from smartphones for modelling user profile. Reference [5] applied supervised learning methods to infer user profiles, such as religion, relationship status, spoken languages, countries of

interest, and whether the user is a parent of small children, by observing only a single snapshot of installed apps. Experiments on 200 smartphones show that for most traits the model can achieve over 90% precision. Reference [6] focused on analysing user profiles by mining usage information from multiple NFC-based applications, both on smartphone and server. Reference [7] firstly attempted to understand users through Service Set IDentifier (SSID) information. It also proposed a data cleaning and information enrichment framework for enabling user preference understanding based on the collected Wi-Fi logs. Reference [8] developed a method to predict the next app which improves home screen apps' usage experience.

This paper aims at solving two tasks. The first one is to detect getting up and going to bed incidents. The second one is to predict the next wake time and bedtime.

The issue of sleep study is put forward by medical care institutes at first. Hospital apply Polysomnography (PSG) [9, 10] to diagnose sleep disorders by records brains waves, oxygen level in blood, heart rate, and breathing, as well as eye and leg movements of participants. However, this method requires professional devices and specialized sleep environment, which is infeasible for daily use.

Wearable devices are the second choice. Devices are worn on the wrist, used to record movements and estimate sleep parameters which are called wrist actigraph (ACT) [11]. Although actigraph has been doubt the validity of identifying wake or sleep from a medical point of view, wearable device with fine identification algorithm is still fit for daily use owing to its low cost and portability. Reference [12] proposed using a wrist actigraph to estimate sleep time at an early stage. Reference [13] later developed a monitor system based on it and replacing manual calculation by scoring algorithms. And [14] further improved it, making the algorithms distinguish sleep from wakefulness in approximately 88% of minutes scored. In recent years, development of integrated circuit (IC) and Internet of things (IoT) enables miniaturization of sensors. Activity trackers like Fitbit wristband and smartwatches like Apple Watch make it possible to collect and analyse wrist movement data of users. Reference [15] employed an artificial neural network to classify sleep and wakefulness based on night wrist actigraph data. Reference [16] integrated unsupervised machine learning algorithms and domain knowledge heuristics to detect sleep or wake status.

Without extra device, there are also methods that utilize smartphone as a monitor and upload data to calculate sleep durations. Reference [17] presented a model named BES inferring sleep duration by analysing smartphone usage patterns. The features include external brightness, phone usage, motion status, and background noise. The collection of these attributes requires a resident monitoring application on smartphones. Reference [18] designed and implemented an application named Toss "N" Turn that can infer sleep and sleep quality by captured sensor data including sound amplitude, acceleration, light intensity and screen proximity, running apps, battery, and screen status. This paper verifies its validity by notify users to enter ground truth every morning.

However, there are three limitations of these methods listed above:

(1) Although wearable devices have developed rapidly in recent years, the market penetration rate is still limited. Worldwide shipments of wearable devices are 25.1 million, while there are 344.4 million smartphones (source: International Data Corporation, IDC). In addition, different manufactures would apply different types of sensors, and there is no unified API for developing software for all of them. This solution evidently limits intelligent services' coverage rate.

(2) Most of current methods demand transmitting data to web server and executing algorithms on server cluster. That would be involved in privacy protection issue. Uploading privacy data without users' explicit permission would cause severe punishment by law in many countries. General Data Protection Regulation (GDPR) regulated that data transfer requires user's consent in European Union. Besides, claiming various large amount data collection would cause disgust of users.

(3) The computational resources of smartphone are strictly limited. Massive and heavy computation would cause rapid loss of power, high occupancy of memory and kernel. Each of these factors would burden the running time of device and lead to decrease of user experience. Furthermore, computing environment on smartphone does not support machine learning especially deep learning algorithms very well.

For the second task, to the best of our knowledge, previous research is rare to be find. Literatures retrieved by keywords like "sleep time prediction" always focused on how to detect user's wake/sleep status or calculate sleep duration. In this paper, we classify them as solutions to task one, detection. Since intelligent services based on user profile and usage scenario just become popular recently, lack of study on predicting wake time and bedtime is reasonable, for example, Google released Awareness API on I/O developer conference, 2016 [19]. Awareness API focuses on enabling context-aware services by mining user habits. Alarm clock would not disturb your sleep if it learned when you fall asleep last night and next meeting. BTP takes a step forward, attempt to predict user's future action which is a valuable and novel work.

The contributions of our work are as follows:

(1) A lightweight predicting algorithm: BTP simply utilize data collected from smartphone without additional wearable devices. Screen status log data could be retrieved from smartphone operating system because screen light up and extinguish event are standardized system broadcasts. Besides, thanks to the ubiquity of smartphone, BTP can cover the widest potential user.

(2) Avoid the dilemma of collecting data without breaking privacy protection regulations. BTP is also not involved with web service or cloud computing. In other words, BTP plays a role of data processor rather than controller. Users would not worry about the risk of personal information disclosure.

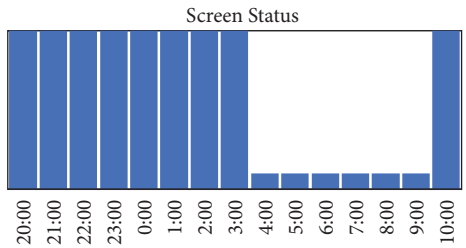


Figure 1: Back home late case (the user always goes to bed around 22:00).

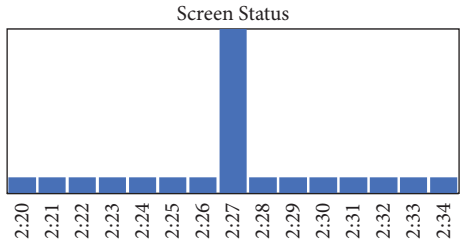


Figure 2: Unexpected interruption case (there is an incoming call at 2:27).

(3) BTP consume limited resource on smartphone. It does not need special framework or extra library. The calculation would not cause any appreciable delay. By our experiments, BTP performs acceptable complexity.

2. Ideas

2.1. Difficulties. Detection and prediction process of BTP may be crushed by many incidents. It is not enough to use screen status log data only. In actual projects and researches, we met the following difficulties.

The first is abnormal behaviour. Someone lives regularly, but back home very late or stay out all night occasionally. This case should be determined as outlier in data analysis theory. BTP contains models based on pattern knowledge, so these outliers must be detected and abandoned. Figure 1 demonstrates a typical outlier.

The second is unexpected interruption. When people are sleeping, incoming calls, receiving messages, and notifications from applications may light up screen, ring, or vibrate and then cause waking up. It is also possible that people fall into deep sleep and ignore these events. But for screen status log data, a continues screen extinguish period is interrupted by a strange event. The screen states of an unexpected interruption case is shown in Figure 2.

The third is shutting down of smartphone. If smartphone is shut down, both screen status and auxiliary events are missing. Some people do not like flight mode, they shut down smartphone at bedtime and start up it when getting up. But a worse case is, user may forget to charge the battery and then causing smartphone power off automatically. Losing log data is the biggest threaten and risk of BTP. Figure 3 shows the screen states of a shutting down case.

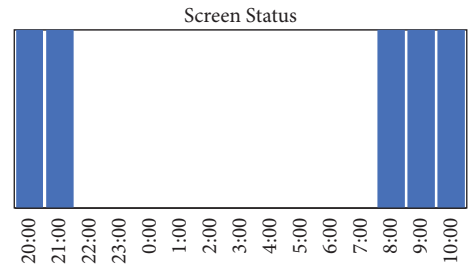


Figure 3: Shutting down case (the user shut down smartphone before 22:00 and starts up it after 8:00).

The above three difficulties are mentioned with detection, and the fourth is about prediction. Because of different life experience, regulations of wake time and bedtime between people are customized.

Most of people works at office or factory, that is, a stable target location. They show an identifiable wake-up time and bedtime distribution during weekdays. But many of them are freestyle on weekends. Others do not work from Monday to Friday. Because of their occupational characteristics, they exhibit strange distributions, such as going to work every two days or three days' work and one day's rest. That leads to big error in predicting with simple time series models or trying to describe these patterns by weekday and weekend frame. Figure 4 shows the patterns of weekdays and weedends.

2.2. Solutions. Besides difficulties, we can utilize two advantages to fix and improve our model. One is alarm clock event. For most people, they go to work at weekdays. Alarm clock ringing moment is also wake time. This rule has very high confidence and we can check accelerator activated event to confirm that. Figure 5 shows screen states when alarm clock event occures.

The other is accelerator activated event. When accelerator is activated, we know that smartphone is moved and the user is awakened. Figure 6 shows screen states when accelerator activated.

We overcome and reduce the effect of the four difficulties mentioned above by the following solutions.

(1) Abnormal Behaviour. BTP supposes that people sleep at home. It abandoned sequences not at home to avoid the impact of abnormal behaviour. Figure 7 shows screen states after abandoning abnormal behaviours.

(2) Unexpected Interruption. Sleep disturbances are resolved by merging adjacent periods of screen extinguish status. Here we define central sleep point, about 3 o'clock, the sleepiest time of the day. When people are waked by interruption event near central sleep point, they are likely to remain sleep in a few minutes. But when they are less sleepy, such as 5:00, people may get up directly. Figure 8 shows the case about connections two adjacant periods.

(3) Shutting Down of Smartphone. For the users who switch on and off regularly and reasonably, BTP regard the power on and off events as wake time and bedtime. But we treat

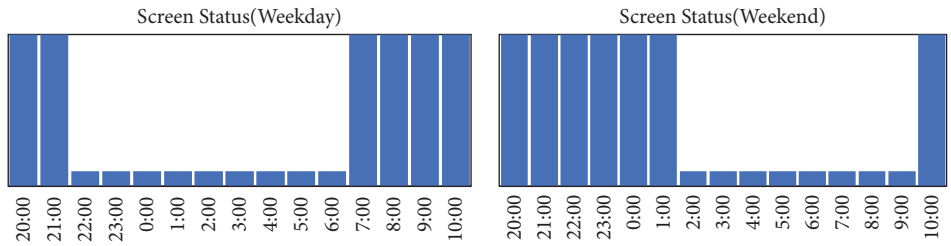


Figure 4: Weekday and weekend patterns (in weekend, go to bed and get up later than weekday).

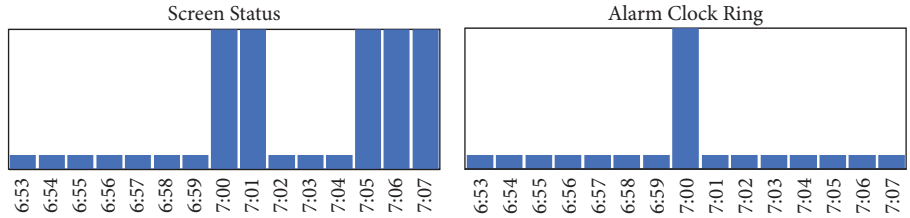


Figure 5: Alarm clock event (alarm rings at 7:00 and wakes the user).

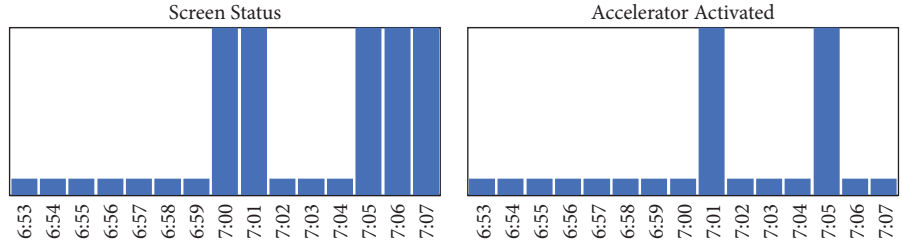


Figure 6: Accelerator activated event (the user touches screen at 7:01 to stops ringing and gets up at 7:05).

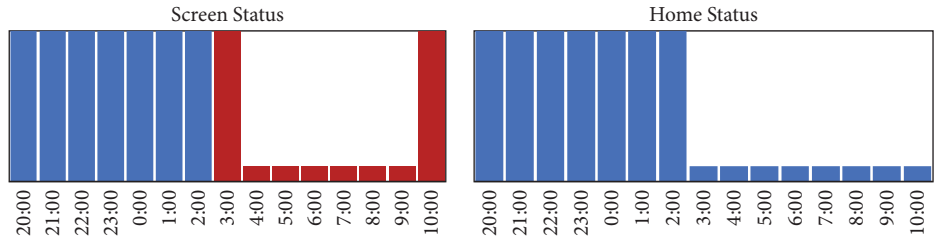


Figure 7: Abandon abnormal behaviour (the sequence after 3:00 is abandoned for going out home).

Figure 8: Connection of adjacent period (screen light up at 2:27 is ignored by BTP).

unregular power off event as outlier and abandon it. Figure 9 shows screen states when power off event occured.

(4) Customized Regulations between People. We used to think that user behaviour prediction should be based on weekday and weekend frame, but it is confused by customized weekday regulations, resulting in greater error. Then we changed our strategy and matched the current screen status sequence with other sequences of each past day. The most similar sequences represent the same behaviour pattern. Since waking up and

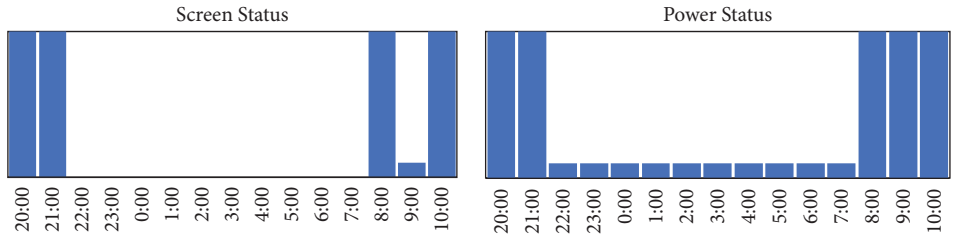


Figure 9: Reasonable power off event (when powering off, screen status is "null" but not "0").

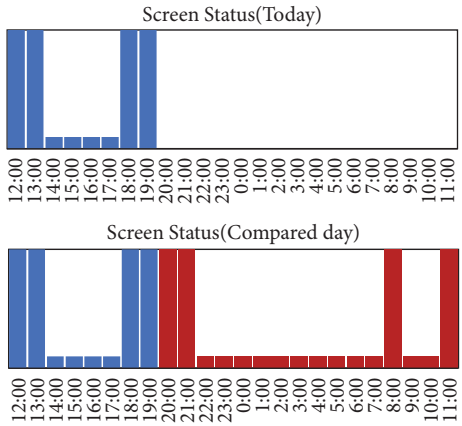


Figure 10: Similar behaviour pattern (next bedtime and wake time are predicted as 22:00 and 8:00).

going to bed are beginning and end of whole day activities, they are naturally reflected in this similar pattern system. Figure 10 shows two similar behaviour patterns.

To sum up, we effectively improving the accuracy and availability of screen status sequence in detecting and predicting wake-up time and bedtime by introducing a series of auxiliary events. The main process of BTP is shown in Figure 11. BTP firstly collects the user's screen status log data and other auxiliary events and then searches and connect main screen extinguish periods to get daily detection result. Then prediction is operated by matching current screen status sequence with the historical sequences like K-Nearest Neighbour.

3. Methods

3.1. Data Structure. Firstly, we define data structure. Log data contains information about both screen status and auxiliary factor. An event consists of date, moment, and event. Here is an example of JSON string.

```
{
 "Date": "2018-09-01",
 "Moment": "22:39:02",
 "Event": "SON"
}.
```

Table 1: Event code.

Target	Event Code	Meaning
Record screen status	SON	Screen light up
	SOFF	Screen extinguish
Filter period not at home	AH	Arrive at home
	LH	Leave home
Filter unexpected interrupt	IC	Incoming call
	RM	Receiving message
	AN	App notifications
Record alarm clock status	ACR	Alarm clock ring
Record power status	PON	Power on
	POFF	Power off
Decide awakened or not	AA	Accelerometer activated

Different events are distinguished by event codes. Here the code SON represents screen light up. All the event codes and corresponding meanings are shown in Table 1.

For task one, detection of wake time and bedtime, we input log dataset with all types of events from a smartphone throughout many days. Here is part of it.

```
{...
 {"Date": "2018-09-01", "Moment": "22:39:02", "Event":
  "SON"}, //Screen light on;
 {"Date": "2018-09-01", "Moment": "22:45:02","Event":
  "SOFF"}, //Scree extinguish;
 {"Date": "2018-09-01", "Moment": "22:49:50", "Event":
  "AA"}, //Pick out phone;
 {"Date": "2018-09-01", "Moment": "22:49:55", "Event":
  "SON"}, //Unlock phone;
 {"Date": "2018-09-01", "Moment": "22:50:01", "Event":
  "SOFF"}, //Screen extinguish;
 {"Date": "2018-09-01", "Moment":"22:50:02", "Event":
  "POFF"} //Phone shutdown.
...}.
```

The output of detection is as follows:

```
{...
 {"Date": "2018-09-01", "Bedtime": "22:39:02", "Wake
  time": "6:29:59"},
 {"Date": "2018-09-02", "Bedtime": "23:12:45", "Wake
  time": "6:40:04"},
```

Figure 11: Processing flow of BTP.

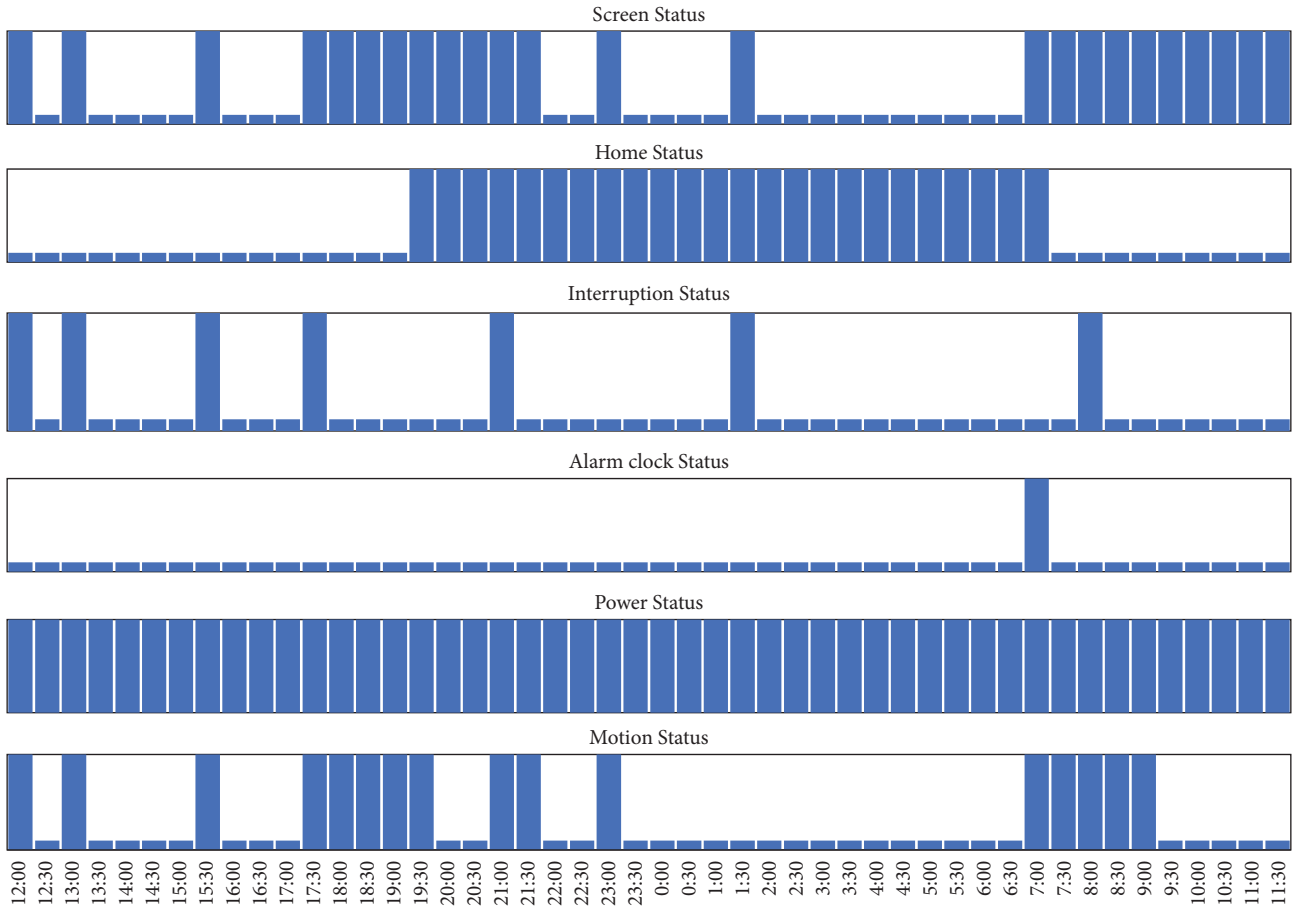


Figure 12: Whole day time-event sequences.

{"Date": "2018-09-03", "Bedtime": "22:59:12", "Wake time": "6:30:09"},
...}.

The human sleep pattern follows cycles of days generally. When discussing a single day, we must define the beginning of it. We set border point between two days at 12:00 instead of 0:00. Then we avoid interrupting continues sleeping that is also a continues screen extinguish period in our model.

For task two, prediction of wake time and bedtime, we also input log dataset as above. The output of prediction is as follows, if current time is "2018-09-10 18:00:00",

{"Date": "2018-09-10", "Bedtime": "22:40:15", "Wake time": "6:30:00"}.

Here wake time is referring to "2018-09-11 6:30:30" actually.

3.2. Data Collection and Preprocessing. Before running detection and prediction process, we must transform log data into several binary sequences. Considering that there are 1440 minutes in a day, we separate different events into six 1440-length sequences. For screen status, 0 represents screen extinguish and 1 represents screen light up. The sequences are shown in Table 2. For screen status and power status, we sampled the first second status of a minute. Yet for the other sequence, if there is any event occurred in the minute, the status will be 1.

Figure 12 demonstrates a timeline of a volunteer.

3.3. Method Process. BTP generally has two main phases, detection and prediction. Figure 13 shows the floatchart of detection.

In detection part, BTP follow four steps for each day to be detected.

Table 2: Sequence list.

Sequence	Coding Pattern	
	1/Event Code	0/Event Code
Screen Status	Screen light up/SON	Screen Extinguish/SOFF
Home Status	Arriving home/AH	Leave home/LH
Interruption Status	Incoming call/IC Receiving message/RM App notifications/AN	No interruption event
Alarm clock Status	Alarm clock ring/ACR	No ringing event
Power Status	Power on/PON	Power off/POFF
Motion Status	Accelerometer activated/AA	No activated

Table 3: Pseudocode of detection.

Series	Procedure	Description
1	for d_i in UndetectedDays = $\{d_1, \cdots, d_n\}$ do	
2	d_i.SSS = Filter(d_i.SSS)	Filter out screen status sequence that user is not at home
3	OffList = SelectLong (d_i.SSS)	Select continuous screen off sequences that last over 30 minutes
4	<sleep_time, wake_time> = JoinSeries(OffList)	Merge periods and get wake time and bedtime
5	SWTimeList.add(<sleep_time, wake_time>)	Record the results of consecutive days
6	end for	
7	end	

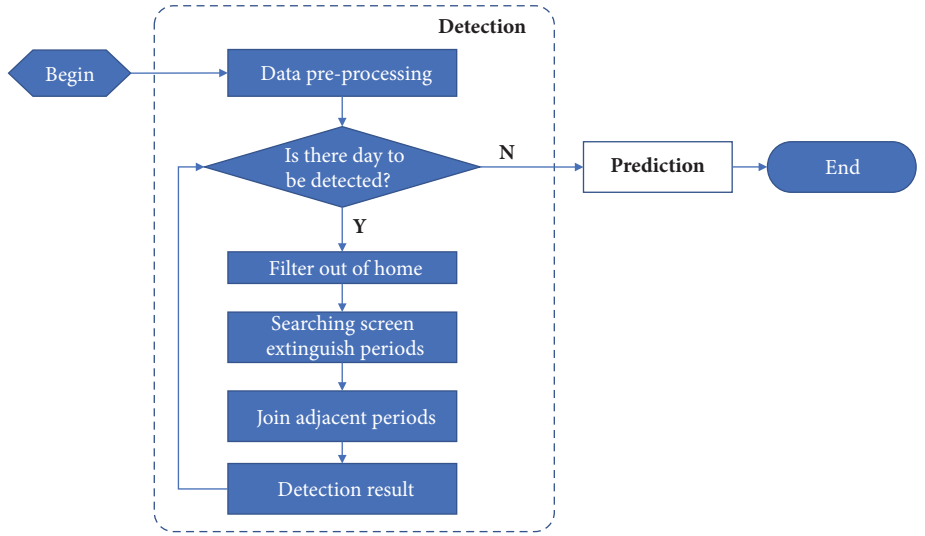


Figure 13: Flowchart of detection part.

(1) Filter out period of screen extinguish status when user is not home by searching home status sequence.

(2) Search screen status sequence and find out continues screen extinguish period which last over 30 minutes, then sort them in timeline.

(3) For each period selected by step 2, BTP uses (1) to determine whether this period can be united with its adjacent periods; that is, if time interval between them is shorter than T, we will conjunct the two adjacent sequences as one. In formula 2, T is a threshold that determine whether to join two sequences, l is used to adjust absolute value, t is minutes from 12:00 to middle of period, c is minutes from 12:00 to the time that people have deepest sleep, and off is an offset value.

$$T = l \times \frac{1}{1 + e^{|t-c|/60-off}} \quad (1)$$

(4) Adjust result with sequences of interruption status, alarm clock status, power status, and motion status.

Table 3 shows the pseudocode of detection.

After getting wake time and bedtime, as shown in Figure 14, predicting part has following steps:

Table 4: Pseudocode of prediction.

Series	Procedure	Description
1	for d_i in HistoryDays = $\{d_1, \cdots, d_n\}$ do	
2	SimList.add(CalSim(d, d_i))	Calculate similarities between today and history days
3	end for	
4	SelList = GetK(SimilarityList, k)	Select the K most similar sequences
5	Result = WA(SelList)	Calculate result Using weighted average
6	end	

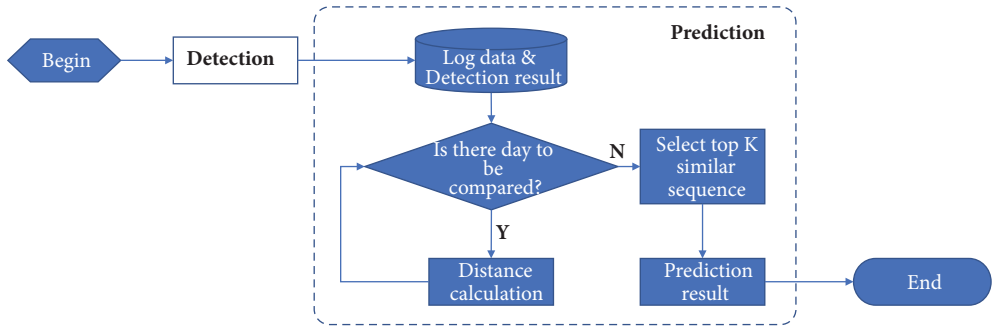


Figure 14: Flowchart of prediction part.

(1) Collect screen status sequence of today (from 0:00 to current time).

(2) Calculate similarities between today's screen sequence and each historical sequence using (2). Here, s is screen status sequence from 0:00 to current time, t is the sequence to be compared, and m is minutes from 0:00 to current time.

$$dist\,(s,t) = \sum_{i=1}^{m} |s_i - t_i| \tag{2}$$

(3) Select detection result according to the k most similar sequences and use (3) to generate result. *result* is prediction result, s_i is the i_{th} sequence of the k most similar sequences, and D_i is previous i_{th} detection result.

$$result = \sum_{i=1}^{k} \left[\frac{\sum_{j=1}^{k} \left(dist\left(s, s_j\right) - dist\left(s, s_i\right) \right)}{(\mathrm{k}-1) * \sum_{j=1}^{k} dist\left(s, s_j\right)} * D_i \right] \tag{3}$$

Table 4 shows the pseudocode of prediction.

4. Experimental Result

4.1. Experimental Background. The measurement criteria for detection is Mean Absolute Error (MAE), shown in (4). Here, n is number of people involved in this experiment, y_i is true value, and $f(x_i)$ is the detected value of methods to be tested.

$$MAE = \frac{\sum_{i=1}^{n} |y_i - f\,(x_i)|}{n} \tag{4}$$

The measurement criteria for prediction are RMSE, which are shown in (5). Here N is number of samples. x_i represents true value, and x_i' is the predicted value.

$$RMSE = \sqrt{\frac{\sum_{1}^{N} \left(x_i - x_i'\right)^2}{N}} \tag{5}$$

Our dataset contains 30 volunteers' data. They installed the same data acquisition application we distributed on their smartphones. The application has four missions: collecting log data, detecting both wake time and bedtime, uploading original data and result to server, and fetching feedback. Here we upload data for debug and sign agreement with volunteers. The client app is resident in android and collects log data of system day and night. At the end of a day, usually around 0:00, it starts our detection method to detect when the user wakes up and goes to bed. It also regularly displays questionnaires and returns feedback to server. These questionnaires only have a simple question: When did you fall asleep last night and wake up today? To ensure that the volunteers answer questions carefully, some feedback which is always confused with prediction value is filtered after checking their original log data. After completing 13 days' continuous test, 39 volunteers accomplished experiments. But nine of them were filtered out for careless answers. During the experiment, we ran the phase one of BTP to detect wake time and bedtime and ran BES, proposed by Chen et.al. [17], to detect sleep duration. Then, we ran the phase two of BTP, moving average (MA) [20], and exponential smoothing (ES) [20] to predict future wake time and bedtime on the left 30 samples with 13 days records and then calculate RMSE.

All experiments were performed on volunteers' Huawei P10 smartphones with EMUI 8.0 (Android 8.0), Kirin 960

Table 5: Detection of sleep duration.

Task	Sleep Duration Detection	
Algorithm	BTP	BES
MAE	25.84	48.19

Table 6: Detection result of BTP.

Criteria	Wake Time	Bedtime	Sleep Duration
RMSE	0.41	0.75	0.81
E_{max}	170	286	282
E_{min}	0	0	0
MAE	10.56	19.09	25.84

Table 7: MAE of different c ($off = 3$, $l = 17$).

c	Wake Time	Bedtime	Sleep Duration
840/2:00	12.99	21.86	29.57
900/3:00	10.56	19.09	25.84
960/4:00	14.61	19.32	27.13

(4×2.4GHz+4×1.8GHz), 4GB RAM, and 64GB ROM. We used Android Studio 2.3.3 with Android SDK 7.0 and Java 1.8 to develop the data acquisition application, which collects and uploads data to server regularly. The server for receiving data runs a CentOS 6.5 environment coupled with Intel(R) Xeon(R) CPU 4×2.5GHz, 64GB RAM, 1200GB HD@15000rpm. To compare BTP and other algorithms, they were both executed on a workstation of Think Station P300 with Windows10 64bit, Intel i7@4×3.6GHz, 16GB RAM@1600MHz, SATA3 hard disc 2TB@7200rpm.

4.2. Result of Detection

4.2.1. Comparison. The BES algorithm we implemented has a MAE of 48.19 rather than 42.5 in Chen's paper because of different datasets. The comparison result is shown in Table 5. BTP has a much lower MAE than BES on the same practical dataset.

Table 6 further presents RMSE, Maximum Absolute Error (E_{max}), Minimum Absolute Error (E_{min}), and MAE of the detection part of BTP for showing its reliability. Since alarm clock rings regularly in weekday, wake time is more accurate than bedtime.

4.2.2. Parameter Discussion. There are three parameters in Eq. (1), c, off, and l. We use control variate method to carry out parameter tuning by using MAE as evolution metric. Table 7 shows that the best c is around 900. That means 3:00 is really the central point of people's sleeping.

In Tables 8 and 9, we found that the best choice is $off = 3$ and $l = 17$. The two parameters together decide how far we should join two adjacent screens extinguish period. A too small length will bring irrelevant periods and too big length will also make sleep duration over splitting.

Table 8: MAE of different off ($c = 900$, $l = 17$).

off	Wake Time	Bedtime	Sleep Duration
2	12.86	18.86	26.98
2.5	10.56	19.09	25.84
3	10.23	18.78	24.79

Table 9: MAE of different l ($off = 3$, $c = 900$).

l	Wake Time	Bedtime	Sleep Duration
13	15.60	24.39	34.35
15	12.15	20.32	28.24
17	10.56	19.09	25.84
19	12.00	9.06	27.67
21	18.03	29.35	38.54

Table 10: Time complexities of BTP and BES.

Execute Times	BTP (s)	BES (s)
10	6.03	6.61
20	12.70	13.52
40	25.51	26.90
80	50.00	53.87
160	101.01	107.25

4.2.3. Time Complexity. From Table 10, we prove that BTP and BES are both following a linear complexity. They are fast so we repeated them to record their time consuming. BTP does not need regression as BES but must endure heavier I/O consuming.

4.3. Result of Prediction

4.3.1. Comparison. We compare BTP with Moving Average (MA) and Exponential Smoothing (ES). MA and ES are two classic algorithms for time series prediction. The result is shown in Table 11. BTP outperforms MA and ES in all metrics because wake time and bedtime have periodicity and the cycles vary from person to person.

4.3.2. Parameter Discussion. We applied control variate method to identify the influences brought by different parameters in m in (2) and k in (3). Table 12 illustrates that, with the incensement of k, the result is better. It proves that assembly learning takes advantage of robustness. But we only have records of 13 days; k is limited.

Table 13 shows that accuracy increases with longer compared sequence. Here, $m = 1080$; the 75% length of a day strikes balance between accuracy and usability.

D represents the number of historical days we use for prediction. If we have more historical samples, the model will be more robust. Here $D = 6$ may meet an overfitting condition. The result of experiments with different D are shown in Table 14.

4.3.3. Time Complexity. The time complexities of BTP, MA, and ES are $O(n)$. In practical, MA and ES would execute faster

Table 11: Prediction results.

Status	Wake Time			Bedtime		
Algorithm	BTP	Moving average	Exponential smoothing	BTP	Moving average	Exponential smoothing
RMSE	0.34	0.47	0.70	0.52	0.97	0.70
E_{max}	61.03	84.25	141.40	72.20	149.75	86.43
E_{min}	0.76	1.16	0.79	0.36	1.67	1.13
MAE	9.26	21.86	34.23	24.15	47.04	34.79

Table 12: MAE of different k ($m = 1080$, $D = 12$).

k	Wake Time	Bedtime
1	6.23	24.65
3	9.26	24.15
5	6.66	23.55
7	5.47	21.81

Table 13: MAE of different m with ($k = 3$, $D = 12$).

m	Wake Time	Bedtime
360	13.17	27.89
720	12.89	26.20
1080	9.26	24.15
1440	9.25	23.63

Table 14: MAE of different D ($k = 3$, $m = 1080$).

D	Wake Time	Bedtime
6	11.64	23.58
9	17.86	29.73
12	9.72	24.16

because the size of data is rather smaller. BTP uses historical screen status sequence for matching similar pattern. Each prediction requires traversing $1440 \times T$, and T represents number of compared days. However, MA and ES only use historical wake time and bedtime. This difference is negligible when running in a smartphone instead of centralized calculation in server.

4.4. Survey. To collect feedback of volunteers, we design a survey for them. The questionnaire only has two questions:

(1) Did you fell asleep in ["detected bedtime"] last night?

   (a) Exactly or deviation is under 15mins (get score 1)
   (b) Deviation is in 15-30mins (get score 0.5)
   (c) Deviation is more than 30mins (get score 0)

(2) Did you get up in ["detected wake time"] this morning?

   (a) Exactly or deviation is under 15mins (get score 1)
   (b) Deviation is in 15-30mins (get score 0.5)
   (c) Deviation is more than 30mins (get score 0)

Table 15: Results of the survey.

Answers	Wake Time	Bedtime
a	25	9
b	3	14
c	2	7
Total Score	26.5	16

Results of our survey are shown in Table 15. It is quite relevant to MAE of algorithms.

5. Conclusions

This paper presented a bedtime predicting algorithm named BTP. The goal of BTP is to precisely detect and predict wake time and bedtime of a person by mining screen status log of smartphone. The implement of BTP is independent of any actigraph or cloud infrastructure, which makes it fit for popular application and satisfy privacy protection. Experiments on 13 days screen log data of 30 persons indicated that BTP can effectively accomplish detection and prediction tasks without introducing extra devices or uploading data. We also compared it with BES, another sleep detection algorithm, and BTP exceed BES in accuracy and time consuming. Furthermore, our work shows that BTP could be applied as a critical context-aware component to improve intelligent service on smartphone.

Data Availability

Experiments in this paper involve practical datasets. The practical dataset belongs to certain enterprise and is limited to science research and protected by confidentiality agreement. But researchers can develop their own app to record people's behaviour to test their work.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work is supported by the Projects on International Scientific and Technological Cooperation under the National Key R&D Program (NKP) no. 2016YFE0204500, the National Natural Science Foundation of China under Grants no. 61671081 and no. 61720106007, the Beijing Natural Science Foundation under Grant no. 4172042, and the Fundamental

Research Funds for the Central Universities, Beijing University of Posts and Telecommunications 2017 Education and Teaching Reform Project no. 2017JY31.

References

[1] F. P. Cappuccio, F. M. Taggart, N.-B. Kandala et al., "Meta-analysis of short sleep duration and obesity in children and adults," *SLEEP*, vol. 31, no. 5, pp. 619–626, 2008.

[2] M. Kuppermann, D. P. Lubeck, P. D. Mazonson et al., "Sleep problems and their correlates in a working population," *Journal of General Internal Medicine*, vol. 10, no. 1, pp. 25–32, 1995.

[3] https://pzizz.com/.

[4] https://www.sleepcycle.com/.

[5] S. Seneviratne, A. Seneviratne, P. Mohapatra, and A. Mahanti, "Predicting user traits from a snapshot of apps installed on a smartphone," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 18, no. 2, pp. 1–8, 2014.

[6] A. Andersen and R. Karlsen, "User profiling through NFC interactions: Mining NFC-based user information from mobile devices and back-end systems," in *Proceedings of the 14th ACM International Symposium on Mobility Management and Wireless Access, MobiWac 2016*, pp. 173–176, Malta, November 2016.

[7] Y. Fan, Y. Chen, K. Tung, K. Wu, and A. L. Chen, "A framework for enabling user preference profiling through Wi-Fi logs," in *Proceedings of the 2016 IEEE 32nd International Conference on Data Engineering (ICDE)*, pp. 1550-1551, Helsinki, Finland, May 2016.

[8] R. Baeza-Yates, F. Silvestri, D. Jiang, and B. Harrison, "Predicting the next app that you are going to use," in *Proceedings of the 8th ACM International Conference on Web Search and Data Mining, WSDM 2015*, pp. 285–294, China, February 2015.

[9] C. A. Kushida, M. R. Littner, T. Morgenthaler et al., "Practice parameters for the indications for polysomnography and related procedures: an update for 2005," *SLEEP*, vol. 28, no. 4, pp. 499–521, 2005.

[10] N. J. Douglas, S. Thomas, and M. A. Jan, "Clinical value of polysomnography," *The Lancet*, vol. 339, no. 8789, pp. 347–350, 1992.

[11] C. P. Pollak, W. W. Tryon, H. Nagaraja, and R. Dzwonczyk, "How accurately does wrist actigraphy identify the states of sleep and wakefulness?" *SLEEP*, vol. 24, no. 8, pp. 957–965, 2001.

[12] D. J. Mullaney, D. F. Kripke, and S. Messin, "Wrist-actigraphic estimation of sleep time," *SLEEP*, vol. 3, no. 1, pp. 83–92, 1980.

[13] J. B. Webster, D. F. Kripke, S. Messin, D. J. Mullaney, and G. Wyborney, "An activity-based sleep monitor system for ambulatory use," *SLEEP*, vol. 5, no. 4, pp. 389–399, 1982.

[14] R. J. Cole, D. F. Kripke, W. Gruen, D. J. Mullaney, and J. C. Gillin, "Automatic sleep/wake identification from wrist activity," *SLEEP*, vol. 15, no. 5, pp. 461–469, 1992.

[15] G. Orellana, C. M. Held, P. A. Estevez et al., "A balanced sleep/wakefulness classification method based on actigraphic data in adolescents," in *Proceedings of the 2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBC 2014*, pp. 4188–4191, USA, August 2014.

[16] A. Domingues, T. Paiva, and J. M. Sanches, "Sleep and wakefulness state detection in nocturnal actigraphy based on movement information," *IEEE Transactions on Biomedical Engineering*, vol. 61, no. 2, pp. 426–434, 2014.

[17] G. Yang, J. Chen, H. Tenhunen, and L. Zheng, "Intelligent electrode design for long-term ECG monitoring at home: Prototype design using FPAA and FPGA," in *Proceedings of the 3d International ICST Conference on Pervasive Computing Technologies for Healthcare*, London, UK, August 2009.

[18] J.-K. Min, A. Doryab, J. Wiese, S. Amini, J. Zimmerman, and J. I. Hong, "Toss 'N' turn: Smartphone as sleep and sleep quality detector," in *Proceedings of the 32nd Annual ACM Conference on Human Factors in Computing Systems, CHI 2014*, pp. 477–486, Canada, May 2014.

[19] https://developers.google.cn/awareness/.

[20] V. Kotu and B. Deshpande, *Predictive Analytics and Data Mining*, 2015.

Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 8267838, 16 pages
https://doi.org/10.1155/2018/8267838

Review Article

A Survey on Mobile Edge Computing: Focusing on Service Adoption and Provision

Kai Peng,[1,2,3] **Victor C. M. Leung**,[2] **Xiaolong Xu**,[4] **Lixin Zheng**,[1,3] **Jiabin Wang**,[1,3] **and Qingjia Huang**[5]

[1]*College of Engineering, Huaqiao University, Quanzhou, Fujian 362021, China*
[2]*Department of Electrical and Computer Engineering, The University of British Columbia, Vancouver, BC, Canada V6T 1Z4*
[3]*Fujian Provincial Academic Engineering Research Centre in Industrial Intellectual Techniques and Systems, Quanzhou, Fujian 362021, China*
[4]*School of Computer and Software, Nanjing University of Information Science and Technology, Nanjing 210044, China*
[5]*Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China*

Correspondence should be addressed to Kai Peng; pkbupt@gmail.com

Received 18 April 2018; Revised 29 August 2018; Accepted 23 September 2018; Published 10 October 2018

Academic Editor: Pierre-Martin Tardif

Mobile cloud computing (MCC) integrates cloud computing (CC) into mobile networks, prolonging the battery life of the mobile users (MUs). However, this mode may cause significant execution delay. To address the delay issue, a new mode known as mobile edge computing (MEC) has been proposed. MEC provides computing and storage service for the edge of network, which enables MUs to execute applications efficiently and meet the delay requirements. In this paper, we present a comprehensive survey of the MEC research from the perspective of service adoption and provision. We first describe the overview of MEC, including the definition, architecture, and service of MEC. After that we review the existing MUs-oriented service adoption of MEC, i.e., offloading. More specifically, the study on offloading is divided into two key taxonomies: computation offloading and data offloading. In addition, each of them is further divided into single MU offloading scheme and multi-MU offloading scheme. Then we survey edge server- (ES-) oriented service provision, including technical indicators, ES placement, and resource allocation. In addition, other issues like applications on MEC and open issues are investigated. Finally, we conclude the paper.

1. Introduction

In recent years, with the continuous development of cloud computing (CC), big data, mobile network, software defined network (SDN), and upgrading of intelligent mobile terminals [1–11], the number of mobile users (MUs) has exploded rapidly. According to Cisco's latest forecast, global mobile data traffic (MDT) in 2020 will be 8 times of that in 2015, and the number of MUs will reach 2.63 billion [12]. However, compared to a traditional device such as PC, a MU has certain limitations in terms of computing power, storage capacity, and especially battery capacity, which greatly restricts the further promotion of the service. The emergence of mobile cloud computing (MCC) provides a good opportunity to relieve such limitations. MCC brings new kinds of services and facilities MUs to make the best use of CC [13].

However, cloud is usually located far away from the MUs, which leads to high network delay for data transmission between MUs and cloud. In order to solve the issue of network delay, a new paradigm named mobile edge computing (MEC) has been proposed. The MEC can be seen as a specific case of the MCC.

A number of surveys on MEC have been published recently [14–17]. A survey on MEC focusing on the general overview was presented in [14]. A comprehensive survey of the MEC research from the perspective of communication was provided in [15]. Wang et al. presented a comprehensive a survey on service migration in MEC in [16]. Two conceptions similar to service migration, such as live migration for data centers and handover in cellular networks, were investigated. Wang et al. [17] made an exhaustive review on the current research efforts on MEC. They firstly gave an overview

of MEC which included the definition, architecture, and advantages. Additionally, the issues on computing, caching, and communication techniques are also presented.

Different from the existing surveys, we provide a comprehensive survey of the state-of-the-art MEC research, focusing on service adoption and provision. More specifically, MEC is mainly composed of two parts, including MU and edge server (ES). On the one hand, we surveyed MUs-oriented service adoption, such as computation offloading and data offloading. On the other hand, ES-oriented service provision, including ES deployment and resource allocation, is investigated.

The summary of abbreviations in this paper is shown in Table 1. The rest of the paper is organized as follows. In Section 2, we give an overview of MEC. We describe MUs-oriented service adoption in Section 3. Followed by Section 4, ES-oriented service provision is mainly investigated. And then applications of MEC are introduced in Section 5. Section 6 elaborates open issues. Finally, we conclude the paper.

2. MEC Overview

In this section, we introduce the definition and architecture of MEC in Section 2.1, and then show the comparison of several modes in Section 2.2. A summary of literatures on different modes is shown in Table 2.

2.1. Definition, Architecture, and Service of MEC

2.1.1. Definition. MEC is a new network paradigm that provides information technology services and CC capabilities within the mobile access network of MUs and has become a technology. European Telecommunications Standards Institute (ETSI) proposed standardization of MEC in 2014 [18] and pointed out that MEC provided a new ecosystem and value chain that can use MEC to migrate intensive computing tasks of MUs to nearby ESs ([19, 20]). Since the MEC is located within the radio access network (RAN) and is close to MUs, it can achieve higher bandwidth with lower latency to improve quality of service (QoS) and quality of experience (QoE). MEC is also a key technology for the development of 5G [21], which helps to meet the high standards of 5G from delay, programmability, and scalability. By deploying services and caching at the edge of network, MEC can not only reduce congestion, but also efficiently respond to user requests. The most similar concept to MEC is EC. The EC refers to a new computing mode that performs computation at the edge of the network. The edge data in the EC represents the cloud service, and the upstream data represents the interconnection service. The edge of the EC refers to any computation and edge of the network from the data source to the CC center. The MEC emphasizes that the ES between the CC center and the edge device performs the task of computing the MU data on the ESs, but the MU does not basically have the computing capability. On the contrary, the MU in the EC model has strong computing capability. Therefore, the MEC can be seen as a part of the EC model [22]. In [23], Mach et al. firstly described the main use cases and reference scenarios for MEC. And they reviewed the existing notions of integrating MEC functionalities into mobile networks and discussed the progress of MEC standardization. They mainly reviewed user-oriented cases in the MEC, i.e., computation offloading. More specifically, the study of computation offloading was dived into three subissues: decision making, resources allocation, and mobility management. Liang [24] mainly focuses on MEC in 5G system and beyond.

In this paper, we mainly focus on the MEC and present a comprehensive survey from the perspective of service of MEC. More specially, we mainly review the service adoption of MU's and service provision of ES'. MUs in this paper can be mobile phones, tablets, and other intelligent devices. In fact, it is not very critical whether MUs have computing capability or not. The main issue of MUs is offloading to ESs. Meanwhile, the ES placement and resource allocation of ES is very important.

2.1.2. Architecture. As shown in Figure 1 [25], we argue that the MEC architecture can be seen as the middle layer between MUs and cloud which is closer to MUs and provide service for them and can communicate directly with cloud. In addition, it can also be seen as an independent two-tier architecture consisting of MUs and ESs. MU generally refers to mobile phones, tablets, laptops, and so on. In addition, ES generally refers to base station combined with server or cloudlet. MUs offload computing tasks to ES, which not only enables MUs to execute applications efficiently, but also reduces latency due to access to public cloud.

2.1.3. Service. In order to better understand of MEC, the authors in [26] begin with a discussion of the potential service scenarios and identified the design challenges of a MEC enabled network. MEC services mainly contain many aspects, such as service provision and service deployment. More specifically, as MUs and ESs are the core parts of MEC, and thus we mainly focus on MUs-oriented service adoption and ES-oriented service provision. For MUs, how to obtain service they need (i.e., offloading) to meet application requirements is more critical. For ESs, how to manage resources (i.e., ESs) effectively is critical.

2.2. Comparison of Several Modes. In this section, we introduce the existing surveys on conceptual comparison in Section 2.2.1. After that the similar terms are described in Section 2.2.2.

2.2.1. Surveys of Conceptual Comparison. Comparison of EC implementations: fog computing, cloudlet, and MEC are shown in [27]. In [28], the basic issues of the distribution and active caching computing tasks in fog environment are studied under the constraints of delay and reliability. We describe our comparison in Section 2.2.2.

2.2.2. Similar Terms. In this section, similar terms, such as mobile cloud computing (MCC), fog computing (FC), and cloudlets, are introduced and compared.

(1) Mobile Cloud Computing (MCC). The definition of MCC was shown in [29, 30]. The main idea is as follows. MCC refers

Table 1: Summary of abbreviations.

Access Point(s)	AP(s)
Base Stations	BSs
Cloud Computing	CC
Deep Supervised Learning	DSL
Dynamic Voltage Scaling	DVS
Edge Computing	EC
Edge Server(s)	ES(s)
Efficient Computation Offloading	ECO
End-to-End	E2E
Energy-Efficient Computation Offloading	EECO
European Telecommunications Standards Institute	ETSI
Fiber-Wireless	FiWi
Fog Computing	FC
Full Duplex	FD
Green-energy Aware Avatar Placement	GAP
Incentive-compatible Auction Mechanism	ICAM
Integer Linear Programming	ILP
Internet of Things	IoT
Intrusion Detection System	IDS
Iterative Improvement	Π
Markov Decision Process	MDP
Mixed Integer Linear Programming	MILP
Mobile Cloud Computing	MCC
Mobile Data Traffic	MDT
Mobile Data Traffic Offloading	MDTO
Mobile Edge Computing	MEC
Mobile Edge Internet of Things	MEIoT
Mobile Edge Computing-wireless Power Transfer	MEC-WPT
Multiple Knapsack Problem	MKP
Multi-objective Optimization Problem	MOOP
Multi-user Computation Offloading	MUCO
Multi-user Data Offloading	MUDO
Mobile User(s)	MU(s)
Physical Resource Block	PRB
Profit Maximization Avatar Placement	PRIMAL
Quality of Experience	QoE
Quality of Service	QoS
Radio Access Networks	RANs
Radio Access Points	RAPs
Software Defined Networking	SDN
Stochastic Reward Net	SRN
Two-Phase Optimization	TPO
Vehicular Ad Hoc Networks	VANET
Unmanned Aerial Vehicles	UAVs
Vehicular Cyber-physical Systems	VCPSs
Virtual Machine	VM
Wireless Access Point(s)	WAP(s)

Table 2: Summary of literatures on different modes.

Item	Related work	Key points
MEC	Definition [22–24]	[22] The difference between EC and MEC [23] The main use cases and reference scenarios standardization [24] MEC on 5G
	Architecture [25]	[25] MEC architecture
	Different network types [26]	[26] Potential service scenarios
Comparison of several modes	[27, 28]	[27] A set of parameters, decision tree [28] The basic problems of the distribution and active caching computing tasks
MCC	[29–35]	[29, 30] MCC definition [31] Review of MCC [32] Public cloud [33] Cloudlet [34] Ad hoc mobile cloud [35] MCC Integrating cloudlets and MEC
FC	[41, 42]	[41] The definition and similar concepts [42] A more comprehensive definition
Cloudlet	[46, 47]	[46] Cloudlet's computer cluster as a complement to computing offloading for cloud [47] Cloudlet can only be used for wireless access environments

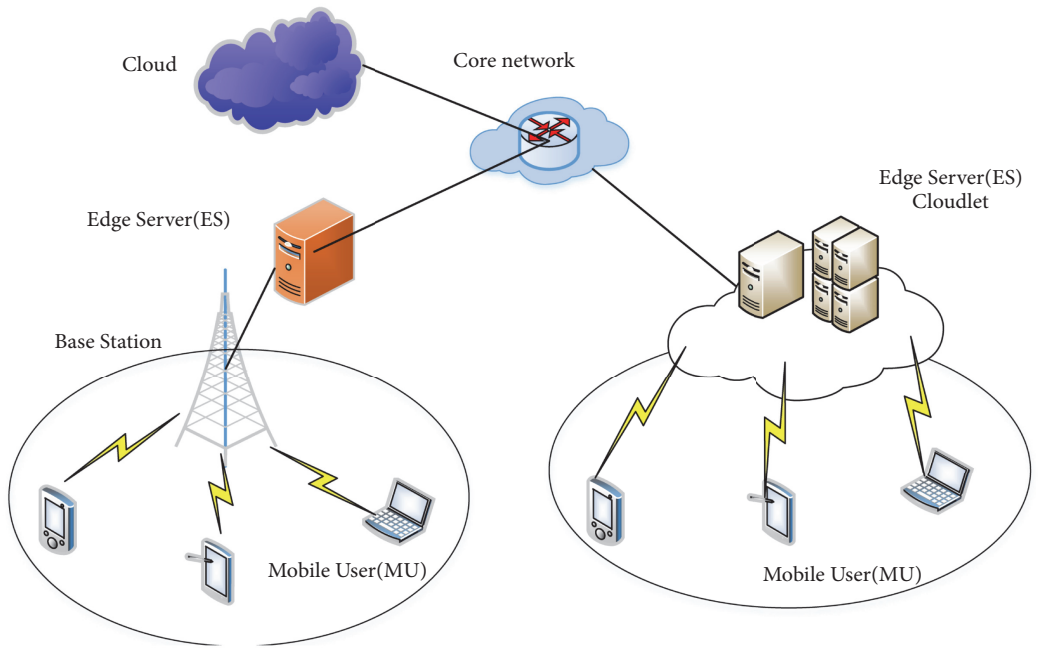


Figure 1: Mobile edge computing (MEC) architecture.

to the infrastructure that data storage and data processing happens outside of MU. Three kinds of MCC architectures are surveyed in [31], including the public cloud [32], the cloudlets [33], and the ad hoc mobile cloud [34]. The authors in [35] propose that the future of MCC will be integrated cloudlets and MEC. We think that the MCC and the MEC are different if the destination of the offloading in the MCC is cloud, but their concept is the same if the offloading destination is cloudlet.

As shown in Figure 2, a basic MCC system architecture is presented. MEC mainly consists of two parts, such as MUs and cloud servers. In general, it is assumed that the resource of cloud server is infinite and faraway from MU, while the MEC is closer to the MU. The MEC can reduce the MU's

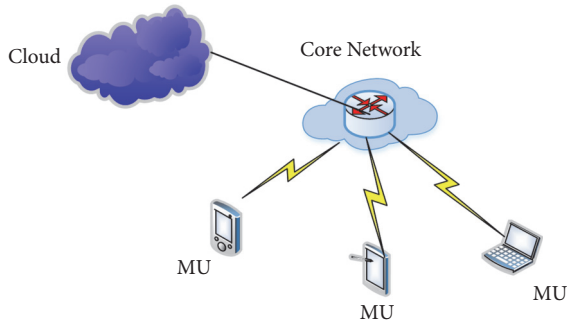


Figure 2: Mobile cloud computing (MCC) architecture.

access delay and improve QoS of MU. However, the ES of MEC has limited resource types and computing capabilities. When the request of MUs cannot be executed by ESs which needs to be further decided whether to execute the MU's request locally by MU itself or forward the request to the cloud.

(2) Fog Computing (FC). Although CC is widely used, due to the inherent problems (i.e., unreliable delay, lack of mobility support, and location awareness), there are still some problems needed to be figured out. FC can figure out the above issues by providing flexible resources and services for the MUs. FC was proposed by Cisco in 2012 [36] and was defined as a highly virtualized computing platform that previously performed tasks from CC centers to MUs. A comprehensive definition of FC was given by Vaquero in [37]. FC [38] extended the cloud-based network architecture by introducing an intermediate layer between cloud and MUs, and the middle layer is essentially a fog layer composed of fog servers deployed at the network edge [39]. The bandwidth load and energy consumption of the backbone link can be significantly reduced by fog servers [40]. In addition, the fog servers can be interconnected with cloud and use the computing power and rich applications and services of cloud. In [41], Yi et al. discussed the definition and similar concepts of FC, introduced representative application scenarios, and pointed out various problems that may be encountered when designing and implementing FC system. Yi et al. [42] discussed the current definitions of FC and similar concepts and proposed a more comprehensive definition. They also implement and evaluate a prototype FC platform. The authors in [22, 43] argue that MEC is substitutable with FC, and MEC mainly focuses more on the things side, whereas the latter one focuses more on the infrastructure side. In this paper, we think that there is no essential difference between MEC and FC. Both of them are closer to MUs and can be seen as an effective complement to CC and an important foundation for the realization of the Internet of Things (IoT).

(3) Cloudlets. Cloudlet was firstly proposed by authors in [33] and widely used in pervasive computing [44]. In the existing surveys, for one thing, this computing mode appeared relatively early. For another, the MUs are severed by a single cloudlet. And thus this mode is considered to be an independent service mode. Different from them, we hold the opinion that cloudlet can provide service for MUs by multiple cloudlet collaborations, which is the important way of service provision. In addition, the authors in [45, 46] also supported this view. In [47], the authors believe that cloudlet can only be used for wireless access environments. However, we believe that the MEC is a more general concept that can support Wi-Fi and is an effective supplement to 5G. Thereby, we hold the opinion that cloudlets play an important part in MEC as ESs for service provision. More specifically, cloudlets play an important role in WLAN and WMAN. More details will be described in Section 4.

3. Mobile Users- (MUs-) Oriented Service Adoption

In this section, we review the research on offloading. A summary of literatures on MUs-oriented service adoption are shown in Table 3, where key points in each paper are concluded. We firstly give a brief introduction of decision making on offloading.

In [48], Zhang et al. present a survey on decision making for task migrations which included decision factors and related algorithms. They point out that more attention should be paid to this issue. In [23], the authors review the decision making offloading issue of MEC in terms of local execution, full offloading, and partial offloading. A survey of computation offloading of mobile systems was proposed in [49]. They describe the two purposes (i.e., improving performance and saving energy) for offloading and provided a review from this perspective. Different from them, we mainly focus on the offloading issue from the perspective of offloading taxonomy. More specifically, we divide the studies on offloading into two key taxonomies: computation offloading and data offloading.

3.1. Offloading Taxonomy. We review computation offloading in Section 3.1.1 and data offloading in Section 3.1.2. The main concern of computation offloading and data offloading is different. Computation offloading refers to the MUs which send the heavy computation task to ESs and receive the results from them [49], while data offloading refers to use novel network techniques to transmit mobile data originally planned for transmission through cellular networks [50].

3.1.1. Computation Offloading. Computation offloading was originally studied in MCC. The offloading problem model in MEC is similar to MCC, but the main difference is the destination of offloading. It is commonly assumed that the implementation of computation offloading relies on a network architecture with the public cloud, while the offloading destination of MEC is ES. In addition, the offloading goals are basically the same which is to minimize total energy consumption or overall task execution time, or both of them. Furthermore, MEC computation offloading is divided into single mobile user computation offloading (single MUCO) and multiple mobile user computation offloading (Multi-MUCO).

(1) Single MUCO. There are many studies on single MUCO. The more representative ones are as follows ([51–58]). We

Table 3: Summary of literatures on MUs-oriented service adoption.

Work area	Related Work	Key Points
Single MUCO	Cloudlet based single MUCO [51–55] Base station based single MU CO [56–58]	[51] Application-aware cloudlet selection in multi-cloudlet environment [52] Cloudlet selection mechanism in a decentralized environment [53] A power and latency aware cloudlet selection in multi-cloudlet environment [54] MILP in the Multi-cloudlet environment [55] A deep learning approach [56] Offloading in task allocation and computational frequency scaling [57] Tradeoff between the latency and reliability in task offloading [58] Partial computation offloading using DVS
Multi-MUCO	Cloudlet based multi-MUCO [59] Base station based multi-MUCO [60–70]	[59] A Game-theoretic machine learning approach [60] MUCO game [61] Power-delay tradeoff in MU MEC systems [62] Multi-objective optimization [63] Sum energy consumption minimization [64] Joint computation offloading and interference management [65] Offloading in 5G heterogeneous networks [66] Joint offloading and computing optimization [67] Multiple knapsack problem for 5G mobile edge computing [68] Joint radio and computational resource management [69] Dynamic offloading approach [70] Joint task offloading and resource allocation
Single MUDO	[50, 71–73]	[71] An adaptive data offloading model. [72] Combination optimization in emerging VCPSs [50] Data offloading technologies (four taxonomies) [73] The UAV trajectory to offload traffic for BSs
Multi MUDO	[74]	[74] A joint coalition-pricing based on coalitional game theory and pricing mechanism

review the literature from the perspective of cloudlet based single MUCO and base station based single MUCO.

Cloudlet Based Single MUCO. To offload applications to the most appropriate cloudlet is very important. In [51], Roy et al. proposed an application-aware cloudlet selection strategy for multiple cloudlets scenario. They assumed that different cloudlets could deal with different types of applications. The application type was first verified when the request came from the MU. And then, the most suitable cloudlet is chosen from multiple cloudlets near the MU on the basis of the application type. By using the proposed strategy, both the energy consumption and the latency of application execution of the MU can be reduced. In [52], a mechanism to identify a cloudlet for computation offloading in a distributed manner was proposed. The mechanism mainly is composed of two phases. In the first phase, cloudlets within Wi-Fi range of the MU were identified. In the second one, the ideal offloading cloudlet was selected. Mukherjee et al. [53] proposed a power and latency aware optimum cloudlet selection method for multiple cloudlets scenario by introducing a proxy server. Theoretical analysis showed that the power and the latency consumption were reduced by about 29%-32% and 33%-36%, respectively, compared with offloading to cloud. A two-stage optimization strategy is proposed in [54]. Firstly, a cloudlet selection model based on mixed integer linear programming (MILP) is proposed to obtain the cloudlet for MUs by optimizing latency and the mean reward. Secondly, a resource allocation model based on MILP is presented to allocate the resources in the selected cloudlet by optimizing reward and the mean resource usage.

In [55], the authors develop a dynamic offloading framework for MUs, taking into account the local overhead of the MU, as well as the finite communication resource and computation resource of ES. They develop the offloading decision problem as a multilabel classification problem and a deep supervised learning (DSL) method is proposed to minimize the computation and offloading overhead. It is shown that system cost can be reduced compared to the existing four schemes, including no offloading scheme, random offloading scheme, total offloading scheme, and multilabel linear classifier based offloading scheme.

Base Station Based Single MUCO. An optimization framework of offloading from a single MU to multiple ESs was proposed in [56]. The authors considered two cases, such as fixed CPU frequency and elastic CPU frequency for the MU. In addition, two different methods were proposed to solve these two cases respectively. It was shown that the proposed algorithms achieved near optimal performance. In [57], the tradeoff between the latency and reliability in task offloading to MEC is studied. A framework is provided, where MU

partitions a task into subtask and offloads them to multiple ESs. In this framework, an optimization problem is formulated to jointly minimize the latency and offloading failure probability. Compared to the previous work, it is shown that the proposed algorithm achieves a good balance between these two objects, namely, latency and reliability. In [58], the authors focus on jointly optimizing communication resource and computation resource for partial computation offloading based on dynamic voltage scaling (DVS) technology. For one thing, the MU severed by a single ES is considered; for another, they investigate the energy consumption of minimization problem and latency of application execution minimization problem in a multiple ESs' scenario, where the MU could offload computation to a group of ESs.

(2) Multi-MUCO. Compared to single MU issue, multi-MUCO is more complex. Considering that resources are finite and MUs are mutually restricted, multi-MUCO is hard to solve. The more representative ones are as follows ([59–70]). We review the literature from the perspective of cloudlet based single MUCO and base station based single MUCO.

Cloudlet Based Multi-MUCO. Cao et al. [59] propose the problem of MUCO for cloudlet in MCC in multichannel wireless contention environment. They formulated this MUCO problem as a noncooperative game. And then a completely distributed computation offloading algorithm was developed so as to achieve the Nash Equilibrium Point (NEP). Finally, the effectiveness of their proposed algorithm was proved.

Base Station Based Multi-MUCO. In [60], the authors firstly studied the MUCO problem for MEC in a multichannel wireless interference environment. They showed that computing a centralized optimal solution is NP-hard, so a game theory was used to implement ECO in a distributed manner. The problem was modeled as MUCO game. They analyzed the structure of the game and showed that the game recognized Nash equilibrium. And then they designed a distributed computing offloading algorithm for the solution. Furthermore, the study is extended to MUCO scenario. It was demonstrated that their proposed algorithm could achieve higher performance. In [61], Mao et al. investigated the tradeoff between two key but contradictory goals in MU MEC systems, such as the power consumption of MUs and the execution delay of computation tasks. Based on the Lyapunov optimization method, an online algorithm for local execution and computation offloading is developed. The analysis showed that power consumption and execution delay followed the $[V\ (1/V), O\ (V)]$ (V is the control parameter). The simulation results confirmed the theoretical analysis. In [62], Liu et al. used queuing theory to study the energy consumption, execution delay, and price of offloading issue in MEC. A multiobjective optimization problem (MOOP) was formulated to minimize these three goals. The effectiveness of the proposed method by extensive simulations is demonstrated. In [63], the authors consider energy-efficient resource allocation for a multi-MU MEC system. The resource consumption for downloading the computation results back to MUs is taken into account. Meanwhile, the authors establish on two computation-efficient models with negligible and nonnegligible BS executing durations, respectively. Then, under each model, they develop a total weighting and energy consumption minimization problem by optimally allocating communication resources and computing resources. In [64], Wang et al. propose joint consideration of computation offloading and interference management so as to improve the performance of wireless cellular networks with MEC. In addition, the computation offloading decision, physical resource block (PRB) allocation, and MEC computation resource allocation are formulated as optimization problems. Simulation results show the effectiveness of the proposed scheme. In [65], Zhang et al. formulated an optimization issue to minimize energy consumption. Based on the multiple access characteristics of 5G heterogeneous networks, they have designed an energy-efficient computation offloading (EECO) method to minimize energy consumption under time delay constraints by integrating optimized offloading and radio resource allocation.

In [66], the authors propose a unified mobile edge computing-wireless power transfer (MEC-WPT) design by considering a wireless powered multi-MU MEC system, where a multiantenna access point (AP) (integrated with an MEC server) broadcasts wireless power to charge multi-MU and each MU relies on the harvested energy to execute computation tasks. They propose an optimal resource allocation scheme which minimizes the total energy consumption of AP's meeting the MUs' individual computation latency constraints. The authors [67] provide a general model of the system that takes account of the E2E computational latency of MEC applications. An evaluation of a multi-MU MEC offloading model reducible to the Multiple Knapsack Problem (MKP) is formulated. In [68], an online joint radio and computational resource management algorithm for multi-MU MEC systems is developed to minimize the long-term average weighted sum power consumption of the MUs and the ES, which are constrained by the stability of the task buffer. In [69], the authors focus on how to dynamically offload the computation intensive tasks of newly executed mobile applications from MUs to the ESs so as to minimize the average application completion time. They consider a system model in which a group of MUs are connected to the ES. Furthermore, they consider possible transmission collisions on a shared network when more than one MU try to transmit data simultaneously. It is shown that the proposed approach outperforms significantly the previous ones. In [70], a MEC enabled multicell wireless network is considered where each BS is equipped with an ES that can be used to assist the MU in performing computationally intensive tasks by offloading. The problem of joint task offloading and resource allocation is formulated to maximize the MUs' computation offloading profit, which is measured by reducing completion time and energy consumption of task.

3.1.2. Data Offloading (DO). Similarly, DO also can be divvied into single MU data offloading (Single MUDO) and multi-MU data offloading (Multi-MUDO).

(1) Single MUDO. Wang et al. proposed multiple effective data offloading methods and obtained better results for different

scenarios [71, 72]. In [71], they proposed a mobile data traffic offloading (MDTO) model which could determine whether to adopt opportunistic communications or communicate by cellular networks adaptively. In addition, Wang et al. [72] proposed a pioneering effort to promote MDTO in the emerging vehicular cyber-physical systems (VCPSs), aiming to reduce the MDT for the QoS aware services. They investigated MDTO models for Wi-Fi and Vehicular ad hoc networks (VANET). In particular, they formulated MDTO as a MOOP to simultaneously minimize MDT and QoS aware service provision. The optimal solutions were obtained by using mixed integer programming (MIP). The simulation results showed that the proposed scheme can offload MDT by up to 84.3% and meet the requirements of the global QoS guaranteed. Zhou et al. [50] discussed the current development of mobile data offloading techniques. In view of the diversity of data offloading sponsor, they divided the current mobile data offloading technologies into four taxonomies, thus, by small cell networks, by Wi-Fi networks, by opportunistic mobile networks, and by heterogeneous networks. In [73], the authors focused on the Unmanned Aerial Vehicles (UAV) trajectory at the edges to offload traffic for the base stations (BSs). An iterative algorithm was developed to solve the optimization issue which was a mixed integer nonconvex problem. The effectiveness of proposed scheme was shown.

(2) Multi-MUDO. The authors in [74] engaged in the issues of scheduling multiple MUs' offloading and choosing the ESs for offloading. They proposed a joint coalition-pricing based data offloading approach based on coalitional game theory and price mechanism. The numerical results showed the effectiveness of the proposed approach.

4. Edge Server- (ES-) Oriented Service Provision

In this section, we mainly review the ES-oriented service provision. In Section 4.1, the technical indicators are introduced, including expenditure in Section 4.1.1 and load balancing in Section 4.1.2. Followed by Section 4.2, ES placement for different scenarios (WLAN (Section 4.2.1) and WMAN (Section 4.2.2)) are reviewed. And then we summarize the resource allocation in Section 4.3, such as resource allocation in cloudlet based MEC system in Section 4.3.1; resource allocation in base station based MEC system in Section 4.3.2 is investigated. A summary of literatures on ES-oriented service provision is shown in Table 4.

4.1. Technical Indicators. In this section, expenditure is introduced in Section 4.1.1, and the load balancing is described in Section 4.1.2.

4.1.1. Expenditure. Different from cloud, the resources of ES are limited. It is critical to minimize the cost of ES and meet the requirements of user tasks. And therefore, the expenditure indictor is highlighted. In [75], an incentive-compatible auction mechanism (ICAM) was proposed for the resource trading between MUs and cloudlets. ICAM could effectively allocate cloudlets meeting MU's service needs and determine pricing. Based on Lyapunov optimization techniques combined with weighted perturbation techniques, Fang et al. [76] proposed a new random control algorithm that makes online decisions for computational request admission and scheduling, computational service purchase, and computational resource allocation. In particular, it was highly effective in practice. It was proved that the proposed algorithm achieved profit optimization and system stability. In [77], Sun et al. proposed a cloudlet network architecture. Each MU can communicate with its Avatar. Live Avatar migration was enabled to maintain the low E2E delay between each MU and its Avatar. They proposed a Profit Maximization Avatar placement (PRIMAL) policy to optimize the tradeoff between the migration profit and the migration cost. The effective of PRIMAL was demonstrated.

4.1.2. Load Balancing. Load balancing is an important indicator for evaluating the MEC system. Jia et al. [78] investigated how to balance the workload between multiple cloudlets. A fast and scalable algorithm was developed to solve the problem. Moreover, the performance was evaluated by experimental simulations which demonstrated the significant potential to reduce the response times of tasks. Xu et al. proposed [79] a dynamic resource allocation for load balancing in FC environment. In [80], the issue of load balancing in multi-MU of FC scenario was solved and the authors proposed a low-complexity algorithm for fog clustering.

4.2. ES Placement. In this section, we mainly review ES placement issue. In base station based MEC system, the ES are assumed to have been placed and are placed in the same location as the base station. So we mainly study the placement of edge servers (i.e., cloudlets) in the cloudlet based MEC system. We mainly review the case of cloudlet placement in VLAN and WLAN. We review the cloudlet placement in WLAN in Section 4.2.1 and the cloudlet placement in WMAN in Section 4.2.2.

4.2.1. Cloudlet Placement in WLAN. In [81], the authors presented a survey on the current mobile cloudlet architectures. They also classified the existing cloudlet solutions by presenting a hierarchical classification. The demands and challenges for deploying the cloudlet in a wireless local area networks (WLAN) were highlighted. In summary, the size of the MUs in the WLAN is relatively small, and the network coverage is relatively small. Therefore, the method cannot be directly used in WMAN. In the next section we will discuss the issue of cloudlet placement in WMAN.

4.2.2. Cloudlet Placement in WMAN. Figure 3 [82] illustrates a WMAN. It is assumed that there are K (K=3) cloudlets (cloudlet A, cloudlet B, and cloudlet C) to be placed to K different locations. For simplicity, it is assumed that the cloudlets will be colocated with some APs. Given the K placed cloudlets, MUs can offload the computation tasks to the cloudlets

Table 4: Summary of literatures on ES-oriented service provision.

Work Area	Related work	Key Points
Technical indicators	Expenditure [75–77] Load Balancing [78]	[75] An incentive-compatible auction mechanism [76] A new stochastic control algorithm [77] PRIMAL strategy [78] Balance the workload optimize mobile application performance
Cloudlet placement	WLAN [81]	[81] Cloudlet placement in WLAN
	WMAN [82–84, 86–88]	[83] The cache placement [84] The server placement [82] Minimize the average access delay [86] Cloudlet placement and mobile user allocation [87] A location-aware services deployment algorithm [88] AP ranking in the cloudlet placement
Resource scheduling	ES placement	[51] Offloading an application to the most appropriate cloudlet [52] Identify a cloudlet in a decentralized manner [53] Power and latency aware cloudlet selection
	Cloudlets deployments[46, 89, 90] Deploy the servers[91, 92]	[46] A MILP optimization model [89] Optimize the power consumption for large scale cloudlets deployment [90] TPO algorithm and an Π algorithm [91] Deploy the servers in a cost effective manner [92] A novel MCC architecture low latency and low energy consumption
	VM migration [99–106],	[99] A dedicated VM is provisioned on a PM [100] Combined performance and availability model using the Stochastic Reward Net [101] GAP minimize the total power consumption [102] Minimize services delay in a scenario with two cloudlet servers [103] A one-on-one contract game model and developed a learning-based price control mechanism [104] Dynamic proxy VM migration minimize the E2E delay [105] Dynamic services migration problem [106] A virtual FD-enabled small cell network with cache

via the local APs. If a cloudlet is colocated with an AP, the MUs at that AP will obtain the minimum cloudlet access delay of the MUs (i.e., MU (B) and MU(C) in Figure 3); otherwise, the MU requests at that AP must be relayed to nearby cloudlets for processing which leads to a cloudlet access delay due to the cumulative delay of multiple hop relays (i.e., MU (A) in Figure 3).

Cloudlets are particularly suited for wireless metropolitan area network (WMAN). And the cloudlet placement problem in WMAN consisting of many WAPs is very promising. There are two classical optimization problems which are closely related to this placement problem, namely, the cache placement [83] and the server placement [84]. Actually, both of these two issues can be solved by a direct reduction to the capacitated K-median problem [85]. However, this placement problem is essentially different from the above two problems; namely, it is assumed that either there is no capacity limitation on caches (or servers), or all caches (or servers) have identical capacities; however, the capacity of each cloudlet may be different and different MU requests may also have different computing resource needs. Xu et al. [82] firstly described the problem as a new capacity cloudlet placement problem.

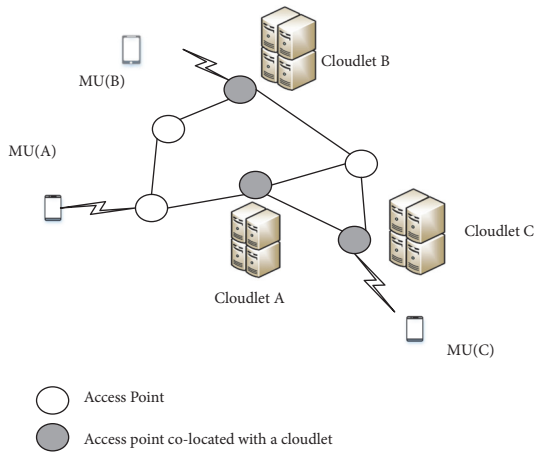


Figure 3: Cloudlet placements in WMAN.

And the object of this issue is to put K cloudlets in some strategic locations so as to minimize the average access delay between MUs and the cloudlets. In addition, they proposed an effective heuristic solution for this issue. It was

shown that the proposed algorithm had good scalability. In [86], Jia et al. formulated cloudlet placement and MUs allocation to the cloudlets issue in WMAN. They developed an algorithm, which enabled the placement of the cloudlets in WMAN and allocated MUs to the placed cloudlets while their workloads were balanced. They conducted experiments through simulation which demonstrate the effectiveness of the algorithm. As known to us, the MU in WMAN is moving, which made necessary to deploy and switch services anytime and anywhere to achieve the minimum network delay of MU services requests. However, the cost of this solution is usually too high for services providers and is invalid for resource utilization. A location-aware services deployment algorithm was proposed by Liang et al. [87] based on K-means to solve this problem. Generally speaking, the proposed algorithm divided the MUs into multiple MU clusters according to the geographic location of the MUs and then deployed the services instances to the ESs nearest to the centers of MU clusters. The performance evaluation showed that the algorithm could not only effectively lower the network latency, but also reduce the number of services instances while meeting the tolerable network latency. In [88], Yang et al. solved the problem of AP ranking in the cloudlet placement of the EC environment. AP ranking is an important step in the cloudlet placement. They proposed an adaptive integrated AP ordering method by analyzing the connection characteristics of APs. The results verified the effectiveness of their proposed approach. Above all, cloudlets deployment in WMAN has great challenges and great application prospects, which is an important part of MEC.

4.3. Resource Allocation. In this section, we divide the issue of resource scheduling into two subsections.

4.3.1. Resource Allocation in Cloudlet Based MEC System

Service Deployments. It is critical to dispatch of MU tasks through multiple cloudlet collaborations in MEC. In [89], Al-Ayyoub et al. solved the issue of optimizing the power consumption of large-scale collaborative cloudlets deployments. The effectiveness of the proposed mode was shown. In [46], Al-Quraan et al. presented a mixed integer linear programming (MILP) optimization model for MEC systems. More specifically, two kinds of cloudlets (i.e., local cloudlets and global cloudlets) are mentioned. A MU first sends the requirements to the local cloudlet. If none of local cloudlets can provide services, then it will be transferred to a global cloudlet. They were evaluated in several practical cases to prove that it can be applied for large-scale MEC systems. The purpose of [90] is to satisfy all computing requirements of each node in network edge within a certain delay based on limited computing resources (such as cells, APs, and macro-BSs), so as to optimize the total consumption. They formulated the issue as an Integer Programming problem and then developed a Two-Phase Optimization (TPO) algorithm and an Iterative Improvement (Π) algorithm for the solution. In [91], how to deploy the servers in an economical and efficient manner without violating the predetermined QoS was investigated by Yao et al. In particular, they practically considered that the available cloudlet servers are heterogeneous. That means the servers have different cost and resource capacities. The problem was formulated into an ILP form, and a low-complexity heuristic algorithm was proposed to address it. Extensive simulation studies validated the efficiency of the algorithm. Meng et al. [92] considered a novel MCC architecture composed of cloud server, cloudlet and MUs to ensure low latency and energy consumption. The joint optimization strategy was proposed to enhance the QoS. They formulated the wireless bandwidth and computing resource allocation model as a three stage Stackelberg game, and then used the backward method to address it. An iterative algorithm was used to achieve the Stackelberg equilibrium. The effectiveness of the method was demonstrated.

VM Migration. Some studies on virtual machine (VM) migration based on cloud have been well investigated ([31, 93–98]). However, these schemes did not take into account the relationship between MU mobility and VM migration, and thus they cannot be directly used in MEC. In [99], Raei et al. modeled a type of MCC known as the cloudlet where the MUs received services from a cloudlet as an intermediary node. A dedicated VM was provisioned on a physical machine (PM) while the PM could be located as a part of the cloudlet or a public cloud. In addition, they also proposed a combined performance and availability model based on the Stochastic Reward Net (SRN) in [100]. Sun et al. [101] proposed a Green-energy aware Avatar Placement (GAP) policy to minimize the total on-grid power consumption of the cloudlets by migrating Avatars among the cloudlets. It was shown that GAP can save 57.1% and 57.6% of on-grid power consumption. A method was proposed by Rodrigues et al. [102] for minimizing service delay in a scenario with two cloudlet servers. Differ from the previous researches, the method focused on both the computation element and communication element, controlling processing delay through VM migration. It was shown that the proposal presented the lowest service delay in all research cases. In [103], the VM migration problem was formulated as a one-on-one contract game model and a learning-based price control mechanism was developed to effectively deal with the MEC's resources. Finally, the extensive simulation results demonstrated the efficiency of the proposed approach. In [104], two dynamic proxy VM migration methods were proposed to minimize the E2E delay between proxy VMs and the IoT devices, as well as minimized the total energy consumption of the cloudlets. In [105], Wang et al. studied the dynamic service migration problem in MEC. They formulated a sequential decision making problem for service migration based on the framework of Markov Decision Process (MDP). A new algorithm and a numerical technique was developed for the solution. The effectiveness of the approach on a real-world dataset was shown. In [106], a virtual FD-enabled small cell network with cache and MEC was investigated for two heterogeneous services, namely, high-data-rate service and computation-sensitive service. Then they formulated a virtual resource allocation problem, because the original issue was a mixed combinatorial problem which was converted into a convex problem. In addition, the effectiveness of the proposed mode was verified by different system configurations.

Table 5: Summary of literatures on applications of MEC.

item	Related work	Key points
Applications	[104, 107–111, 113]	[104] Mobile edge Internet of Things (MEIoT)
		[107] Video streaming smart cities
		[108] A novel approach to MEC (edgeIoT)
		[109] Wearable devices and clouds in realistic setups
		[110] Special issue on the edge of the cloud
		[111] A UAV-based MCC system
		[113] OCR-like arbitrarily divisible applications

4.3.2. Resource Allocation in Base Station Based MEC System. Resource allocation and computation offloading are usually jointly considered in base station based MEC system. And thus we do not repeat introduce the literatures here. More information can be find in Section 3.

5. Applications on MEC

In this section, applications on MEC are reviewed. A summary of literatures on applications of MEC is shown in Table 5.

MEC not only remarkably reduces the cost of network operation and improving QoS of MUs by pushing computation resources closer to the network edges, but also provides a scalable IoT architecture for time-sensitive applications. In [104], Ansari et al. proposed a Mobile Edge Internet of Things (MEIoT) architecture which brought many resources (i.e., computing resource and storage resource) close to IoT devices. Taleb et al. [107] proposed an approach to enhance MUs' experience of video streaming in smart cities. The proposed approach relied on the MEC concept as a key factor to in improving QoS. It maintains QoS by guaranteeing services follow the MUs' mobility and implement the concept of "Follow Me Edge". This scheme provided an important solution to reduce core network traffic and ensure ultra-short delay. Sun et al. [108] proposed a novel approach to MEC named edgeIoT to figure out the data streams at the mobile edge. More specifically, each BS was connected to a fog node for providing computing resources locally. The SDN-based cellular core was designed to facilitate data forwarding among fog nodes. Wearable devices (i.e., smart watches, glasses, and helmets) were becoming more and more popular and were expected to become an indispensable part in our daily life. Despite the continuous upgrade of hardware, the life-time of MUs and functions (i.e., computing, storage) still need to be further improved. MCC can augment the capabilities of wearable devices by providing services. In [109], the authors presented a comprehensive analysis of computational offloading between wearable devices and cloud. The convergence of mobile computing and CC depends on reliable high bandwidth E2E network. These basic requirements are difficult to guarantee in harsh environments (i.e., military operations and disaster recovery). In [110], the authors examined how VM-based cloudlets that are located in close proximity to associated MUs can address these challenges. UAVs have been used to provide strengthened coverage or relay services for MUs in limited infrastructure wireless systems. In [111], a MCC system based on UAV was studied, in which mobile UAVs were given computing power to provider computation offloading opportunities to MUs. The system aimed to minimize the total energy cost of MU and meet the QoS requirements of offloading applications. Offloading was achieved through uplink and downlink communications between MU and the UAV. They formulated and solved a problem of the jointly optimizing of the bit allocation in uplink and downlink communications. The numerical results showed that a large amount of energy can be saved compared with local mobile execution. In addition, a novel EC empowered radio access network architecture was proposed by dong et al. [112], where the links of fronthaul and backhaul are mounted on the UAVs for fast event response and flexible deployment. Li et al. [113] figure out the problem on how to partition and allocate divisible applications to available resources in MEC environments to minimize the completion time of the applications. A theoretical model was developed for partitioning an OCR-like arbitrarily divisible application on the basis of the load of the application and the capabilities of available resources, and the solutions were derived in closed form.

6. Open Issue

In this section, the MEC challenges are introduced. In Section 6.1 open issues for MUs-oriented service adoption and ES-oriented service provision are introduced and then we describe other open issues, including Section 6.2.1 security issue and Section 6.2.2 simulation tools.

6.1. Open Issues for MU-Oriented Service Adoption and ES-Oriented Service Provision. MEC is a very novel and promising research area. Although we have introduced many studies in this paper, there still exist open research issues. From the MU's point of view, it is necessary to further design an efficient algorithm for multiple MUs to select a cloudlet that satisfies different service requirements. From the perspective of ES, MUs and cloudlet need to be jointly optimized for cloudlet placement issues. On the one hand, considering the limited resources of cloudlet, on the other hand, there are tremendous amounts of MUs in WMAN, so it is critical to study low-complexity cloudlet placement and scheduling algorithms. In addition, considering the dynamic changes of the MU's request and the energy consumption of cloudlet, multicloudlet collaboration method and VM

migration algorithm need to be further studied. Service recommendations [114] in MEC are also a promising research issue.

6.2. Other Issues

6.2.1. Security Issue. The main purpose of [115] is to holistically analyze the security threats, challenges, and mechanisms inherent in all edge paradigms. Wang et al. propose fog-based storage technology to fight with Cyber Threat [116]. Not only the traditional security issues, but also a number of new security issues in MEC also need to be concerned about. MEC consists of a large number of mobile devices, so it is also necessary to effectively protect MUs' privacy. Moreover, because the MEC can be seen as a network of multiple nodes, the assessment of node importance [117] and the assessment of invulnerability [118] of MEC also needs to be investigated. In our previous research [119], an Intrusion Detection System (IDS) was developed based on decision tree. Firstly, we developed a preprocessing algorithm to digitize the strings in the given dataset and then normalized the whole data. Secondly, we used decision tree method for our IDS system, and then we compared this method with other two methods, i.e., Naïve Bayesian and KNN. The effectiveness and precision of our proposed IDS system are shown. Above all, security and privacy have always been a fundamental research issue. We emphasize that security issues need to attract more attention.

6.2.2. Simulation Tools. To the best of our knowledge, many tools like Matlab, JAVA, and Python can be used for simulation for CC. In addition, CloudSim [120] is a well-known tool for cloud. In this section, we mainly review the tools for EC and MEC. More especially, we also introduce the tools for FC. A new simulator tool called EdgeCloudSim streamlined for EC scenarios is proposed in [121]. EdgeCloudSim is built on CloudSim to solve the specific needs of EC research and support necessary functionality in terms of computation and networking capabilities. Gupta et al. [122] proposed a simulator called iFogSim to model IoT and FC environments. They described two case studies to demonstrate modeling of an IoT environment and the comparison of resource management strategies. In addition, the scalability of the simulation toolkit of RAM consumption and runtime was verified under different scenarios. The authors in [123] discussed resource allocation in FC in the view of MUs' mobility and introduced MyiFogSim, an extension of iFogSim, which supported mobility through VM migration among cloudlets. In sum, the development of simulation tools is very promising, which can effectively promote the development of MEC and the standardization of experimental design.

7. Conclusion

With the development of mobile network and 5G, MEC has become a promising field in these years. It not only meets the user's more business needs, improves the QoS and QoE of MU, but also brings business benefits to service providers. In this paper, we present a comprehensive survey of MEC from the perspective of service adoption and provision. We firstly describe the overview of MEC. After that we review the existing MUs-oriented service adoption of MEC. And then we survey ES-oriented service provision. Moreover, other issues like applications on MEC, open issues are investigated. We highlight that more researches should focus on services of MEC.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work is supported by the Natural Science Foundation of Fujian Province under Grant no. 2018J05106, National Science Foundation of China under Grant no. 61702277, Quanzhou Science and Technology Project under Grant no. 2015Z115, and the Scientific Research Foundation of Huaqiao University under Grant no. 14BS316. China Scholarship Council (CSC) awarded Kai Peng one year's research abroad at the University of British Columbia. The authors would like to thank Tao Lin for collecting the material for writing.

References

[1] K. Peng, R. Lin, B. Huang, H. Zou, and F. Yang, "Link importance evaluation of data center network based on maximum flow," *Journal of Internet Technology*, vol. 18, no. 1, pp. 23–31, 2017.

[2] W. Quan, Y. Liu, H. Zhang, and S. Yu, "Enhancing crowd collaborations for software defined vehicular networks," *IEEE Communications Magazine*, vol. 55, no. 8, pp. 80–86, 2017.

[3] L. Qi, W. Dou, Y. Zhou, J. Yu, and C. Hu, "A context-aware service evaluation approach over big data for cloud applications," *IEEE Transactions on Cloud Computing*, vol. 1, pp. 1-1, 2015.

[4] Li. W, Y. Xia, M. Zhou, X. Sun, and Q. Zhu, "Fluctuation-aware and predictive workflow scheduling in cost-effective Infrastructure-as-a-Service clouds," *IEEE Access*, 2018.

[5] X. Xu, X. Zhao, F. Ruan et al., "Data placement for privacy-aware applications over big data in hybrid clouds," *Security and Communication Networks*, vol. 2017, pp. 1–15, 2017.

[6] L. Qi, X. Xu, X. Zhang et al., "Structural balance theory-based e-commerce recommendation over big rating data," *IEEE Transactions on Big Data*, 2016.

[7] K. Peng, H. Zou, R. Lin, and F. Yang, "Small business-oriented index construction of cloud data," in *Proceedings of the International Conference on Algorithms and Architectures for Parallel Processing*, pp. 156–165, Springer, Berlin, Germany, 2012.

[8] T. Wang, M. Z. A. Bhuiyan, G. Wang, M. A. Rahman, J. Wu, and J. Cao, "Big data reduction for a smart citys critical infrastructural health monitoring," *IEEE Communications Magazine*, vol. 56, no. 3, pp. 128–133, 2018.

[9] H. Xiang, X. Xu, H. Zheng et al., "An adaptive cloudlet placement method for mobile applications over GPS big data," in *Proceedings of the Global Communications Conference (GLOBECOM, 2016)*, pp. 1–6, IEEE, 2016.

[10] H. Zhou, V. C. M. Leung, C. Zhu, S. Xu, and J. Fan, "Predicting temporal social contact patterns for data forwarding in opportunistic mobile networks," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 11, pp. 10372–10383, 2017.

[11] K. Peng, V. C. M. Leung, and Q. Huang, "Clustering Approach Based on Mini Batch Kmeans for Intrusion Detection System over Big Data," *IEEE Access*, vol. 6, pp. 11897–11906, 2018.

[12] Cisco, Cisco visual networking index: global mobile data traffic forecast update, 2016-2021 white paper (EB/OL). (2017-03-01). http://10.3.200.202/cache/10/03/cisco.com/89e8529e7886890c-828d4a976994f806/mobile-white-paper-c11-520862.pdf.

[13] 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, pp. 1587–1611, 2013.

[14] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie, "Mobile Edge Computing: A Survey," *IEEE Internet of Things Journal*, vol. 5, no. 1, pp. 450–465, 2018.

[15] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A Survey on Mobile Edge Computing: The Communication Perspective," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2322–2358, 2017.

[16] S. Wang, J. Xu, N. Zhang, and Y. Liu, "A Survey on Service Migration in Mobile Edge Computing," *IEEE Access*, vol. 6, pp. 23511–23528, 2018.

[17] S. Wang, X. Zhang, Y. Zhang, L. Wang, J. Yang, and W. Wang, "A Survey on Mobile Edge Networks: Convergence of Computing, Caching and Communications," *IEEE Access*, vol. 5, pp. 6757–6779, 2017.

[18] European Telecommunications Standards Institute, Multi-access Edge Computing, http://www.etsi.org/technologies-clusters/technologies/multi-access-edge-computing.

[19] T. Zhao, S. Zhou, X. Guo, Y. Zhao, and Z. Niu, "A cooperative scheduling scheme of local cloud and internet cloud for delay-aware mobile cloud computing," in *Proceedings of the IEEE Globecom Workshops, GC Wkshps 2015*, USA, December 2015.

[20] R. R. Sarukkai and A. Mendhekar, Yahoo! Inc, (2004). Method and apparatus for accessing targeted, personalized voice/audio web content through wireless devices. U.S. Patent 6,728,731.

[21] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "Mobile edge computing—A key technology towards 5G," *ETSI white paper*, vol. 11, no. 11, pp. 1–16, 2015.

[22] W. S. Shi, H. Sun, and J. Cao, "Edge computing: an emerging computing model for the internet of everything era," *Journal of Computer Research and Development*, vol. 54, no. 5, pp. 907–924, 2017.

[23] P. Mach and Z. Becvar, "Mobile Edge Computing: A Survey on Architecture and Computation Offloading," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1628–1656, 2017.

[24] B. Liang, *Mobile edge computing*, Cambridge University Press, 2017.

[25] H. Tian, S.-S. Fan, X.-C. Lü, P.-T. Zhao, and S. He, "Mobile Edge Computing for 5G Requirements," *Beijing Youdian Daxue Xuebao/Journal of Beijing University of Posts and Telecommunications*, vol. 40, no. 2, pp. 1–10, 2017.

[26] B. P. Rimal, D. P. Van, and M. Maier, "Mobile Edge Computing Empowered Fiber-Wireless Access Networks in the 5G Era," *IEEE Communications Magazine*, vol. 55, no. 2, pp. 192–200, 2017.

[27] K. Dolui and S. K. Datta, "Comparison of edge computing implementations: Fog computing, cloudlet and mobile edge computing," in *Proceedings of the 2017 Global Internet of Things Summit, GIoTS 2017*, pp. 1–6, Geneva, Switzerland, June 2017.

[28] M. S. Elbamby, M. Bennis, and W. Saad, "Proactive edge computing in latency-constrained fog networks," in *Proceedings of the 2017 European Conference on Networks and Communications, EuCNC 2017*, Finland, June 2017.

[29] M. Ali, "Green cloud on the horizon," in *Proceedings of the IEEE International Conference on Cloud Computing*, pp. 451–459, Berlin, Germany, December 2009.

[30] http://www.mobilecloudcomputingforum.com/.

[31] F. Liu, P. Shu, H. Jin et al., "Gearing resource-poor mobile devices with powerful clouds: architectures, challenges, and applications," *IEEE Wireless Communications Magazine*, vol. 20, no. 3, pp. 14–21, 2013.

[32] Z. Liu, Y. Feng, and B. Li, "Socialize spontaneously with mobile applications," in *Proceedings of the IEEE Conference on Computer Communications, INFOCOM 2012*, pp. 1942–1950, USA, March 2012.

[33] M. Satyanarayanan, P. Bahl, R. Cáceres, and N. Davies, "The case for VM-based cloudlets in mobile computing," *IEEE Pervasive Computing*, vol. 8, no. 4, pp. 14–23, 2009.

[34] 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.

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

[36] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," in *Proceedings of the 1st ACM Mobile Cloud Computing Workshop, MCC 2012*, pp. 13–16, Finland, August 2012.

[37] L. M. Vaquero and L. Rodero-Merino, "Finding your way in the fog: Towards a comprehensive definition of fog computing," *Computer Communication Review*, vol. 44, no. 5, pp. 27–32, 2014.

[38] T. H. Luan, L. Gao, Z. Li et al., *Fog computing: Focusing on mobile users at the edge*, https://arxiv.org/abs/1502.01815.

[39] G. I. Klas, Fog computing and mobile edge cloud gain momentum open fog consortium, etsi mec and cloudlets. Google Scholar, 2015.

[40] Blue an exclusive wifi enabled onboard entertainment system https://www.greyhound.com/en/about/media/2013/07-08-2013.

[41] S. Yi, C. Li, and Q. Li, "A survey of fog computing: concepts, applications and issues," in *Proceedings of the Workshop on Mobile Big Data (Mobidata '15)*, pp. 37–42, ACM, Hangzhou, China, June 2015.

[42] S. Yi, Z. Hao, Z. Qin, and Q. Li, "Fog computing: Platform and applications," in *Proceedings of the 3rd Workshop on Hot Topics in Web Systems and Technologies, HotWeb 2015*, pp. 73–78, USA, November 2015.

[43] M. R. Anawar, S. Wang, M. Azam Zia et al., "Fog Computing: An Overview of Big IoT Data Analytics," *Wireless Communications and Mobile Computing*, vol. 2018, Article ID 7157192, 22 pages, 2018.

[44] S. Clinch, J. Harkes, A. Friday, N. Davies, and M. Satyanarayanan, "How close is close enough? Understanding the role of cloudlets in supporting display appropriation by mobile users," in *Proceedings of the 10th IEEE International Conference on Pervasive Computing and Communications, PerCom 2012*, pp. 122–127, Switzerland, March 2012.

[45] Y. Jararweh, M. Al-Ayyoub, M. Al-Quraan, L. A. Tawalbeh, and E. Benkhelifa, "Delay-aware power optimization model for mobile edge computing systems," *Personal and Ubiquitous Computing*, vol. 21, no. 6, pp. 1067–1077, 2017.

[46] M. Al-Quraan, M. Al-Ayyoub, Y. Jararweh, L. Tawalbeh, and E. Benkhelifa, "Power optimization of large scale mobile cloud system using cooperative cloudlets," in *Proceedings of the 4th International Conference on Future Internet of Things and Cloud Workshops, W-FiCloud 2016*, pp. 34–38, Austria, August 2016.

[47] P. Hu, S. Dhelim, H. Ning, and T. Qiu, "Survey on fog computing: architecture, key technologies, applications and open issues," *Journal of Network and Computer Applications*, vol. 98, pp. 27–42, 2017.

[48] W. Zhang, S. Tan, F. Xia et al., "A survey on decision making for task migration in mobile cloud environments," *Personal and Ubiquitous Computing*, vol. 20, no. 3, pp. 295–309, 2016.

[49] K. Kumar, J. Liu, Y.-H. Lu, and B. Bhargava, "A survey of computation offloading for mobile systems," *Mobile Networks and Applications*, vol. 18, no. 1, pp. 129–140, 2013.

[50] H. Zhou, H. Wang, X. Li, and V. C. M. Leung, "A Survey on Mobile Data Offloading Technologies," *IEEE Access*, vol. 6, pp. 5101–5111, 2018.

[51] D. G. Roy, D. De, A. Mukherjee, and R. Buyya, "Application-aware cloudlet selection for computation offloading in multi-cloudlet environment," *The Journal of Supercomputing*, vol. 73, no. 4, pp. 1672–1690, 2017.

[52] D. Parmar, A. S. Kumar, A. Nivangune, P. Joshi, and U. P. Rao, "Discovery and selection mechanism of cloudlets in a decentralized MCC environment," in *Proceedings of the IEEE/ACM International Conference on Mobile Software Engineering and Systems, MobileSoft 2016*, pp. 15-16, USA, May 2016.

[53] A. Mukherjee, D. De, and D. Roy, "A Power and Latency Aware Cloudlet Selection Strategy for Multi-Cloudlet Environment," *IEEE Transactions on Cloud Computing*, 2016.

[54] L. Liu and Q. Fan, "Resource Allocation Optimization Based on Mixed Integer Linear Programming in the Multi-Cloudlet Environment," *IEEE Access*, vol. 6, pp. 24533–24542, 2018.

[55] S. Yu, X. Wang, and R. Langar, "Computation offloading for mobile edge computing: a deep learning approach," in *Proceedings of the Personal, Indoor, and Mobile Radio Communications (PIMRC, 2017)*, pp. 1–6, IEEE, 2017.

[56] T. Q. Dinh, J. Tang, Q. D. La, and T. Q. S. Quek, "Offloading in Mobile Edge Computing: Task Allocation and Computational Frequency Scaling," *IEEE Transactions on Communications*, vol. 65, no. 8, pp. 3571–3584, 2017.

[57] J. Liu and Q. Zhang, "Offloading Schemes in Mobile Edge Computing for Ultra-Reliable Low Latency Communications," *IEEE Access*, vol. 6, pp. 12825–12837, 2018.

[58] Y. Wang, M. Sheng, X. Wang, L. Wang, and J. Li, "Mobile-Edge Computing: Partial Computation Offloading Using Dynamic Voltage Scaling," *IEEE Transactions on Communications*, vol. 64, no. 10, pp. 4268–4282, 2016.

[59] H. Cao and J. Cai, "Distributed multiuser computation offloading for cloudlet-based mobile cloud computing: A game-theoretic machine learning approach," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 1, pp. 752–764, 2018.

[60] 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.

[61] Y. Mao, J. Zhang, S. H. Song, and K. B. Letaief, "Power-delay tradeoff in multi-user mobile-edge computing systems," in *Proceedings of the In Global Communications Conference (GLOBECOM, 2016)*, pp. 1–6, 2016.

[62] L. Liu, Z. Chang, X. Guo, and T. Ristaniemi, "Multi-objective optimization for computation offloading in mobile-edge computing," in *Proceedings of the 2017 IEEE Symposium on Computers and Communications, ISCC 2017*, pp. 832–837, Greece, July 2017.

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

[64] C. Wang, F. R. Yu, C. Liang, Q. Chen, and L. Tang, "Joint Computation Offloading and Interference Management in Wireless Cellular Networks with Mobile Edge Computing," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 8, pp. 7432–7445, 2017.

[65] K. Zhang, Y. Mao, S. Leng et al., "Energy-Efficient Offloading for Mobile Edge Computing in 5G Heterogeneous Networks," *IEEE Access*, vol. 4, pp. 5896–5907, 2016.

[66] F. Wang, J. Xu, X. Wang, and S. Cui, "Joint Offloading and Computing Optimization in Wireless Powered Mobile-Edge Computing Systems," *IEEE Transactions on Wireless Communications*, vol. 17, no. 3, pp. 1784–1797, 2018.

[67] I. Ketykó, L. Kecskés, C. Nemes et al., "Multi-user computation offloading as multiple knapsack problem for 5G mobile edge computing," in *Proceedings of the Networks and Communications (EuCNC), 2016 European Conference*, pp. 225–229, 2016.

[68] Y. Mao, J. Zhang, S. H. Song, and K. B. Letaief, "Stochastic Joint Radio and Computational Resource Management for Multi-User Mobile-Edge Computing Systems," *IEEE Transactions on Wireless Communications*, vol. 16, no. 9, pp. 5994–6009, 2017.

[69] K. Guo, M. Yang, Y. Zhang, and Y. Ji, "An Efficient Dynamic Offloading Approach based on Optimization Technique for Mobile Edge Computing," in *Proceedings of the Mobile Cloud Computing, Services, and Engineering (MobileCloud, 2018)*, pp. 29–36, 2018.

[70] T. X. Tran and D. Pompili, *Joint task offloading and resource allocation for multi-server mobile-edge computing networks*, https://arxiv.org/abs/1705.00704.

[71] T. Lei, S. Wang, J. Li, and F. Yang, "AOM: adaptive mobile data traffic offloading for M2M networks," *Personal and Ubiquitous Computing*, vol. 20, no. 6, pp. 863–873, 2016.

[72] S. Wang, T. Lei, L. Zhang, C.-H. Hsu, and F. Yang, "Offloading mobile data traffic for QoS-aware service provision in vehicular cyber-physical systems," *Future Generation Computer Systems*, vol. 61, pp. 118–127, 2016.

[73] F. Cheng, S. Zhang, Z. Li et al., "UAV Trajectory Optimization for Data Offloading at the Edge of Multiple Cells," *IEEE Transactions on Vehicular Technology*, 2018.

[74] T. Zhang, W. Chen, and F. Yang, *Data offloading in mobile edge computing: A coalitional game based pricing approach*, 2017, https://arxiv.org/abs/1709.04148.

[75] A.-L. Jin, W. Song, and W. Zhuang, "Auction-Based Resource Allocation for Sharing Cloudlets in Mobile Cloud Computing," *IEEE Transactions on Emerging Topics in Computing*, vol. 6, no. 1, pp. 45–57, 2018.

[76] W. Fang, X. Yao, X. Zhao, J. Yin, and N. Xiong, "A Stochastic Control Approach to Maximize Profit on Service Provisioning for Mobile Cloudlet Platforms," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 48, no. 4, pp. 522–534, 2018.

[77] X. Sun and N. Ansari, "PRIMAL: PRofIt Maximization Avatar pLacement for mobile edge computing," in *Proceedings of the 2016 IEEE International Conference on Communications, ICC 2016*, Malaysia, May 2016.

[78] M. Jia, W. Liang, Z. Xu, and M. Huang, "Cloudlet load balancing in wireless metropolitan area networks," in *Proceedings of the INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications*, pp. 1–9, San Francisco, CA, USA, April 2016.

[79] X. Xu, S. Fu, Q. Cai et al., "Dynamic Resource Allocation for Load Balancing in Fog Environment," *Wireless Communications and Mobile Computing*, vol. 2018, Article ID 6421607, 15 pages, 2018.

[80] J. Oueis, E. C. Strinati, and S. Barbarossa, "The fog balancing: Load distribution for small cell cloud computing," in *Proceedings of the 81st IEEE Vehicular Technology Conference, VTC Spring 2015*, UK, May 2015.

[81] U. Shaukat, E. Ahmed, Z. Anwar, and F. Xia, "Cloudlet deployment in local wireless networks: Motivation, architectures, applications, and open challenges," *Journal of Network and Computer Applications*, vol. 62, pp. 18–40, 2016.

[82] Z. Xu, W. Liang, W. Xu, M. Jia, and S. Guo, "Efficient Algorithms for Capacitated Cloudlet Placements," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 10, pp. 2866–2880, 2016.

[83] L. Qiu, V. N. Padmanabhan, and G. M. Voelker, "On the placement of web server replicas," in *Proceedings of the INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 3, pp. 1587–1596, 2001.

[84] H. Yin, X. Zhang, T. Zhan, Y. Zhang, G. Min, and D. O. Wu, "NetClust: A framework for scalable and pareto-optimal media server placement," *IEEE Transactions on Multimedia*, vol. 15, no. 8, pp. 2114–2124, 2013.

[85] M. Charikar, S. Guha, E. Tardos, and D. B. Shmoys, "Constant-factor approximation algorithm for the k-median problem," in *Proceedings of the 1999 31st Annual ACM Symposium on Theory of Computing - FCRC '99*, pp. 1–10, May 1999.

[86] M. Jia, J. Cao, and W. Liang, "Optimal cloudlet placement and user to cloudlet allocation in wireless metropolitan area networks," *IEEE Transactions on Cloud Computing*, 2015.

[87] T.-Y. Liang and Y.-J. Li, "A location-aware service deployment algorithm based on k-means for cloudlets," *Mobile Information Systems*, vol. 2017, Article ID 8342859, 10 pages, 2017.

[88] G. Yang, Q. Sun, A. Zhou, S. Wang, and J. Li, "Poster abstract: Access point ranking for cloudlet placement in edge computing environment," in *Proceedings of the 1st IEEE/ACM Symposium on Edge Computing, SEC 2016*, pp. 85-86, USA, October 2016.

[89] M. Al-Ayyoub, Y. Jararweh, L. Tawalbeh, E. Benkhelifa, and A. Basalamah, "Power Optimization of Large Scale Mobile Cloud Computing Systems," in *Proceedings of the 3rd International Conference on Future Internet of Things and Cloud, FiCloud 2015*, pp. 670–674, Italy, August 2015.

[90] W. Wang and W. Zhou, "Computational offloading with delay and capacity constraints in mobile edge," in *Proceedings of the 2017 IEEE International Conference on Communications, ICC 2017*, France, May 2017.

[91] H. Yao, C. Bai, M. Xiong, D. Zeng, and Z. Fu, "Heterogeneous cloudlet deployment and user-cloudlet association toward cost effective fog computing," *Concurrency Computation*, vol. 29, no. 16, 2017.

[92] S. Meng, Y. Wang, Z. Miao, and K. Sun, "Joint optimization of wireless bandwidth and computing resource in cloudlet-based mobile cloud computing environment," *Peer-to-Peer Networking and Applications*, vol. 11, no. 3, pp. 462–472, 2018.

[93] B.-G. Chun, S. Ihm, P. Maniatis, M. Naik, and A. Patti, "CloneCloud: elastic execution between mobile device and cloud," in *Proceedings of the 6th ACM EuroSys Conference on Computer Systems (EuroSys '11)*, pp. 301–314, ACM, April 2011.

[94] F. Liu, P. Shu, and J. C. S. Lui, "AppATP: An Energy Conserving Adaptive Mobile-Cloud Transmission Protocol," *IEEE Transactions on Computers*, vol. 64, no. 11, pp. 3051–3063, 2015.

[95] F. Xu, F. Liu, L. Liu, H. Jin, B. Li, and B. Li, "iAware: making live migration of virtual machines interference-aware in the cloud," *Institute of Electrical and Electronics Engineers. Transactions on Computers*, vol. 63, no. 12, pp. 3012–3025, 2014.

[96] F. Xu, F. Liu, H. Jin, and A. V. Vasilakos, "Managing performance overhead of virtual machines in cloud computing: A survey, state of the art, and future directions," *Proceedings of the IEEE*, vol. 102, no. 1, pp. 11–31, 2014.

[97] B. Huang, R. Lin, K. Peng, H. Zou, and F. Yang, "Minimizing latency in fetching virtual machine images based on multi-point collaborative approach," in *Proceedings of the 2013 IEEE International Conference on Green Computing and Communications and IEEE Internet of Things and IEEE Cyber, Physical and Social Computing, GreenCom-iThings-CPSCom 2013*, pp. 262–267, China, August 2013.

[98] X. Xu, X. Zhang, M. Khan, W. Dou, S. Xue, and S. Yu, "A balanced virtual machine scheduling method for energy-performance trade-offs in cyber-physical cloud systems," *Future Generation Computer Systems*, 2017.

[99] H. Raei, N. Yazdani, and R. Shojaee, "Modeling and performance analysis of cloudlet in Mobile Cloud Computing," *Performance Evaluation*, vol. 107, pp. 34–53, 2017.

[100] H. Raei and N. Yazdani, "Performability analysis of cloudlet in mobile cloud computing," *Information Sciences*, vol. 388-389, pp. 99–117, 2017.

[101] X. Sun and N. Ansari, "Green Cloudlet Network: A Sustainable Platform for Mobile Cloud Computing," *IEEE Transactions on Cloud Computing*, 2017.

[102] T. G. Rodrigues, K. Suto, H. Nishiyama, and N. Kato, "Hybrid method for minimizing service delay in edge cloud computing through VM migration and transmission power control," *Institute of Electrical and Electronics Engineers. Transactions on Computers*, vol. 66, no. 5, pp. 810–819, 2017.

[103] S. Kim, "One-on-one contract game–based dynamic virtual machine migration scheme for Mobile Edge Computing," *Transactions on Emerging Telecommunications Technologies*, vol. 29, no. 1, 2018.

[104] N. Ansari and X. Sun, "Mobile edge computing empowers internet of things," *IEICE Transactions on Communications*, vol. E101B, no. 3, pp. 604–619, 2018.

[105] S. Wang, R. Urgaonkar, M. Zafer, T. He, K. Chan, and K. K. Leung, "Dynamic service migration in mobile edge-clouds," in *Proceedings of the 2015 14th IFIP Networking Conference, IFIP Networking 2015*, France, May 2015.

[106] Z. Tan, F. R. Yu, and X. Li, "Heterogeneous Services Provisioning in Small Cell Networks with Cache and Mobile Edge Computing," https://arxiv.org/abs/1706.09542.

[107] T. Taleb, S. Dutta, A. Ksentini, M. Iqbal, and H. Flinck, "Mobile edge computing potential in making cities smarter," *IEEE Communications Magazine*, vol. 55, no. 3, pp. 38–43, 2017.

[108] X. Sun and N. Ansari, "EdgeIoT: Mobile Edge Computing for the Internet of Things," *IEEE Communications Magazine*, vol. 54, no. 12, pp. 22–29, 2016.

[109] C. Ragona, F. Granelli, C. Fiandrino, D. Kliazovich, and P. Bouvry, "Energy-efficient computation offloading for wearable

devices and smartphones in mobile cloud computing," in *Proceedings of the 58th IEEE Global Communications Conference, GLOBECOM 2015*, December 2015.

[110] M. Satyanarayanan, G. Lewis, E. Morris, S. Simanta, J. Boleng, and K. Ha, "The role of cloudlets in hostile environments," *IEEE Pervasive Computing*, vol. 12, no. 4, pp. 40–49, 2013.

[111] S. Jeong, O. Simeone, and J. Kang, "Mobile Edge Computing via a UAV-Mounted Cloudlet: Optimization of Bit Allocation and Path Planning," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 3, pp. 2049–2063, 2018.

[112] Y. Dong, M. Hassan, J. Cheng et al., *An edge computing empowered radio access network with UAV-mounted FSO fronthaul and backhaul: Key challenges and approaches*, https://arxiv.org/abs/1803.06381.

[113] B. Li, M. He, W. Wu, A. K. Sangaiah, and G. Jeon, "Computation Offloading Algorithm for Arbitrarily Divisible Applications in Mobile Edge Computing Environments: An OCR Case," *Sustainability* , vol. 10, no. 5, p. 1611, 2018.

[114] S. Wang, Y. Zhao, L. Huang, J. Xu, and C. Hsu, "QoS prediction for service recommendations in mobile edge computing," *Journal of Parallel and Distributed Computing*.

[115] R. Roman, J. Lopez, M. Mambo et al., "Mobile edge computing, fog et al.: A survey and analysis of security threats and challenges," *Future Generation Computer Systems*, vol. 78, pp. 680–698, 2018.

[116] T. Wang, J. Zhou, M. Huang et al., "Fog-based storage technology to fight with cyber threat," *Future Generation Computer Systems*, vol. 83, pp. 208–218, 2018.

[117] K. Peng, R. Lin, B. Huang, H. Zou, and F. Yang, "Node importance of data center network based on contribution matrix of information entropy," *Journal of Networks*, vol. 8, no. 6, pp. 1248–1254, 2013.

[118] K. Peng and B. Huang, "The invulnerability studies on data center network," *International Journal of Security and Its Applications*, vol. 9, no. 11, pp. 167–186, 2015.

[119] K. Peng, V. C. M. Leung, L. Zheng et al., "Intrusion Detection System Based on Decision Tree over Big Data in Fog Environment," *Wireless Communications and Mobile Computing*, vol. 2018, Article ID 4680867, 10 pages, 2018.

[120] Cloudsim. http://www.cloudbus.org/cloudsim/.

[121] C. Sonmez, A. Ozgovde, and C. Ersoy, "EdgeCloudSim: An environment for performance evaluation of Edge Computing systems," in *Proceedings of the 2nd International Conference on Fog and Mobile Edge Computing, FMEC 2017*, pp. 39–44, Spain, May 2017.

[122] H. Gupta, A. Vahid Dastjerdi, S. K. Ghosh, and R. Buyya, "iFogSim: A toolkit for modeling and simulation of resource management techniques in the Internet of Things, Edge and Fog computing environments," *Software: Practice and Experience*, vol. 47, no. 9, pp. 1275–1296, 2017.

[123] M. M. Lopes, W. A. Higashino, M. A. Capretz, and L. F. Bittencourt, "MyiFogSim: A Simulator for Virtual Machine Migration in Fog Computing," in *Proceedings of the10th International Conference on Utility and Cloud Computing*, pp. 47–52, Austin, Texas, USA, December 2017.

Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 4801465, 12 pages
https://doi.org/10.1155/2018/4801465

Research Article

An Efficient Forwarding Capability Evaluation Method for Opportunistic Offloading in Mobile Edge Computing

Qian Wang[1], **Zhipeng Gao**[1], **Kun Niu**[2], **Yang Yang**[1] and **Xuesong Qiu**[1]

[1]*Network Management Center, The State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 10086, China*
[2]*School of Software Engineering, Institute of Network Technology, Beijing University of Posts and Telecommunications, Beijing 10086, China*

Correspondence should be addressed to Zhipeng Gao; gaozhipeng@bupt.edu.cn

Received 13 April 2018; Revised 18 August 2018; Accepted 4 September 2018; Published 25 September 2018

Guest Editor: Kok-Seng Wong

Opportunistic offloading can be utilized to offload computing tasks and traffic data in Mobile Edge Computing (MEC). To improve the ratio of successful data offloading and reduce unnecessary data redundancy in opportunistic forwarding process, some methods of evaluating a device's forwarding capability are proposed. However, most of these methods do not consider the temporal impact from device mobility and the efficiency influence from the capability computation process. To settle these problems, we proposed a Transient-cluster-based Capability Evaluation Method (TCEM) to evaluate a device's data forwarding capability. The TCEM can be divided into two steps. The first step aims to reduce computational complexity by evaluating a device's possibility of contacting the destination within a time constraint based on the transient cluster generated by our proposed Transient Cluster Detection Method (TCDM). The second step is to calculate a device's probability of directly and indirectly forwarding data to the destination. The probability as a metric of evaluating a device's forwarding capability can be used in different data forwarding strategies. Simulation results demonstrate that the TCEM-based data forwarding strategy outperforms other data forwarding strategies from the aspect of the proportion of the data delivery ratio to the data redundancy.

1. Introduction

Opportunistic offloading as an emerging communication paradigm can be used in MEC to offload computation or traffic data by leveraging the opportunistic network formed by mobile devices [1–3]. For example, in computation offloading an edge mobile device with limited computing resources can offload computing tasks through opportunistic communication to other nearby devices with idle computing resources, and in traffic offloading the edge devices that cache the repeatedly requested or popular contents transfer the contents to subscribers by opportunistic communication to reduce network traffic and devices' energy consumption. Opportunistic offloading is based on opportunistic connection between mobile devices causing random delay; thus it is suitable for non-real contents like e-mail, podcast, and weather forecast, which can tolerate some delay in the delivery.

To maximally offload computing tasks and traffic data within a time constraint, the ratio of successful data delivery over opportunistic data forwarding needs to be maximized. In opportunistic offloading, the offloading device could forward the data to opportunistic encountered devices via store-carry-forward mechanism to increase the ratio of successful delivery. Forwarding data to every opportunistic encountered device until the deadline of the data or the destination is reached maximizes the data delivery ratio, but this process produces large amounts of redundant data carried by devices who cannot contact the destination within valid time of the data. These redundant data consumes large amounts of devices' limited resources, such as storage capacity and battery life. Therefore, in this paper, we study how to reduce the redundant data in the opportunistic forwarding process while maximizing the ratio of successful data delivery, when a device sends its caching data with a time constraint to another

requested device. The key challenge is how to make strategy on replicating data to opportunistic encountered devices.

Recent literatures make data forwarding strategies based on comparison of the device's data forwarding capability. Devices carried data only replicate the data to devices with higher capability or the device with the highest capability. Although data redundancy is reduced through this method, there exist some problems on evaluating the device's data forwarding capability. For example, the evaluation methods [4, 5] are derived from communities formed by the aggregation of contact information. However, contact patterns are time-varying. By aggregating contact information into an aggregated contact graph, some important contact information, e.g., the burst behaviour, may be lost, weakening the accuracy of evaluating forwarding capability. Additionally some evaluation methods such as EER [6] need to get global contact information in advance. It is unrealistic in large-scale mobility scenarios.

In this paper, to optimize data delivery ratio and to reduce unnecessary redundancy, we propose a distributed method—Transient-cluster-based Capability Evaluation Method (TCEM). The first step of TECM is to judge the possibility of a device contacting the destination within the time constraint of the data using our proposed Transient Cluster Detection Method (TCDM). The second step is calculating the device's probability of encountering destination within the time constraint of the data as evaluation metric. The contributions of this method can be summarized as follows:

(i) Our proposed TCEM is distributed. Every device uses this method to evaluate its forwarding capability based on the transient contact information it obtains. Comparing with some centralized methods, which need to get global contact information in advance, our proposed method is more feasible under mobility circumstance of the MEC.

(ii) The proposed cluster detection method TCDM is formed by time-varying contact patterns of devices. Compared with other two transient cluster detection methods DRAFT [7] and CCM [8], TCDM is simpler and more efficient.

(iii) Compared with some methods, which directly calculate every encountered device's probability as forwarding capability evaluation metric, our proposed TCEM effectively reduces the computational complexity because we first evaluate the encountered device's possibility of forwarding data to the destination, and only the device with high possibility calculates the probability.

(iv) TCEM accurately evaluates a device's capability of forwarding data to a specific device within a time constraint, since it is based on individual pairs' Inter Contact Time (ICT) distributions and considers the influence of transient cluster's duration time in the probability calculation process.

The rest of the paper is organized as follows. Section 2 gives a brief overview of the related work. In Section 3, we give an overview of a TCEM-based data forwarding strategy. Section 4 presents our proposed TCEM in detail. It contains TCDM and probability calculation. Section 5 evaluates the performance of our approach by simulation, and Section 6 concludes the paper.

2. Related Work

Opportunistic data forwarding protocol originates from Epidemic routing [9] which floods the network. Although this flooding-based algorithm can achieve the highest packet delivery ratio, it cause large amounts of redundant data copies in the network. Later studies devote to develop forwarding protocols to approach the performance of Epidemic routing with lower cost, which is measured by the number of data copies in the network. Currently, the most successful approaches for opportunistic forwarding are social-aware strategies [10]. The community structure formed by exploiting social contact patterns of nodes has been widely utilized in the methods of evaluating a node's forwarding capability, since it is more reliable and less susceptible to the randomness of human mobility.

In Bubble Rap [4], SimBet [11], MDDPC-based [12], and RPC-based [13] data forwarding strategies, the data forwarding capability of a node is represented by its social importance that is the degree of facilitating the communication among other nodes. They are evaluated based on communities. For increasing the efficiency of the decision making process of routing, some schemes combined features are proposed. For example, SimbetTS [14] adds the tie strength for utility calculation, Oi [15] and SCORP [16] combine users' social ties with their interest for social-aware opportunistic routing, and GROUPS-Net [17] combines social awareness with a probabilistic approach using group meetings as a measure of social context to improve data delivery ratio. SAMPLER [18] uses social communities and social popularity metrics as they were introduced in the original Bubble Rap [4] scheme and adds the individual mobility and points of interest within a region to them, helping to achieve high delivery ratio and reducing network overhead. LASS [19] is proposed taking into account the difference of members' internal activity within each community. It utilizes different levels of local activity within communities to realize efficient data forwarding. However, in these strategies the community is formed based on the previous cumulative contact knowledge. The transient characteristics of node contact pattern, which would influence data forwarding performance, are ignored.

References [6, 7, 20–24] consider time-varying contact pattern between nodes. Wei et al. [20] develop a novel evaluation method to analytically predict the forwarding capability of a mobile node based on a proposed transient community structure. An efficient temporal closeness and centrality-based data forwarding strategy TCCB [23] are proposed by predicting nodes' future temporal social contact patterns. However, these two schemes are unsuitable for our proposed situation. Since evaluating a node's data forwarding capability [20] is based on the centrality within its transient community and [23] is based on the centrality within all nodes in the network, they do not reflect the node's capability of forwarding data to a destination. The method DRAFT [7] measures the

forwarding capability of a node through judging the node's 2-hop cluster whether it contains destination or not. The cluster is formed based on participants' cumulative or decayed contact duration. SimBetAge [24] improves upon SimBet by adopting an aged graph to calculate the social metrics dynamically. These approaches are not efficient enough since they improve the performance of data forwarding by replicating data to all high possibility nodes, which still produce a lot of redundant data. TSM [21] evaluates a node's forwarding capability based on three types of time-varying social metrics: betweenness centrality, similarity, and tie strength, which are derived from analysing two sets of social data. Reference [22] proposes a CAOF scheme, which includes intercommunity and intracommunity phase. In the intercommunity phase, the node with higher global activeness and source-to-destination probability is selected to serve as the relay. Besides, in the intracommunity phase, the forwarding decisions are determined by the local metrics. EER [6] proposes to evaluate a node's forwarding capability based on its encounter value (EV), which is the number of directly encountering participants within the data's valid time. These four methods are unpractical in some cases, since they need to get global contact information within data's valid time in advance. Comparatively our proposed TCEM is distributed and effectively reduces the computational complexity because we first evaluate the encountered device's possibility of forwarding data to the destination and only the device with high possibility calculates the probability.

The second step of our proposed TCEM needs to calculate the probability that a node transfers data to a specific node within data's valid time. ICT is defined as the time elapsing between two consecutive encounters of two devices; thus it can be used to calculate the probability. Most probability calculation methods use the aggregate ICT distribution: power law + exponential tail or exponential, which is obtained by considering the samples from all pairs together. However Hernández-Orallo et al. [25] propose using the distribution of aggregate ICT to represent individual pairs' ICTs distributions will not be correct in general, if the network is heterogeneous. Hence in this paper we consider the difference of individual pairs' ICTs distributions in the probability calculation process for more accurately evaluating the forwarding capability of a node.

3. TCEM-Based Data Forwarding Strategy

In the process of opportunistic data forwarding, the device with data makes data forwarding strategy based on encountered devices' data forwarding capability. In fact the method of evaluating a device's data forwarding capability within time constraint is independent from data forwarding strategies. Different forwarding capability evaluation methods can be applied in the same data forwarding strategy. For example, [8, 20] all forward data to devices with higher capability, whereas they use different forwarding capability evaluation methods. In this section we apply our proposed device's data forwarding capability evaluation method TCEM in a data forwarding strategy that the carried data participant forwards data to participants with higher capability.

The data forwarding strategy is composed of two parts. First it uses the TCEM to evaluate the connected device's capability of forwarding data to the destination. Then it decides whether to replicate data to this device on the evaluation result. The detailed process is illustrated as follows.

When a device u_1 with data connects a device u_2, u_1 first judges whether u_2 is carrying this data. If u_2 carries this data, u_1 directly skips u_2. Otherwise u_1 executes the data forwarding strategy as illustrated in Figure 1.

In part A, the strategy uses TCEM to evaluate u_2's data forwarding capability. The TCEM consists of two steps. In the first step, it judges whether u_2 has possibility of transferring data to the destination within the time constraint based on transient clusters. Only when u_2's transient cluster or his adjacent transient clusters contain destination, the second step will be continued, otherwise u_1 will not replicate data to u_2 and the strategy will end. The detailed Transient Cluster Detection Method (TCDM) is demonstrated in Section 4.1. The second step is to use ICT distribution between participants to calculate the probability P_{u_2} that u_2 forwards data to destination within the time constraint of the data. The calculation process is shown in Section 4.2. In our proposed method the probability is used as the metric of evaluating a participant's forwarding capability.

In part B, the strategy determines whether replicating data to u_2 according to the forwarding probability of u_2. If u_2's probability of contacting destination within data's valid time is higher than u_1, u_1 will replicate data to u_2, or u_1 will not. Devices with data execute the afore-mentioned data forwarding strategy, when they opportunistically connect devices without carrying this data. The process is continued until the destination receives the data within data's valid time.

4. Data Forwarding Capability Evaluation Method of the Device (TCEM)

Our proposed TCEM consists of two steps: possibility evaluation and probability calculation. These two steps are based on transient clusters. In this section, we first describe our proposed Transient Cluster Detection Method TCDM, and how to evaluate a device's possibility of contacting destination (Section 4.1). We then elaborate on how to calculate the device's probability of successful data delivery within time constraint t (Section 4.2).

4.1. Transient Cluster Detection Method. In this paper, every device $u_i(1, 2, \ldots, n)$ has a time-varying transient cluster T_i, which reflects a set of devices that have high contact rate with it, and the device u_i's adjacent cluster A_i refers to transient clusters sharing the same device with u_i. As illustrated in Figure 2, a dashed circle represents a transient cluster. T_1 is the device u_1's transient cluster at the time t. It contains u_2, u_3, u_4 which have high contact rate with u_1 at this moment, and T_2, T_3, T_4 are u_1's adjacent clusters. We set a device that has a possibility to transfer data to a destination, only when its transient cluster or its adjacent transient clusters contain the destination. For example, in Figure 2 if the destination of the data is u_3 or u_5, which is in u_1's transient cluster or in u_1's adjacent cluster T_2, respectively, the device u_1 will

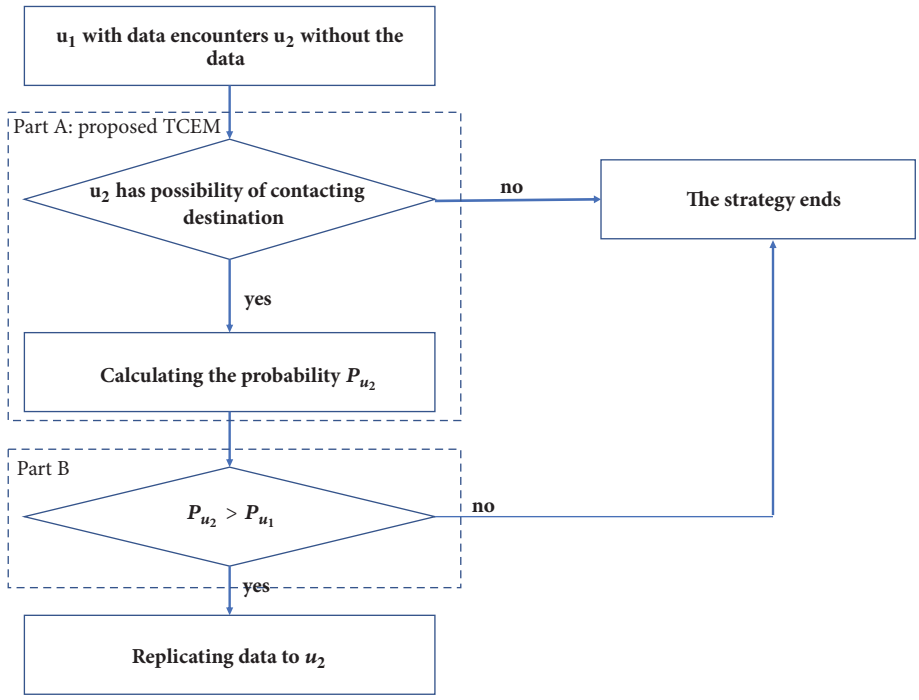


Figure 1: A TCEM-based opportunistic data forwarding strategy.

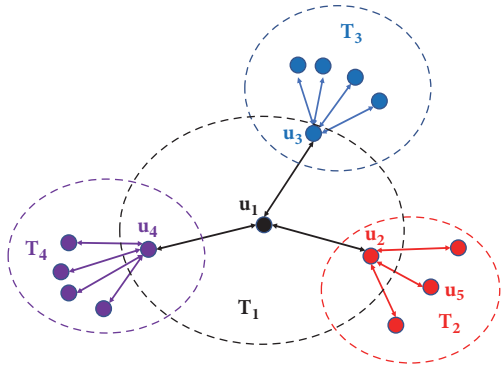


Figure 2: A device's transient cluster and its adjacent clusters.

have the possibility to transfer data to the destination. The evaluation standard is set because of two reasons. First the efficient content dissemination is mostly due to high contact rate nodes [26]. Second setting a device's neighbourhood view more than 2 hops does not improve the forwarding efficiency and even dramatically increases data redundancy [27].

Therefore, for evaluating a device u_i's possibility of contacting a destination, the device also stores its adjacent clusters A_i, besides storing its transient cluster T_i. The contact rate used to construct transient clusters can be represented by ICT which is the time interval between two consecutive encounters. We define the contact rate between two devices is high if and only if the ICT between them is shorter than a predefined threshold x. A device would delete devices whose ICTs are bigger than x from its transient cluster. The value of x is set based on different traces. Since pairwise ICTs are time-varying [21, 28], to accurately acquire the transient cluster and adjacent clusters of the current time, every x hours a device updates its transient cluster which only contains devices it encounters during this time period and updates its adjacent clusters when the device encounters a device. The process of building storage information of a device is described in Algorithm 1. It does not need centralized control and is independently performed by every device.

4.2. Calculating the Probability. After the first step of TCEM, a device's possibility of transferring data to destination has been determined. However, there is no guarantee that the data can be delivered to destination within a time constraint by the device, even though the path is at most 2 hops. To minimize the data redundancy, then in the second step, we calculate the device's probability of transferring data to destination within the time constraint, which represents the device's forwarding capability. The device with data only replicates data to encountered devices whose probabilities are higher than him. The calculation process is divided into two situations: (1) destination is in u_i's transient cluster, and (2) destination is in u_i's adjacent transient clusters. Table 1 gives the notations that are used in this section.

(1) The Destination in u_i's Transient Clusters. We have defined that a device has possibility to transfer data to another device only if the path between them is no more than 2 hops. Hence u_i can transmit data to the destination d via two routes, as illustrated in Figure 3. The first route is that u_i directly transmits data to d. The second route is that u_i indirectly transmits data to d through a device u_j, who is a member of u_i's transient cluster.

Input: Start time t_0.
Output: u_i's transient cluster T_i, u_i's adjacent clusters A_i.
Initialize: T_i={∅}, A_i= {∅}, t_j=0.
For every encountered participant u_j, u_i **do**
  t_{ij}= $t_{current}$-t_j // t_{ij} is the ICT at the current time
  If ($t_{ij} < x$) // u_j has high contact rate with u_i
    T_i=T_i+u_j // Adding encountered u_j to T_i
    t_j=$t_{current}$ // Recording the encountered time
    If (A_i contains T_j)// Whether encountered u_j before
      Update T_j in A_i
    Else
      A_i=A_i+T_j
  Else
    t_j=$t_{current}$ //Recording the time that u_i encounters u_j
End
For (($t_{current}$-t_0) % x==0) **do** // Updating T_i every x hours
  For every u_j in T_i **do**
    If (($t_{current}$-t_j) > x) //The ICT is larger than x hour
      T_i=T_i-u_j // Delete u_j from T_i
      A_i=A_i-T_j// Delete u_j's transient cluster from A_i
  End
End

Algorithm 1: Building information stored by the device u_i.

Table 1: Symbols and their definitions.

The symbols	The definitions
$P_{u_id}(t)$	the probability that u_i contacts d within t
$P^d_{u_id}(t)$	the probability that u_i directly contacts d within t
$P^{u_j}_{u_id}(t)$	the probability that u_i through a relay device u_j contacts d within t
$P_c(t_1, t_2)$	The probability of a transient cluster existing duration $t \in (t_1, t_2)$
$P_{u_id}(t \le t_v)$	u_i's probability of contacting d before t_v
$P^1_{u_id}(t_e, t_v)$	The probability of u_i contacting d when ($t_{ct} > t_v$)
$P^2_{u_id}(t_e, t_v)$	The probability of u_i contacting d when ($t_{ct} \le t_v$)
$P'_{u_id}(t)$	the probability that u_i only indirectly contacts d within t
$P^{1'}_{u_id}(t_e, t_v)$	The probability of u_i indirectly contacting d when ($t_{ct} > t_v$)
$P^{2'}_{u_id}(t_e, t_v)$	The probability of u_i indirectly contacting d when ($t_{ct} \le t_v$)
$P'_{u_id}(t \le t_v)$	u_i's probability of only indirectly contacting d before t_v

In this paper, data forwarding depends on opportunistic connections between devices. We assume the size of sensed data is small, and it can be completely transferred during one contact and the transmission time can be neglected. Hence the device's probability of transferring data to destination within the time constraint is equal to the probability of contacting destination within the time constraint. The ICT distribution between devices in the real-world mobility traces shows some probability distribution models. We use individual pairwise ICT distribution to calculate the probability that a device directly or indirectly contacts destination within the time constraint and use f_{ij} to represent the ICT distribution between devices u_i and u_j.

The probability $P_{u_id}(t)$ that u_i contacts d within time constraint t contains two parts: the probability of directly and indirectly contacts d. $P_{u_id}(t)$ is calculated by

$$P_{u_id}(t) = P^d_{u_id}(t) + \sum_{u_j \in D} P^{u_j}_{u_id}(t) \tag{1}$$

where

$$P^d_{u_id}(t) = \int_0^t f_{u_id}\, dt$$
$$P^{u_j}_{u_id}(t) = \int_0^t \left(\int_0^{t_0} f_{u_iu_j}\, dt \int_0^{t-t_0} f_{u_jd}\, dt \right) dt_0 \tag{2}$$

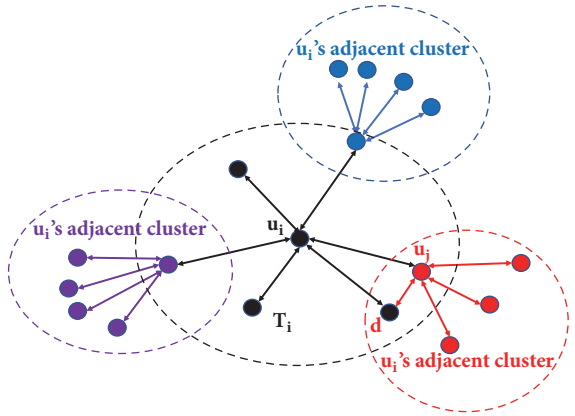


FIGURE 3: The destination is in a participant's transient cluster.

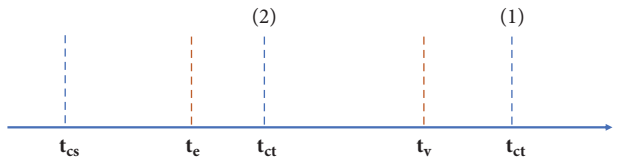


FIGURE 4: The relationship between cluster's duration and data's valid time.

$P^d_{u_i d}(t)$ is the probability that u_i directly contacts d within t, $P^{u_j}_{u_i d}(t)$ is the probability that u_i through a relay device u_j contacts d within t, and D is devices of u_i's transient cluster whose transient cluster contains d.

Since our proposed calculation method is based on the transient cluster, the relationship between duration of the transient cluster and valid time of data would influence the value of the time constraint in the probability calculation process. As illustrated in Figure 4, t_{cs} is the start time of a transient cluster. t_{ct} is the end time of the transient cluster; thus $(t_{ct} - t_{cs})$ is the transient cluster's duration time. t_e is the time that the device carried data encounters another device. t_v is the deadline of the valid time of the data. The first relationship is (1) that the duration time of the cluster is longer than the valid time of the data. The second relationship is (2) that the duration time of the cluster is shorter than the valid time of the data, and under this situation the time constraint of the data becomes $(t_{ct} - t_e)$.

In the first relationship $(t_{ct} > t_v)$, duration of the transient cluster is longer than valid time of the data. Equation (3) is the probability of this situation, where f_c is the distribution of the duration time of the transient cluster. In this situation, the real constraint time of the data is $(t_v - t_e)$. Equation (4) is the probability $P^1_{u_i d}(t_e, t_v)$ that u_i contacts d within data's valid time under this situation. $P_{u_i d}(t_v - t_e)$ is the probability that u_i contacts d within time constraint $(t_v - t_e)$.

To study f_c, we run the TCDM on two real datasets, Infocom6 [29] and Cambridge [30]. The detailed information about these two datasets is illustrated in Table 1. We observe the duration time of a transient cluster on daily basis and find its distribution can be approximated by exponential distribution. We take one transient cluster in Cambridge trace as an example, shown in Figure 5. The approximation does not seem perfect, since the samples used to train the distribution are limited. It should be better if more data are used.

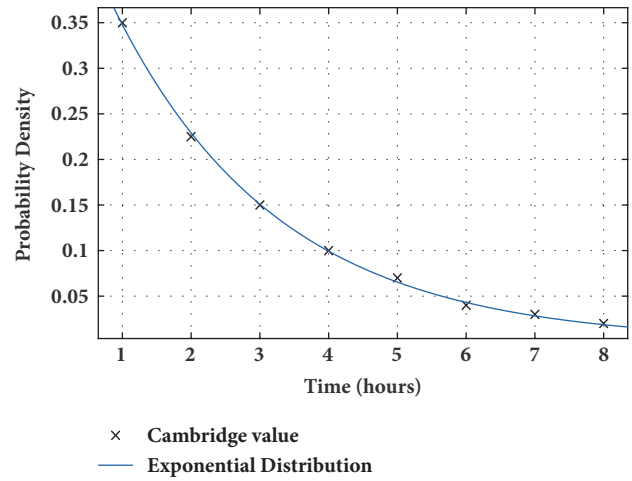


FIGURE 5: The PDF of a transient cluster's duration time.

$$P_c(t_v - t_{cs}, \infty) = \int_{t_v - t_{cs}}^{\infty} f_c \, dt \tag{3}$$

$$P^1_{u_i d}(t_e, t_v) = P_{u_i d}(t_v - t_e) P_c(t_v - t_{cs}, \infty) \tag{4}$$

The second relationship $(t_{ct} \leq t_v)$ is that duration of the transient cluster is smaller than valid time of the data. Equation (5) is the probability of this situation. In this situation, the real constraint time of the data is $(t_{ct} - t_e)$. Equation (6) is the probability $P^2_{u_i d}(t_e, t_v)$ that u_i contacts d within data's valid time under this situation. $P_{u_i d}(t_{ct} - t_e)$ is the probability that u_i contacts d within time constraint $(t_{ct} - t_e)$.

$$P_c(t_e - t_{cs}, t_v - t_{cs}) = \int_{t_e - t_{cs}}^{t_v - t_{cs}} f_c \, dt \tag{5}$$

$$P^2_{u_i d}(t_e, t_v) = P_{u_i d}(t_{ct} - t_e) P_c(t_e - t_{cs}, t_v - t_{cs}) \tag{6}$$

In summary, (7) is the probability $P_{u_i d}(t \leq t_v)$ that u_i contacts the destination before the expiration of the data's valid time.

$$P_{u_i d}(t \leq t_v) = P^1_{u_i d}(t_e, t_v) + P^2_{u_i d}(t_e, t_v) \tag{7}$$

(2) The Destination in u_i's Adjacent Transient Clusters. We discuss the situation that the destination d is not in u_i's transient cluster, but it is in u_i's adjacent transient clusters. Figure 6 shows an example that d is in the u_i's adjacent cluster T_j. u_i can transmit data to d through u_j. The probability $P'_{u_i d}(t)$ that u_i encounters destination d within time constraint t only by indirect contacting is calculated in (8), where T is devices of u_i's transient cluster whose transient clusters contain d.

$$P'_{u_i d}(t) = \sum_{u_j \in T} P^{u_j}_{u_i d}(t) \tag{8}$$

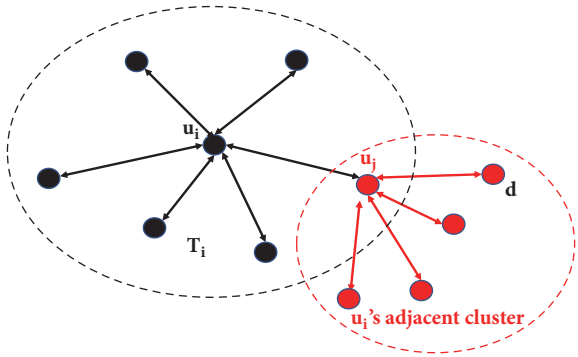


Figure 6: The destination is in a device's adjacent cluster.

The probability calculation is based on transient clusters. The relationship between the time, during which destination is in the opportunistic connected device's adjacent transient cluster, and the valid time of the data causes two possibilities. It is similar with Section 4.2(1).

When $(t_{ct} > t_v)$, (9) is the probability $P^{1'}_{u_i d}(t_e, t_v)$ that u_i encounters d, under this situation that the duration of transient cluster is longer than data's valid time, where $P'_{u_i d}(t_v - t_e)$ is calculated based on (8).

$$P^{1'}_{u_i d}(t_e, t_v) = P'_{u_i d}(t_v - t_e) \int_{t_v - t_{cs}}^{\infty} f_c \, dt \tag{9}$$

When $(t_{ct} \leq t_v)$, (10) is the probability $P^{2'}_{u_i d}(t_e, t_v)$ that u_i encounters destination d, under this situation that the duration of transient cluster is smaller than valid time of the data, where $P'_{u_i d}(t_{ct} - t_e)$ is calculated based on (8). In summary, (11) is the probability that u_i contacts destination d before the expiration of valid time of the data.

$$P^{2'}_{u_i d}(t_e, t_v) = P'_{u_i d}(t_{ct} - t_e) \int_{t_e - t_{cs}}^{t_v - t_{cs}} f_c \, dt \tag{10}$$

$$P'_{u_i d}(t \leq t_v) = P^{1'}_{u_i d}(t_e, t_v) + P^{2'}_{u_i d}(t_e, t_v) \tag{11}$$

5. Performance Evaluation

Our proposed TCEM is a method of measuring a device's data forwarding capability. Since it is based on the transient cluster, the performance of forming transient clusters will influence the performance of the method. Therefore we evaluate the performance of the Transient Cluster Detection Method (TCDM) of TCEM. Meanwhile, we evaluate the performance of a TCEM-based data forwarding strategy. In the experiments, the data delivery ratio is the proportion of data items, which are successfully delivered to the destination through opportunistic forwarding before data expires. It is improved by increasing the number of data copies, which represented the network overhead. Hence, to reflect the efficiency, the performance of the data forwarding strategy is evaluated by the proportion of data delivery ratio to the network overhead.

Table 2: Trace summary.

	Infocom6	Cambridge
Environment	Conference	City
Duration (days)	3	12
Device number	78	36
Inter-probe time/s	120	600

Our experiments are based on two real datasets, Infocom6 [29] and Cambridge [30]. These datasets are formed based on the devices that periodically detect their peers via Bluetooth interfaces, and a contact is recorded when two devices move into the communication range of each other. The details of two datasets are shown in Table 2.

5.1. Transient Cluster Evaluation. We compare the performance of TCDM with two transient cluster detection methods: Contact-burst-based Clustering Method (CCM) [8] and Distributed Rise and Fall spatio-Temporal (DRAFT) clustering method [7]. Besides the distributed feature and the feasibility of the method, the size of clusters detected by the method would also influence efficiency of the data forwarding strategy, as a data forwarding strategy decides whether to forward data to an encountered participant by searching all numbers of this participant's cluster. Hence in this section we compare the performance of methods from three metrics: mean cluster size, max cluster size, and the used cluster size of evaluating forwarding capability.

CCM. It detects transient clusters by clustering pairs of nodes with similar contact bursts together. A contact burst between two nodes refers to a period when contacts frequently appear between these two nodes. The transient cluster detected by CCM is time-varying, and the cluster's forming process needs central control.

DRAFT. It is distributed. Every node forms a cluster based on cumulated or decayed contact duration. It uses three parameters τ, t, δ to govern the rate at which clusters grow and decay. If the cumulated contacted duration is longer than τ, the encountered device will be added to the cluster. Otherwise the device will be deleted from the cluster. A time frame of length t seconds governs the interval at which the cumulative durations for each device are decayed. At the end of each time frame, the contacted duration is decreased by multiplying the parameter δ.

Our proposed TCDM is distributed. Every device forms a time-varying transient cluster, which contains a set of devices that have high contact rate with him. In this paper, we set a device that will add an encountered device to his transient cluster, only if these two devices' intercontact time is lower than predetermined parameter x. The value of x is set empirically based on traces.

In these two datasets, we found that each pairwise contact is a series of contact bursts during which two devices' contact rate is high, and two devices' intercontact time is lower than 1 hour in these contact burst periods. For example, we choose any two devices from the Cambridge dataset. The contact is

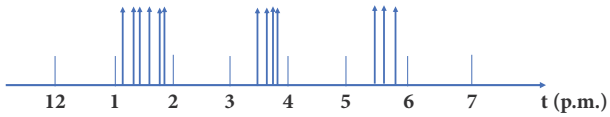


Figure 7: Contact rate of two nodes.

illustrated in Figure 7. An arrow represents a contact between two devices. Thus we set $x = 1$. If these two devices do not contact within one hour that means the two devices' contact rate is not high, and a device will be deleted from another device's transient cluster.

Mean Cluster Size. Figure 8 shows that the mean cluster size of TCDM is the smallest among these three methods. CCM's mean cluster size is the biggest, and it is almost three times bigger than TCDM. We illustrate the reasons as follows.

In the DRAFT method, a cluster is a device-centric one-hop cluster, but a device is added to or deleted from a cluster based on the cumulative or decayed encounter duration time. In these two datasets, the cluster of DRAFT will contain high contact rate devices. For example, in Infocom6 dataset, two devices with high contact rate mean these two devices in a contact burst period. Mean contact duration between devices is 24 s. Two participants contact 6 times during a contact burst period. Therefore, in a contact burst period, two devices' cumulative encounter duration time is 144 s, which exceeds the predetermined threshold 120 s. The device will be added to another device's cluster. However, since a device is deleted from a cluster based on the decayed duration time, it might lead to a cluster that still accumulated some devices which do not have high contact rate with the central device now, whereas members of a TCDM cluster are devices, which have high contact rate with a central device. Therefore, the cluster detected by DRAFT is bigger than TCDM.

The mean size of DRAFT's cluster is smaller than CCM, because a cluster detected by DRAFT method aims to detect devices which have high contact frequency with a central device during a period. It is a one-hop cluster. However in CCM any two devices have high contact rate duration; this period will be added to a cluster. The hops do not have limitation. It is absolutely bigger than a DRAFT cluster.

In summary, comparing with DRAFT, although the complexity of TCDM's cluster forming method is similar, TCDM's cluster size is smaller than DRAFT. Comparing with CCM, besides the cluster size TCDM is more feasible and efficient. We demonstrate it from two aspects. First CCM needs to get all devices' contact information in advance. Second in the process of forming cluster, CCM has to traverse all clusters to merge similar clusters. This process wastes large amounts of computing resources and time. However, our proposed TCDM is distributed. Every device forms his transient cluster only based on two devices' real-time contact frequency. Therefore, comparing with CCM, TCDM is more feasible and efficient.

Max Cluster Size. Figure 9 shows the max cluster detected by these three methods. The max cluster detected by TCDM and DRAFT method is smaller than cluster detected by CCM. CCM's max cluster contains more than half of the whole devices. It would extremely increase computation complexity.

The Cluster Size of Evaluating Forwarding Capability. DRAFT-based and our proposed TFCM-based data forwarding strategy determine whether forwarding data to an encountered device is based on its 2-hop cluster. Therefore, a device's 2-hop cluster size is very important for the performance of these two methods. However, CCM-based data forwarding strategy is based on a cluster. Hence, we compare mean 2-hop cluster size of TCDM and DRAFT, and mean cluster size of CCM. Figure 10 shows our proposed TCDM's cluster is the smallest among these three cluster detection methods. It includes more or less 30% of all devices, but DRAFT's 2-hop cluster contains almost half of all devices and CCM's cluster contains more than half of all devices. The reasons are illustrated as follows. First, since the 1-hop cluster of TCDM is smaller than 1-hop cluster of DRAFT, 2-hop cluster of TCDM is smaller than 2-hop cluster of DRAFT. Second, in DRAFT, a cluster contains 2-hop participants but a cluster of CCM does not limit hops; hence the cluster size of CCM is bigger than DRAFT.

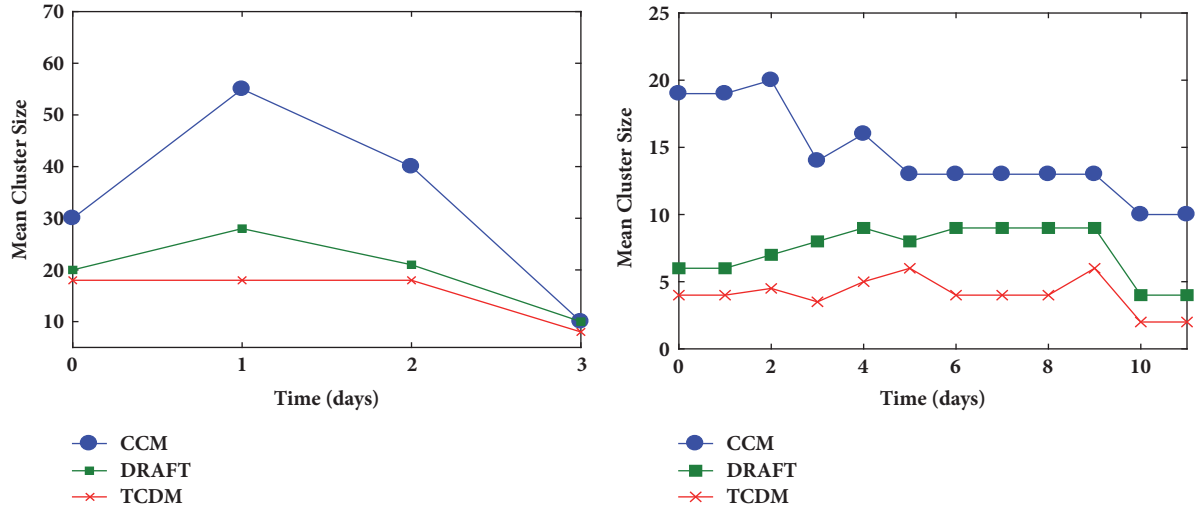
In conclusion, first, the cluster size of our proposed TCDM is the smallest. It makes the efficiency of judging the forwarding capability of a device by searching all members in the cluster the best. In addition, comparing with CCM, our proposed TCDM is more feasible and more efficient because of its distribution and simplicity of cluster formation method. Comparing with DRAFT's cluster, which cannot accurately express the change of contact rate between participants, TCDM can reflect the change of devices' contact in real time. Since evaluating a participant's data forwarding capability is based on contact rate between devices, TCDM is more accurate in the evaluation of a device's forwarding capability.

5.2. Data Forwarding Performance Evaluation. In the experiment for fairness sources and destinations are picked randomly, and the data's generation time is randomly chosen in the daytime because nodes' activity time is low at night, which may result in inaccurate comparison. The performance is measured by the metric: the proportion of data delivery ratio to network overhead. Our proposed TCEM-based data forwarding strategy compares with Epidemic [9], Bubble Rap [4], DRAFT-based [7], and CCM-based [8] data forwarding strategy.

Epidemic. The data item is forwarded to every encountered device without the same data. It serves as upper bound.

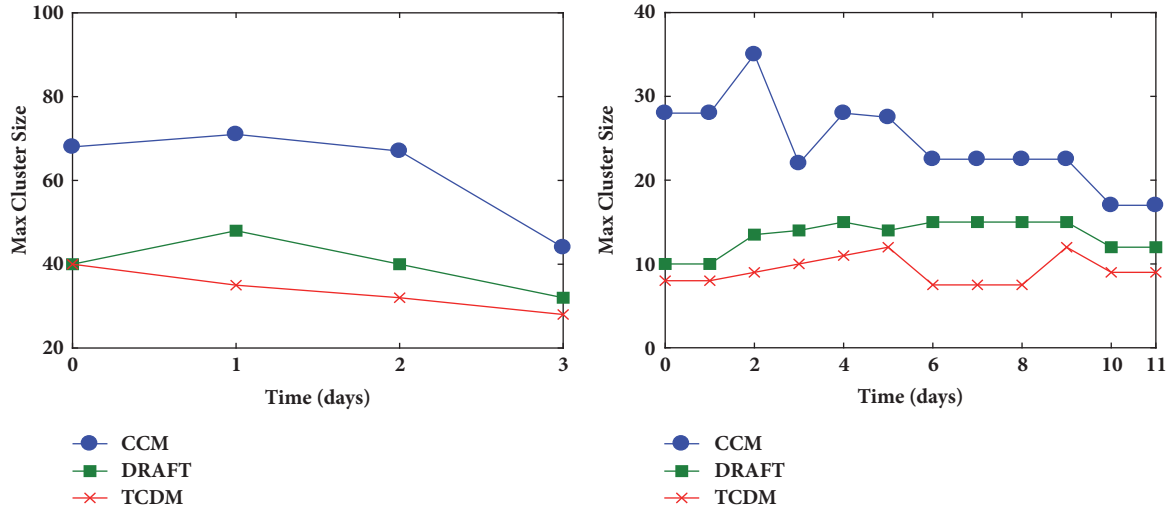
Bubble Rap. This strategy utilizes both centrality and community. CPM (K-clique) is used to detect communities. The data item is always forwarded to a higher centrality device, until it reaches a device that belongs to the same community as the destination. When the data item reaches the destination community, it is forwarded to higher centrality device within the community's scope until the destination is reached.

DRAFT-Based Date Forwarding Strategy. The data will be replicated to encountered devices whose 2-hop clusters contain the destination until a carried data device encounters destination.



(a) Infocom6 (b) Cambridge

Figure 8: Mean cluster size.



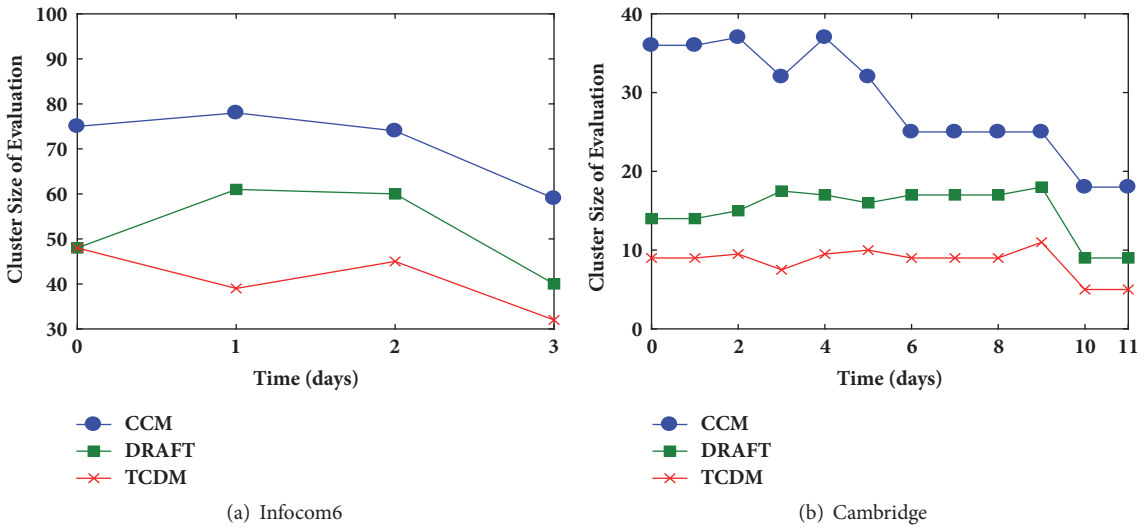
(a) Infocom6 (b) Cambridge

Figure 9: Max cluster size.

CCM-Based Data Forwarding Strategy. It utilizes transient cluster (TC) as the forwarding unit, and data is always forwarded to the TC with better relaying capability to the destination within data's time constraint. Once data reaches a new TC with larger relaying capability, the data is distributed to all nodes met in the TC. The relaying capability of the current TC is computed by summing the probability that each participant of the TC appears in destination's transient cluster within data's valid time. The carried data device deletes data when he has neither encountered a device in TC with larger relaying capability nor gone to a TC with larger relaying capability in an appointed time period. The above-mentioned process ended when data reached the destination or the time exceeds data's valid time.

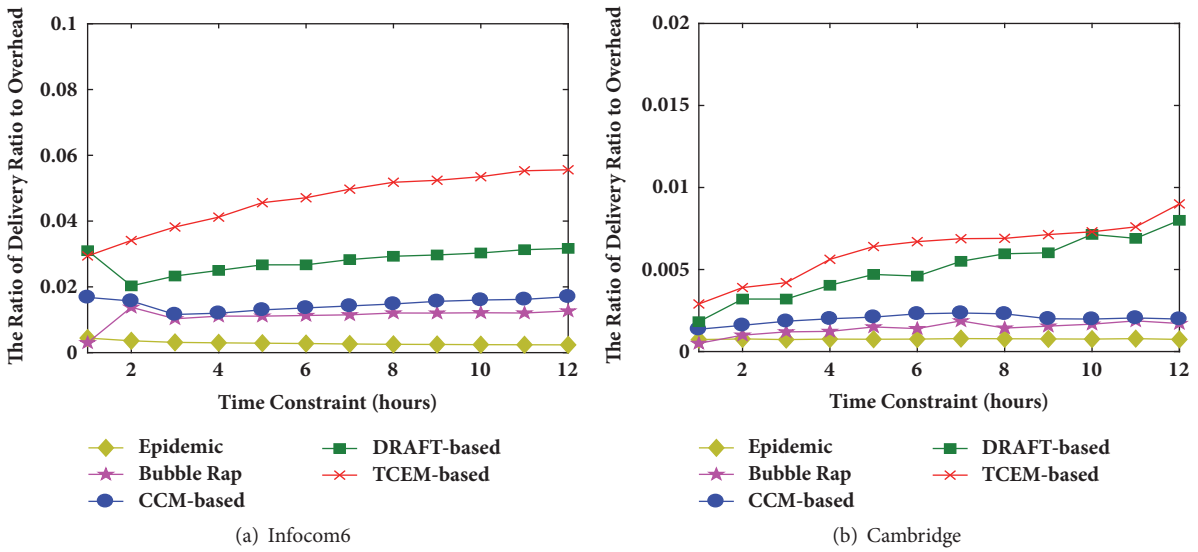
The result is shown in Figure 11. Considering the delivery ratio and the overhead the performance of Bubble Rap is the worst comparing with TCEM-based, DRAFT-based, and CCM-based data forwarding strategies. The reason is that these three strategies consider device's contact rate is time-varying, but Bubble Rap uses aggregate contact information which cannot reflect device's contact rate within data's valid time. Comparing with other three data forwarding strategies, Bubble Rap cannot accurately evaluate device's forwarding capability within the time constraint. Thus, its data forwarding performance is the worst.

Comparing with DRAFT-based and CCM-based data forwarding strategies, our proposed TCEM-based data forwarding strategy's delivery ratio is the lowest, but at the same time its overhead is also the lowest as illustrated in Figures 12 and 13. Integrating these two parameters its proportion of data delivery ratio to network overhead is highest. First, data forwarding of DRAFT and TCEM is based on the



(a) Infocom6 (b) Cambridge

FIGURE 10: The cluster size of evaluating forwarding capability.

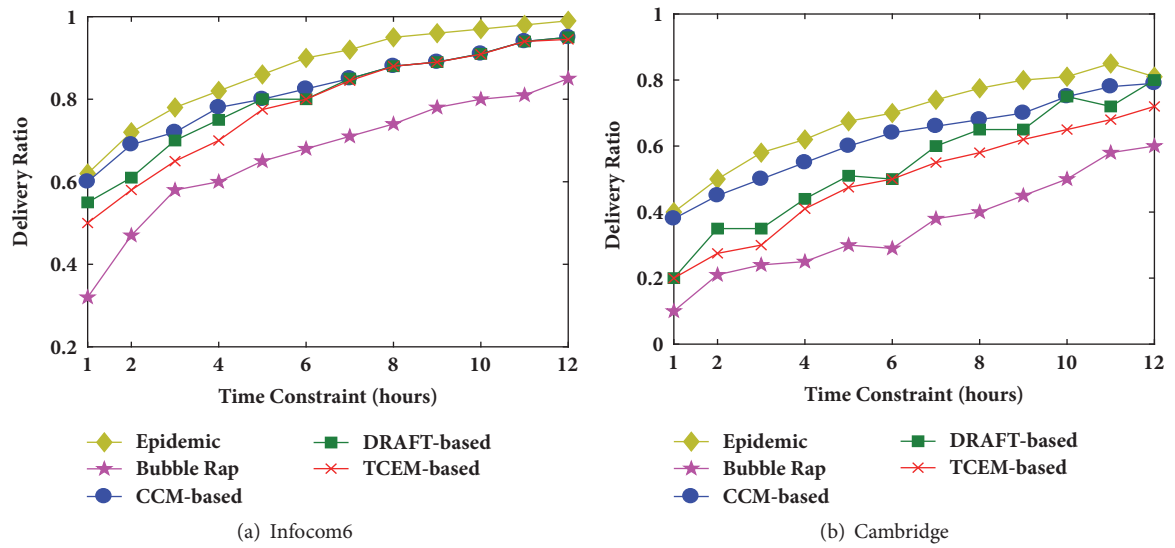


(a) Infocom6 (b) Cambridge

FIGURE 11: The proportion of delivery ratio to overhead.

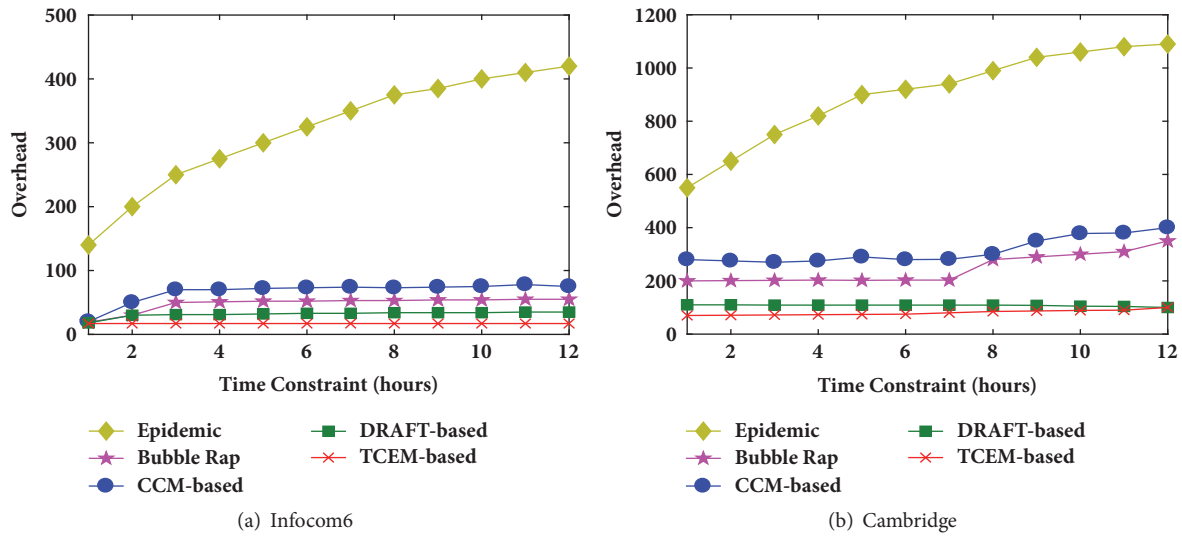
encountered device's 2-hop transient cluster. DRAFT's 2-hop cluster size is larger than TCEM's 2-hop cluster, and it contains all members of TCEM's cluster. In the process of data forwarding, DRAFT forwards data to all encountered devices whose 2-hop cluster contains destination, but TCEM forwards data to an encountered device whose 2-hop cluster contains destination and probability of encountering destination is higher than the carried data participant. TCEM-based data forwarding strategy reduces some unnecessary overhead of replicating data to devices whose probability of encountering destination is low. Thus, although TCEM-based data forwarding strategy's data delivery ratio is still a little lower than DRAFT-based data forwarding strategy, it greatly reduces overhead.

Second, data delivery ratio and overhead of CCM-based data forwarding strategy are the highest. Comparing with DRAFT and TCEM only forward data to the encountered device, CCM's data forwarding is based on the transient cluster. The data is distributed to all devices encountered in the transient cluster. Meanwhile, CCM's transient cluster is the largest. It contains all devices that directly or indirectly and frequently contact with the encountered device. Considering these two reasons, CCM's data delivery ratio is the highest, but its overhead is also high. At the same time, since in CCM strategy the carried data device deletes data when he has neither encountered a device in TC with larger relaying capability nor gone to a TC with larger relaying capability in every appointed time period, it will lead to the increase of the overhead as data may be copied many times to the same device. Hence, CCM's overhead is extremely high and its proportion of data delivery ratio to network overhead is the lowest. Moreover, the computation complexity of



(a) Infocom6 (b) Cambridge

Figure 12: Data delivery ratio.



(a) Infocom6 (b) Cambridge

Figure 13: The network overhead.

CCM-based data forwarding strategy is greatly high, because it needs to find every encountered device's current TC and calculate the probability that every member of TC encounters destination.

In conclusion, integrating delivery ratio and overhead the data forwarding performance of Bubble Rap is the worst. Comparing with DRAFT-based and CCM-based data forwarding strategies, although our proposed TCEM-based data forwarding strategy's data delivery ratio is the lowest, taking account of delivery ratio and network overhead, the performance of our proposed method is the best.

6. Conclusions

In this paper, we proposed a method TCEM to evaluate the device's data forwarding capability. The method is based on the time-varying transient cluster and we also propose a method TCDM to detect the transient cluster. Simulation results show that the data forwarding strategy based on our proposed TCEM outperforms other existing data forwarding approaches.

Data Availability

The Infocom6 and Cambridge datasets supporting this meta-analysis are from previously reported studies and datasets, which have been cited.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work is supported by National Science & Technology Pillar Program (2015BAH03F02) and National Key Research and Development Program of China (2016YFE0204500).

References

[1] D. Xu, Y. Li, X. Chen et al., "A survey of opportunistic offloading," *IEEE Communications Surveys & Tutorials*, 2018.

[2] Y. Mao, C. You, J. Zhang et al., "A survey on mobile edge computing: The communication perspective," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2322–2358, 2017.

[3] Y. Han, T. Luo, D. Li, and H. Wu, "Competition-based participant recruitment for delay-sensitive crowdsourcing applications in D2D networks," *IEEE Transactions on Mobile Computing*, vol. 15, no. 12, pp. 2987–2999, 2016.

[4] P. Hui, J. Crowcroft, and E. Yoneki, "BUBBLE Rap: social-based forwarding in delay-tolerant networks," *IEEE Transactions on Mobile Computing*, vol. 10, no. 11, pp. 1576–1589, 2011.

[5] V. Erramilli, M. Crovella, A. Chaintreau, and C. Diot, "Delegation forwarding," in *Proceedings of the 9th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc '08)*, pp. 251–260, May 2008.

[6] H. Chen and W. Lou, "Contact expectation based routing for delay tolerant networks," *Ad Hoc Networks*, vol. 36, pp. 244–257, 2016.

[7] M. Orlinski and N. Filer, "The rise and fall of spatio-temporal clusters in mobile ad hoc networks," *Ad Hoc Networks*, vol. 11, no. 5, pp. 1641–1654, 2013.

[8] X. Zhang and G. Cao, "Transient community detection and its application to data forwarding in delay tolerant networks," *IEEE/ACM Transactions on Networking*, vol. 25, no. 5, pp. 2829–2843, 2017.

[9] W. Mitchener and A. Vadhat, "Epidemic routing for partially connected ad hoc networks," Technical Report CS-2000-06, 2000.

[10] Y. Li, T. Wu, P. Hui, D. Jin, and S. Chen, "Social-aware D2D communications: qualitative insights and quantitative analysis," *IEEE Communications Magazine*, vol. 52, no. 6, pp. 150–158, 2014.

[11] E. M. Daly and M. Haahr, "Social network analysis for routing in disconnected delay-tolerant MANETs," in *Proceedings of the 8th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc'07)*, pp. 32–40, September 2007.

[12] H. Zhou, L. Tong, T. Jiang, S. Xu, J. Fan, and K. Lv, "Maximum data delivery probability-oriented routing protocol in opportunistic mobile networks," *Peer-to-Peer Networking and Applications*, vol. 10, no. 3, pp. 500–509, 2017.

[13] J. Wu, J. Wang, L. Liu, M. Tanha, and J. Pan, "A data forwarding scheme with reachable probability centrality in DTNs," in *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC '16)*, pp. 1–6, April 2016.

[14] E. M. Daly and M. Haahr, "Social network analysis for information flow in disconnected delay-tolerant MANETs," *IEEE Transactions on Mobile Computing*, vol. 8, no. 5, pp. 606–621, 2009.

[15] L. Amaral, R. Sofia, P. Mendes, and W. Moreira, "Oi! - Opportunistic data transmission based on Wi-Fi direct," in *Proceedings of the IEEE INFOCOM - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS '16)*, pp. 578-579, San Francisco, Calif, USA, April 2016.

[16] W. Moreira, P. Mendes, and S. Sargento, "Social-aware opportunistic routing protocol based on users interactions and interests," in *Proceedins of the International Conference on Ad Hoc Networks*, pp. 100–115, Springer, 2013.

[17] I. O. Nunes, C. Celes, P. O. S. Vaz de Melo, and A. A. F. Loureiro, "GROUPS-NET: Group meetings aware routing in multi-hop D2D networks," *Computer Networks*, vol. 127, pp. 94–108, 2017.

[18] I. O. Nunes, C. Celes, I. Nunes, P. O. S. Vaz De Melo, and A. A. F. Loureiro, "Combining spatial and social awareness in D2D opportunistic routing," *IEEE Communications Magazine*, vol. 56, no. 1, pp. 128–135, 2018.

[19] Z. Li, C. Wang, S. Yang, C. Jiang, and X. Li, "LASS: Local-activity and social-similarity based data forwarding in mobile social networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 1, pp. 174–184, 2015.

[20] W. Gao, G. Cao, T. La Porta, and J. Han, "On exploiting transient social contact patterns for data forwarding in delay-tolerant networks," *IEEE Transactions on Mobile Computing*, vol. 12, no. 1, pp. 151–165, 2013.

[21] D. Xie, X. Wang, L. Liu, and L. Ma, "Exploiting time-varying graphs for data forwarding in mobile social Delay-Tolerant Networks," in *Proceedings of the 24th IEEE/ACM International Symposium on Quality of Service (IWQoS '16)*, June 2016.

[22] J. Tao, H. Wu, S. Shi, J. Hu, and Y. Gao, "Contacts-aware opportunistic forwarding in mobile social networks: A community perspective," in *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC '18)*, pp. 1–6, April 2018.

[23] H. Zhou, V. C. M. Leung, C. Zhu, S. Xu, and J. Fan, "Predicting temporal social contact patterns for data forwarding in opportunistic mobile networks," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 11, pp. 10372–10383, 2017.

[24] J. Á. B. Link, N. Viol, A. Goliath, and K. Wehrle, "SimBetAge: utilizing temporal changes in social networks for pocket switched networks," in *Proceedings of the 1st ACM Workshop on User-Provided Networking: Challenges and Opportunities (U-NET '09)*, pp. 13–18, ACM, New York, NY, USA, December 2009.

[25] E. Hernández-Orallo, J. C. Cano, C. T. Calafate, and P. Manzoni, "New approaches for characterizing inter-contact times in opportunistic networks," *Ad Hoc Networks*, vol. 52, pp. 160–172, 2016.

[26] A.-K. Pietilänen and C. Diot, "Dissemination in opportunistic social networks: the role of temporal communities," in *Proceedings of the 13th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc '12)*, pp. 165–174, June 2012.

[27] T. Phe-Neau, M. Dias De Amorim, and V. Conan, "The strength of vicinity annexation in opportunistic networking," in *Proceedings of the 32nd IEEE Conference on Computer Communications (IEEE INFOCOM '13)*, pp. 3369–3374, April 2013.

[28] S. C. Nelson, M. Bakht, R. Kravets, and A. F. Harris, "Encounter-based routing in dtns," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 13, no. 1, pp. 56–59, 2009.

[29] N. Eagle and A. Pentland, "Reality mining: sensing complex social systems," *Personal and Ubiquitous Computing*, vol. 10, no. 4, pp. 255–268, 2006.

[30] J. Scott, R. Gass, J. Crowcroft, P. Hui, C. Diot, and A. Chaintreau, "CRAWDAD trace - Haggle," 2009, http://crawdad.cs.dartmouth.edu/cambridge/haggle.

Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 1324897, 10 pages
https://doi.org/10.1155/2018/1324897

Research Article

Dynamic Service Request Scheduling for Mobile Edge Computing Systems

Ying Chen, Yongchao Zhang, and Xin Chen

Computer School, Beijing Information Science and Technology University (BISTU), Beijing 100101, China

Correspondence should be addressed to Ying Chen; chenying@bistu.edu.cn

Received 20 April 2018; Accepted 3 July 2018; Published 13 September 2018

Academic Editor: Kok-Seng Wong

Nowadays, mobile services (applications) running on terminal devices are becoming more and more computation-intensive. Offloading the service requests from terminal devices to cloud computing can be a good solution, but it would put a high burden on the network. Edge computing is an emerging technology to solve this problem, which places servers at the edge of the network. Dynamic scheduling of offloaded service requests in mobile edge computing systems is a key issue. It faces challenges due to the dynamic nature and uncertainty of service request patterns. In this article, we propose a Dynamic Service Request Scheduling (DSRS) algorithm, which makes request scheduling decisions to optimize scheduling cost while providing performance guarantees. The DSRS algorithm can be implemented in an online and distributed way. We present mathematical analysis which shows that the DSRS algorithm can achieve arbitrary tradeoff between scheduling cost and performance. Experiments are also carried out to show the effectiveness of the DSRS algorithm.

1. Introduction

With the rapid development of Information Technology and increasing promotion of terminal devices [1], the mobile services (applications) running on terminal devices are becoming more and more complex and computation-intensive [2, 3]. However, the computing capacity and battery life of terminal devices are generally limited, and these devices cannot afford to process all these service requests locally on devices. To solve this problem, some researches propose to offload the service requests from terminal devices to cloud computing, which has more computing resources and larger capacity [4–8]. Nevertheless, cloud computing is usually located remotely that is far away from terminal devices. Besides, with the increasing popularity of mobile services running on terminal devices, scheduling all the offloaded service requests to cloud computing can put a significant burden on the networks [9, 10]. To meet this challenge, recent researches have been proposed to put edge servers with computing capacities at the edge of the networks in close proximity to terminal devices. Mobile Edge Computing (MEC) is an emerging technology based on this idea [11–14], and it has drawn extensive attention from both academy and industry [15–18].

One of the key issues in MEC research is how to schedule service requests [2, 19, 20]: when a large number of service requests are offloaded, how to schedule service requests among multiple MEC systems in order to reduce scheduling cost while providing performance guarantees. It is intuitive that there exist tradeoffs between scheduling cost and performance. Besides, the service request scheduling problem among multiple MEC systems is challenging due to several reasons. Firstly, as terminal devices are moving and the service environment varies over time [21, 22], how to make dynamic request scheduling decisions in accordance with the uncertainty of request patterns and changing environment is a great challenge [23]. Secondly, with the increasing promotion of terminal devices and mobile services, both the number of terminal devices and mobile services are rising dramatically, making the service request scheduling problem more complicated.

Some existing researches have studied the service request scheduling problem in MEC systems. Reference [24] modelled the server in the MEC as one $M/M/1$ queue. Reference

[2] assumed the offloaded service requests arrived at the MEC system according to a Poisson process. These works assumed the request arrival followed certain distribution. However, in reality, the request arrival process is highly dynamic, and the statistical information of request arrival can hardly be obtained or precisely predicted [25, 26]. Besides, with the number of terminal devices and mobile services increasing, traditional centralized optimization techniques such as combination optimization and dynamic programming may suffer from high-complexity and result in long execution time.

In this article, we introduce a dynamic online service request scheduling mechanism which requires no prior information of the statistical information of request arrivals. Specifically, the request scheduling among multiple MEC systems is formulated as an optimization problem, and the goal is to minimize request scheduling cost while providing performance guarantees. Based on Lyapunov optimization techniques, we propose a Dynamic Service Request Scheduling (DSRS) algorithm. DSRS uses a parameter V to control the tradeoff between scheduling cost and queue length. Mathematical analysis is presented which proves that DSRS is $O(1/V)$-optimal with respect to the average scheduling cost while still bounding the average queue length by $O(V)$. Experiments are also conducted which demonstrate that DSRS can make dynamic control decisions to adjust to variable environments and achieve the tradeoff between scheduling cost and queue length.

The remainder of this article is organized as follows. In Section 2, we present the system model for dynamic request scheduling among multiple MEC systems and formulate the optimization problem. In Section 3, based on Lyapunov optimization techniques, we propose an online and Dynamic Service Request Scheduling algorithm. Theoretical analysis of the scheduling algorithm is presented in Section 4. Experiments are conducted to evaluate the efficiency and effectiveness of the scheduling algorithm in Section 5. We conclude this article in Section 6.

2. System Model

2.1. Overview. Consider n mobile edge computing (MEC) systems. Each MEC system has an edge server virtualized to m virtual machines to process offloaded requests of m types of services from the terminal devices [25, 27]. More specifically, the i-th virtual machine on each edge server in the MEC system serves the offloaded requests for the i-th type of service. Let I be the collection of indexes for applications and J be the collection of indexes for edge server in the MEC systems. Without loss of generality, the edge servers in different MEC systems are supposed to be heterogeneous. We consider a time-slotted model and the length of time slot is denoted by τ. The main notations in this section are listed in Table 1.

2.2. Problem Formulation

2.2.1. Service Request Scheduling. In each time slot $t \in \{0, 1, \ldots, T-1\}$, a number of service requests for the m types of services are offloaded. Let $A_i(t)$ be the number of requests for service i offloaded to the MEC systems in time slot t. In our article, we require no prior knowledge of the statistics of $A_i(t)$, which is generally hard to obtain or precisely predict in real-life. $a_{ij}(t)$ represents the number of requests for service i that are scheduled to the edge server in the j-th MEC system in time slot t. $a_{ij}(t)$ is the request scheduling control variable. It should be satisfied that

$$\sum_{j \in J} a_{ij}(t) = A_i(t), \quad \forall i \in I. \tag{1}$$

The request scheduling method in our article will make use of the diversity of different MEC systems to provide service in order to reduce scheduling cost while providing performance guarantees.

2.2.2. Scheduling Cost. Let $\gamma_{ij}(t)$ be the unit cost of scheduling requests for service i to the j-th MEC system. $\gamma_{ij}(t)$ can be different among different services i and different MEC systems j. It can also vary across time for other factors such as traffic, wireless fading, the available resources, etc. The request scheduling cost of service i in time slot t can be calculated as $\sum_{j \in J} \gamma_{ij}(t) a_{ij}(t)$. The total scheduling cost for all the services can be expressed as

$$g(t) = \sum_{i \in I} \sum_{j \in J} \gamma_{ij}(t)\, a_{ij}(t). \tag{2}$$

Instead of studying the instantaneous scheduling cost, we focus on the long-term average cost. The time-average scheduling cost across time slots $t \in \{0, 1, \ldots, T-1\}$ can be expressed as

$$g = \lim_{T \to \infty} \frac{1}{T} \sum_{t=0}^{T-1} \mathbf{E}\{g(t)\}. \tag{3}$$

g is the minimization objective of the request scheduling problem in this article.

2.2.3. Performance. Queueing delay is one of the most important performance metrics. According to *Little's Law*, queueing delay is in proportion to the number of requests waiting in the queue. Thus, we seek to reduce queue length and maintain low congestion states. Let $Q_{ij}(t)$ represent the queue length of service i in the j-th MEC system in time slot t. $b_{ij}(t)$ denotes the number of requests for service i that can be served by the j-th MEC system. Thus, the queue length $Q_{ij}(t)$ evolves as

$$Q_{ij}(t+1) = \max\left[Q_{ij}(t) - b_{ij}(t), 0\right] + a_{ij}(t). \tag{4}$$

To reduce queueing delay and maintain system stability, we seek to bound the average queue length. Let the time-average queue length across the $t \in \{0, 1, \ldots, T-1\}$ slots represented by q_{ij}. The service request scheduling method in this article bounds the average queue length as

$$q_{ij} = \lim_{T \to \infty} \frac{1}{T} \sum_{t=0}^{T-1} \mathbf{E}\left\{Q_{ij}(t)\right\} < \zeta, \quad \exists \zeta \in \mathbb{R}^+. \tag{5}$$

Table 1: Notations and definitions.

Notation	Definition
I	Services set.
J	MEC systems set.
$A_i(t)$	Number of requests for service i in time slot t.
$a_{ij}(t)$	Number of requests for service i that are scheduled to the j-th MEC system in time slot t.
$b_{ij}(t)$	Number of requests for service i that can be served by the j-th MEC system in time slot t.
$\gamma_{ij}(t)$	Unit cost of scheduling requests for service i to the j-th MEC system.
$Q_{ij}(t)$	Queue length of service i on the j-th MEC system in time slot t.
$g(t)$	Scheduling cost for all the services in time slot t.

2.2.4. Unified Framework. To combine scheduling cost and performance, the request scheduling problem in this article is formulated as

$$\text{minimize} \quad g = \lim_{T\longrightarrow\infty} \frac{1}{T} \sum_{t=0}^{T-1} \mathbf{E}\left\{g\left(t\right)\right\}; \tag{6}$$

subject to constraints (1), (5).

Solving problem (6) offline requires the future information (such as requests arrival information, scheduling cost information) which is generally hard to obtain or precisely predict in practice. Thus, we propose an online Dynamic Service Request Scheduling algorithm to solve the problem, which will be shown in Section 3.

3. Dynamic Request Scheduling Algorithm Design

In this section, based on the Lyapunov optimization framework [28], we decompose the original optimization problem into a series of independent subproblems. Then, we design a Dynamic Service Request Scheduling algorithm to solve these subproblems in a distribute way.

3.1. Problem Transformation Using Lyapunov techniques. Based on Lyanuov optimization techniques, we define $\Theta(t) = (Q_{ij}(t))$ as the queue length matrix of the MEC systems. Then, we denote $L(\Theta(t))$ as the Lyanuov function as follows, which is a scalar measure of the queue congestion state in the system,

$$L\left(\Theta\left(t\right)\right) = \frac{1}{2} \sum_{i\in I} \sum_{j\in J} Q_{ij}^2\left(t\right). \tag{7}$$

A small value of $L(\Theta(t))$ indicates that the queue lengths of all MECs are small, which represents a low congestion state of the MEC systems according to the *Little's Law*. In order to reduce queue length and maintain system stability, we seek to keep the Lyanuov function at a small value. Then, we define the *conditional Lyapunov drift* $\Delta(\Theta(t))$,

$$\Delta\left(\Theta\left(t\right)\right) = \mathbf{E}\left\{L\left(\Theta\left(t+1\right)\right) - L\left(\Theta\left(t\right)\right) \mid \Theta\left(t\right)\right\}. \tag{8}$$

By reducing the value of $\Delta(\Theta(t))$, we can push the Lyanuov function towards to a small value. To integrate scheduling cost and queue length in the MEC systems, we define the *drift plus cost* according to Lyapunov optimization framework, which is expressed as

$$\Delta\left(\Theta\left(t\right)\right) + V\mathbf{E}\left\{g\left(t\right) \mid \Theta\left(t\right)\right\}. \tag{9}$$

The parameter V can be considered as the tradeoff parameter between the scheduling cost and queue length, which can be determined by service providers or users according to their requirements in real applications. Next in Theorem 1, we show that the *drift plus cost* is upper bounded if the service arrival rate can be upper bounded.

Theorem 1 (bounding drift plus cost). *In each time slot t, under any algorithm, for all possible values of $\Theta(t)$ and any parameter value of V, if there exits a peak value A_i^{max} that upper bounds the number of requests arrived in each time slot, the drift plus cost can be upper bounded by*

$$\begin{aligned}
&\Delta\left(\Theta\left(t\right)\right) + V\mathbf{E}\left\{g\left(t\right) \mid \Theta\left(t\right)\right\} \\
&\quad\le B + \sum_{i\in I} \sum_{j\in J} Q_{ij}\left(t\right) \mathbf{E}\left\{a_{ij}\left(t\right) - b_{ij}\left(t\right) \mid \Theta\left(t\right)\right\} \\
&\qquad + V \sum_{i\in I} \sum_{j\in J} \mathbf{E}\left\{a_{ij}\left(t\right)\gamma_{ij}\left(t\right) \mid \Theta\left(t\right)\right\},
\end{aligned} \tag{10}$$

where $B = (1/2)[\sum_{i\in I}(A_i^{max})^2 + \sum_{i\in I}\sum_{j\in J}\widehat{b}_{ij}^2]$ is a constant.

Proof. By squaring the both sides of (4) and applying the inequality that $(\max[Q_{ij}(t) - b_{ij}(t), 0])^2 \le (Q_{ij}(t) - b_{ij}(t))^2$, we have

$$\begin{aligned}
Q_{ij}^2\left(t+1\right) \le{}& \left(Q_{ij}\left(t\right) - b_{ij}\left(t\right)\right)^2 + a_{ij}^2\left(t\right) \\
&+ 2a_{ij}\left(t\right) \max\left[Q_{ij}\left(t\right) - b_{ij}\left(t\right), 0\right].
\end{aligned} \tag{11}$$

Then, we define $\overline{b}_{ij}(t)$ as the actual number of requests for service i served by the j-th MEC system in time slot t,

$$\overline{b}_{ij}\left(t\right) = \begin{cases} b_{ij}\left(t\right), & b_{ij}\left(t\right) \le Q_{ij}\left(t\right) \\ Q_{ij}\left(t\right), & otherwise. \end{cases} \tag{12}$$

We can obtain that $\max[Q_{ij}(t) - b_{ij}(t), 0] = Q_{ij}(t) - \overline{b}_{ij}(t)$ and rewrite (11) as follows:

$$\begin{aligned} Q_{ij}^2(t+1) \le{}& Q_{ij}^2(t) + a_{ij}^2(t) + b_{ij}^2(t) \\ &+ 2Q_{ij}(t)\left(a_{ij}(t) - b_{ij}(t)\right) \\ &- 2a_{ij}(t)\,\overline{b}_{ij}(t). \end{aligned} \tag{13}$$

Because $a_{ij}(t)\overline{b}_{ij}(t) \ge 0$, we have

$$\begin{aligned} &\frac{1}{2}\left[Q_{ij}^2(t+1) - Q_{ij}^2(t)\right] \\ &\quad\le \frac{1}{2}\left[a_{ij}^2(t) + b_{ij}^2(t)\right] + Q_{ij}(t)\left[a_{ij}(t) - b_{ij}(t)\right]. \end{aligned} \tag{14}$$

Taking the expectations on the condition of $\Theta(t)$ to both sides in (14) and summing over $i \in I$ and $j \in J$, it can be obtained that

$$\begin{aligned} \Delta(\Theta(t)) \le{}& \frac{1}{2}\sum_{i\in I}\sum_{j\in J}\mathbf{E}\left\{a_{ij}^2(t) + b_{ij}^2(t) \mid \Theta(t)\right\} \\ &+ \sum_{i\in I}\sum_{j\in J}Q_{ij}(t)\,\mathbf{E}\left\{a_{ij}(t) - b_{ij}(t) \mid \Theta(t)\right\}. \end{aligned} \tag{15}$$

Since it holds that $\sum_{j\in J} a_{ij}(t) = A_i(t)$ and $A_i(t) \le A_i^{max}$, we have

$$\sum_{j\in J}\mathbf{E}\left\{a_{ij}^2(t) \mid \Theta(t)\right\} \le \mathbf{E}\left\{A_i^2(t) \mid \Theta(t)\right\} \le \left(A_i^{max}\right)^2. \tag{16}$$

In addition, we define $\widehat{b}_{ij}$ as the upper bound of $b_{ij}(t)$ over all the time slots. We can obtain that

$$\begin{aligned} &\sum_{i\in I}\sum_{j\in J}\mathbf{E}\left\{a_{ij}^2(t) + b_{ij}^2(t) \mid \Theta(t)\right\} \\ &\quad\le \sum_{i\in I}\left(A_i^{max}\right)^2 + \sum_{i\in I}\sum_{j\in J}\widehat{b}_{ij}^2. \end{aligned} \tag{17}$$

By adding $V\mathbf{E}\{g(t) \mid \Theta(t)\}$ to both sides and letting B take the value of $(1/2)[\sum_{i\in I}(A_i^{max})^2] + \sum_{i\in I}\sum_{j\in J}\widehat{b}_{ij}^2$, it can be obtained that

$$\begin{aligned} &\Delta(\Theta(t)) + V\mathbf{E}\left\{g(t) \mid \Theta(t)\right\} \\ &\quad\le B + V\mathbf{E}\left\{g(t) \mid \Theta(t)\right\} \\ &\qquad+ \sum_{i\in I}\sum_{j\in J}Q_{ij}(t)\,\mathbf{E}\left\{a_{ij}(t) - b_{ij}(t) \mid \Theta(t)\right\}. \end{aligned} \tag{18}$$

Substituting (2) into the right-hand-side (R.H.S.) of (18), we can obtain (10). □

3.2. Dynamic Request Scheduling Algorithm. Following the design principles of Lyapunov optimization techniques, we design an efficient Dynamic Service Request Scheduling (DSRS) algorithm to minimize the upper bound of *drift plus cost* in each time slot t. By decomposing the minimization of upper bound problem into a series of independent subproblems, our DSRS algorithm optimizes the average scheduling cost concurrently in a distributed way. In addition, it will be proven that DSRS algorithm can achieve a long-term time-average scheduling cost that is arbitrarily close to the optimal value while maintaining the stability of the MEC systems.

In each time slot t, based on the current queue length matrix $\Theta(t)$ of the MEC systems, the DSRS algorithm makes request scheduling decisions $a_{ij}(t)$ to minimize the upper bound of R.H.S. of (10). Since B and b_{ij} can be considered as constant in the optimization problem, we can rewrite the minimization of upper bound as

$$\min_{a_{ij}(t)} \quad \sum_{i\in I}\sum_{j\in J}a_{ij}(t)\,Q_{ij}(t) + Va_{ij}(t)\,\gamma_{ij}(t). \tag{19}$$

subject to

$$\sum_{j\in J}a_{ij}(t) = A_i(t), \quad \forall i \in I. \tag{20}$$

As the request scheduling decisions $a_{ij}(t)$ are independent among different services, the above centralized minimization problem (19) can be decomposed into the following subproblem (21) for each service $i \in I$, i.e.,

$$\min_{a_{ij}(t)} \quad \sum_{j\in J}a_{ij}(t)\left(Q_{ij}(t) + V\gamma_{ij}(t)\right). \tag{21}$$

subject to

$$\sum_{j\in J}a_{ij}(t) = A_i(t). \tag{22}$$

Problem (21) can be regarded as a generalized min-weight problem, where the number of requests scheduled to the MEC systems is weighted by the value of $Q_{ij}(t) + V\gamma_{ij}(t)$. Therefore, for each service $i \in I$, the optimal solution is to schedule all the requests to the MEC system with the minimum value of $Q_{ij}(t) + V\gamma_{ij}(t)$; i.e.,

$$a_{ij}(t) = \begin{cases} A_i(t), & j = j^* \\ 0, & otherwise \end{cases} \tag{23}$$

where $j^* \in \operatorname{argmin}(Q_{ij}(t) + V\gamma_{ij}(t))$ for all $j \in J$.

Remark. There exist tradeoffs between the scheduling cost and queue length of the MEC systems. Scheduling all the service requests to the MEC system with low cost can reduce the overall scheduling cost; however, the queue length of the MEC system can be very large. The DSRS algorithm combines scheduling cost and queue length, and $Q_{ij}(t) + V\gamma_{ij}(t)$ can be regarded as the penalty factor for each MEC system. Recall that V represents the tradeoff between scheduling cost and queue length. The intuition of the optimal scheduling policy obtained by the DSRS algorithm is to minimize the penalty function of the MEC systems in each time slot. In this way, the DSRS algorithm can reduce both the scheduling cost and the queue length. In addition, by changing the value of V, the

1: In the beginning of each time slot t, observe the current queue length $Q_{ij}(t)$.
2: **for all** $i \in I$ **do**
3: Set auxiliary variable $MinWeight = -\infty$;
4: Set $j^* = 1$;
5: **for all** $j \in J$ **do**
6: Calculate the penalty factor $pf_j = Q_{ij}(t) + V\gamma_{ij}(t)$;
7: **if** $pf_j < MinWeight$ **then**
8: $j^* = j$;
9: **end if**
10: **end for**
11: **for all** $j \in J$ **do**
12: **if** $j == j^*$ **then**
13: Set $a_{ij}(t) = A_i(t)$;
14: **else**
15: Set $a_{ij}(t) = 0$;
16: **end if**
17: **end for**
18: **end for**

Algorithm 1: Dynamic Service Request Scheduling (DSRS).

DSRS algorithm can achieve the arbitrary tradeoff between scheduling cost and queue length.

After the scheduling decisions $a_{ij}(t)$ are determined, the queue length $Q_{ij}(t)$ updates according to (4). The detailed algorithm is shown in Algorithm 1.

4. Algorithm Analysis

In this section, we present mathematical analysis of the boundary of the time-average queue length and scheduling cost of our DSRS algorithm. It can be proven that our algorithm can achieve the scheduling cost arbitrarily close to the optimal value while maintaining the stability of the MEC systems. Let $\overline{Q}$ denote the long-term time-average queue length,

$$\overline{Q} = \lim_{T \longrightarrow \infty} \frac{1}{T} \sum_{t=0}^{T-1} \sum_{i \in I} \sum_{j \in J} \mathbf{E}\left\{Q_{ij}(t)\right\}. \tag{24}$$

We present in Lemma 2 that if the arrival $A_i(t)$ is independent and identically distributed (i.i.d.) over time slots, there exists a randomized policy π^* which can achieve the minimum cost g^* defined in (3), where the control decision $a_{ij}(t)$ follows certain fixed probability distribution independent of the queue length matrix $\Theta(t)$.

Lemma 2. *For any service request arrival rate $\lambda \in \Lambda$, where Λ is the capacity region of the system, if the arrival $A_i(t)$ is i.i.d. over time slots, there exists a randomized policy π^* that determines the control decision $a_{ij}(t)$ in each time slot t and achieves the following:*

$$\begin{aligned} &\mathbf{E}\left\{g^{\pi^*}(t)\right\} = g^*(\lambda); \\ &\mathbf{E}\left\{\sum_{j \in J} a_{ij}^{\pi^*}(t)\right\} \le \mathbf{E}\left\{\sum_{j \in J} b_{ij}(t)\right\}. \end{aligned} \tag{25}$$

where $g^*(\lambda)$ denotes the minimum time-average cost under the arrival rate λ.

Proof. Lemma 2 can be proven by Caratheodory's theorem in [28], we omit the detailed proof here for simplicity and brevity. □

Since it is assumed that there exists upper bound A_i^{max} of the service request arrival rate, there also exists upper bound $\hat{g}$ and lower bound $\check{g}$ of the objective g. Then, we derive the boundary of queue length and scheduling cost of the DSRS algorithm based on Lemma 2.

Theorem 3. *Assume that there exists ε satisfying $\lambda + \varepsilon \in \Lambda$, then, under our DSRS algorithm, for any value of the parameter V, the time-average queue length defined in (24) is bounded as*

$$\overline{Q} \le \frac{B + V(\hat{g} - \check{g})}{\varepsilon}. \tag{26}$$

Furthermore, the time-average system scheduling cost can be bounded by (27), which shows the cost derived by our DSRS algorithm can approach the optimal value by increasing the parameter V. Here, B is the constant defined in Theorem 1.

$$g^{DSRS} \le g^* + \frac{B}{V}. \tag{27}$$

Proof. Since it holds that $\lambda + \varepsilon \in \Lambda$, we can obtain that there exists a randomized policy π' which satisfies (28) and (29) according to Lemma 2.

$$\mathbf{E}\left\{g^{\pi'}(t)\right\} = g^*(\lambda + \varepsilon); \tag{28}$$

$$\mathbf{E}\left\{\sum_{j \in J} a_{ij}^{\pi'}(t)\right\} \le \mathbf{E}\left\{\sum_{j \in J} b_{ij}(t)\right\} - \varepsilon. \tag{29}$$

As our DSRS algorithm can achieve the minimum value of the R.H.S of (10) among all feasible policies (including policy π'), it can be obtained that

$$\begin{aligned}&\Delta(\Theta(t)) + V\mathbf{E}\{g(t) \mid \Theta(t)\}\\ &\le B + V\mathbf{E}\left\{g^{\pi'(t)} \mid \Theta(t)\right\}\\ &\quad + \sum_{i\in I}\sum_{j\in J} Q_{ij}(t)\,\mathbf{E}\left\{a_{ij}^{\pi'}(t) - b_{ij}(t) \mid \Theta(t)\right\}.\end{aligned} \tag{30}$$

Substituting (28) and (29) into the R.H.S. of (30), taking expectations on both sides, and then using iterated expectations, we can yield

$$\begin{aligned}&\mathbf{E}\{L(\Theta(t+1)) - L(\Theta(t))\} + V\mathbf{E}\{g(t)\}\\ &\le B + Vg^*(\lambda+\varepsilon) - \varepsilon\sum_{i\in I}\sum_{j\in J}\mathbf{E}\left\{Q_{ij}(t)\right\}.\end{aligned} \tag{31}$$

Moving $V\mathbf{E}\{g(t)\}$ to the R.H.S. of (31), it can be obtained that

$$\begin{aligned}&\mathbf{E}\{L(\Theta(t+1)) - L(\Theta(t))\}\\ &\le B + V\left(g^*(\lambda+\varepsilon) - \mathbf{E}\{g(t)\}\right)\\ &\quad - \varepsilon\sum_{i\in I}\sum_{j\in J}\mathbf{E}\left\{Q_{ij}(t)\right\}\\ &\le B + V\left(\widehat{g} - \check{g}\right) - \varepsilon\sum_{i\in I}\sum_{j\in J}\mathbf{E}\left\{Q_{ij}(t)\right\}.\end{aligned} \tag{32}$$

To be general, we assume the queue length is empty when $t = 0$. By summing both sides of (32) over $t \in \{0, 1, \ldots T-1\}$ and applying the fact that $L(\Theta(t)) \ge 0$, we can obtain

$$\begin{aligned}\varepsilon\sum_{t=0}^{T-1}\sum_{i\in I}\sum_{j\in J}\mathbf{E}\left\{Q_{ij}(t)\right\} &\le \left(B + V\left(\widehat{g} - \check{g}\right)\right)T\\ &\quad - \mathbf{E}\{L(\Theta(T))\}\\ &\le \left(B + V\left(\widehat{g} - \check{g}\right)\right)T.\end{aligned} \tag{33}$$

Dividing both sides of (33) by εT and taking a lim as $T \longrightarrow \infty$ yield (26).

By summing both sides of (31) over $t \in \{0, 1, \ldots T-1\}$ and applying the fact that $\mathbf{E}\{Q_{ij}(t)\} \ge 0$, it can be obtained

$$V\sum_{t=0}^{T-1}\mathbf{E}\{g(t)\} \le \left(Vg^*(\lambda+\varepsilon) + B\right)T. \tag{34}$$

Dividing both sides of (34) by VT, we have

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbf{E}\{g(t)\} \le g^*(\lambda+\varepsilon) + \frac{B}{V}. \tag{35}$$

Taking a lim of (35) as $T \longrightarrow \infty$, applying Lebesgue's dominated convergence theorem, and letting $\varepsilon \longrightarrow 0$ yield (27). □

Remark. Theorem 3 shows that our DSRS algorithm can achieve a $[O(1/V), O(V)]$ tradeoff between the time-average scheduling cost and queue length. According to (27), the gap between the time-average scheduling cost obtained by our DSRS algorithm and the optimal value is within $O(1/V)$. By setting the value of V sufficiently large, the DSRS algorithm can approach the optimal scheduling cost. However, a large V will cause a large queue backlog of the MEC systems. Nevertheless, the queue length obtained by our DSRS algorithm is also bounded according to (26). And constraint (5) can be satisfied by letting ζ take the value of $(B + V(\widehat{g} - \check{g}))/\varepsilon$.

Then, we analyze the time complexity of the DSRS algorithm. According to Algorithm 1, for the two inner loops (line 5-10 and line 11-17), DSRS algorithm traverses each edge server once. Therefore, each loop terminates in $O(n)$ operations, where n is the number of edge servers. For the outer loop (line 1-18), since the request scheduling of different service applications is independent, it terminates in $O(n)$ operations. Thus, the time complexity of the DSRS algorithm is $O(n)$.

5. Evaluation

In this section, we conduct experiments to evaluate our DSRS algorithm. First, we analyze the impact of parameters. Then, we present comparison experiments which show the effectiveness of our DSRS algorithm.

In the experiments, we consider 4 MEC systems, each with an edge server providing services for the offloaded requests. There are two types of heterogeneous services. For each service $i \in I$, the request arrival process is generated according to Poisson distribution with arrival rate λ_i [29]. Note that the DSRS algorithm actually requires no knowledge of the statistical information of request arrivals. The computing capacity of the MEC systems is set as $\beta_i \cdot \lambda_i$ where $\beta_i > 1$. Without loss of generality, we assume the MEC systems are heterogeneous with different computing capacities. And the unit scheduling costs of different MEC systems are set to be positively related to its computing capacity.

5.1. Parameter Analysis

5.1.1. Effect of Tradeoff Parameter. Figures 1 and 2 show the time-average scheduling cost and queue length of the MEC systems with different values of V. In Figure 1, it can be seen that the scheduling cost decreases as the value of V increases, which is in accordance with (27) in Theorem 3. This is because as V increases, more weight is put on scheduling cost, and the DSRS algorithm would schedule more service requests to the MEC system with lower unit cost in order to reduce the overall scheduling cost. However, Figure 2 shows that the queue length also rises with the increase of V, which is consistent with (26) in Theorem 3. Nevertheless, the queue length would stabilize gradually with far more increase of V. Together with Figures 1 and 2, we can see that the DSRS algorithm can make a tradeoff between scheduling cost and queue length by adjusting the value of V.

5.1.2. Effect of Service Request Arrival Rate. We analyze the effect of service request arrival rate on the scheduling cost and

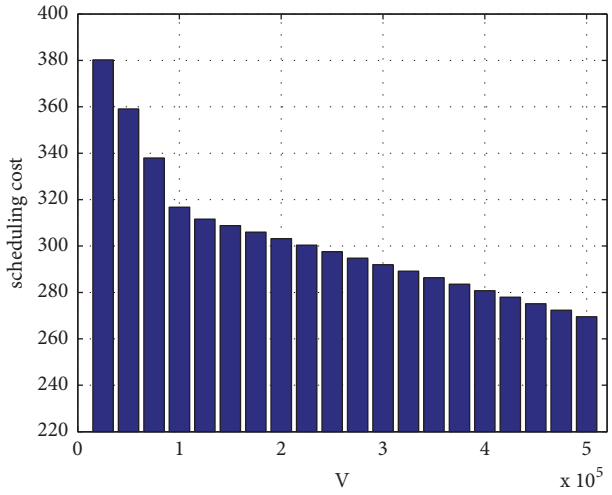


Figure 1: Scheduling cost with different values of V.

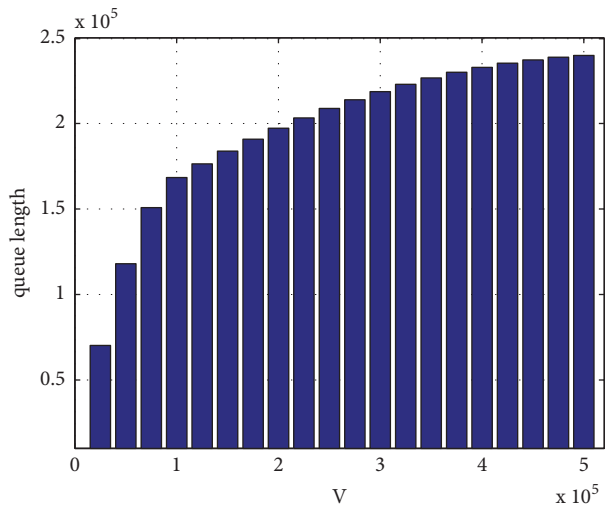


Figure 2: Queue length with different values of V.

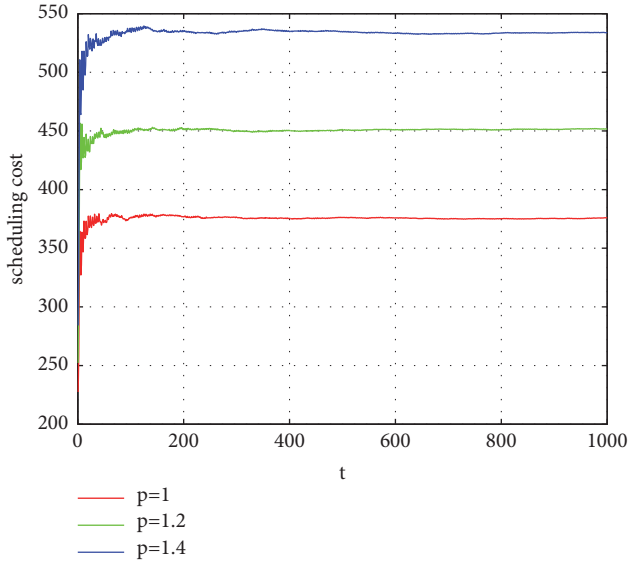


Figure 3: Scheduling cost with different arrival rates.

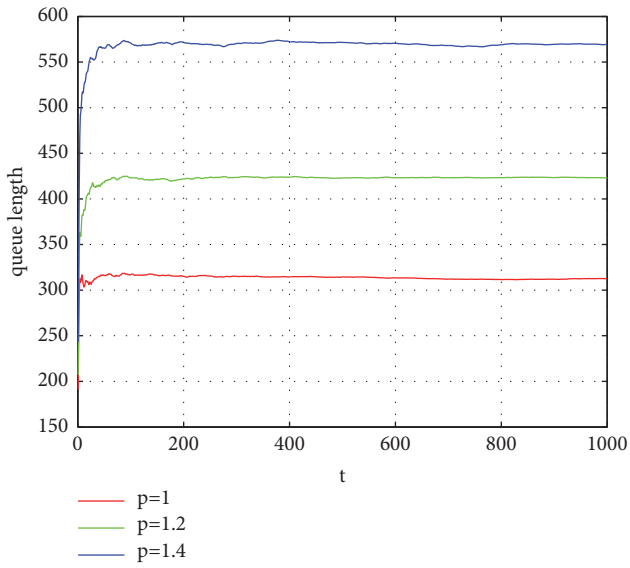


Figure 4: Queue length with different arrival rates.

queue length. In the experiments, for each application $i \in I$, we scale the service request arrival rate up or down to $p \cdot \lambda_i$. We consider three different cases, where $p = 1$, 1.2 and 1.4, respectively. Figures 3 and 4 show that both of the scheduling cost and queue length increase as the request arrival rate increases. Nevertheless, the queue length can stabilize quickly with the increase of service request arrival rate. This shows that our DSRS algorithm can dynamically adjust the request scheduling decisions according to different service request arrivals and maintain the stability of the MEC systems.

5.1.3. Effect of Unit Scheduling Cost. To analyze the effect of unit scheduling cost on the MEC systems, we scale the unit scheduling cost up or down to $q \cdot \gamma_{ij}$. We consider three different cases, where $q = 1$, 1.2 and 1.4, respectively. We can see from Figure 5 that the overall scheduling cost rises as the unit scheduling cost increases, since the scheduling cost of each request increases. In Figure 6, we can see that the queue length of the MEC systems also increases with the increase of unit scheduling cost. The reason is that our DSRS algorithm tries to achieve low scheduling cost by scheduling more requests to the MEC with smaller unit scheduling cost. However, this will lead to the larger queue backlog in some MEC systems.

5.2. Comparison Experiment. We conduct comparison experiment and compare our DSRS algorithm with Randomized algorithm to evaluate the effectiveness of the DSRS algorithm. The Randomized algorithm schedules all the service requests to each MEC system randomly. The scheduling costs and queue lengths of the two algorithms are shown in Figures 7 and 8, respectively.

We can see from Figure 7 that the scheduling cost of our DSRS algorithm is smaller than that of Randomized

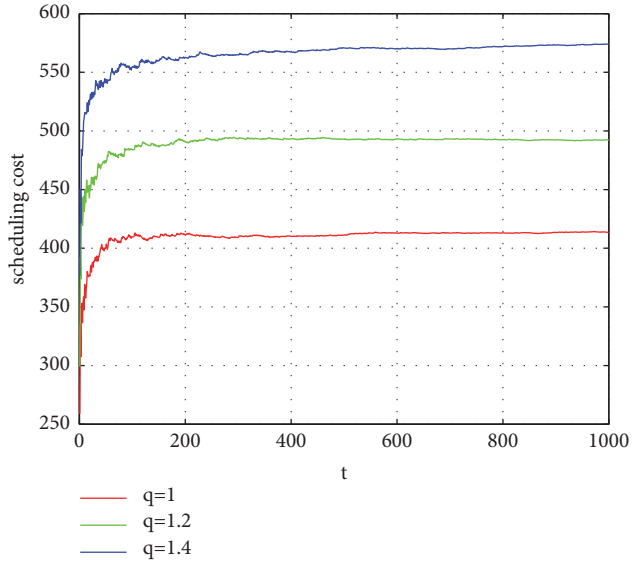


Figure 5: Scheduling cost with different unit scheduling costs.

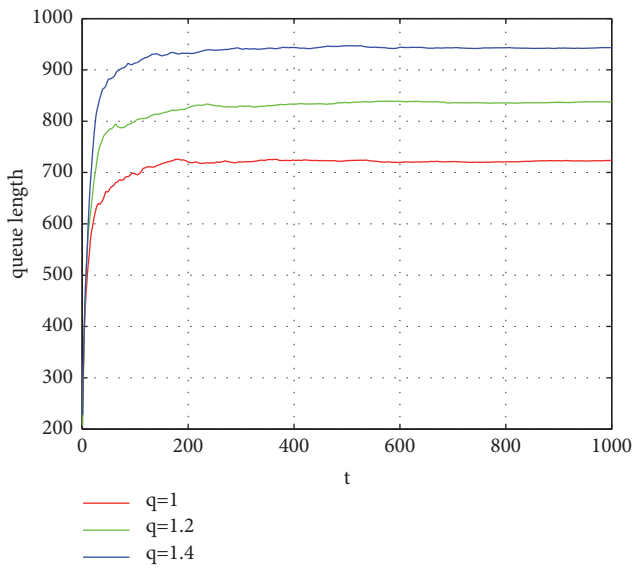


Figure 6: Queue length with different unit scheduling costs.

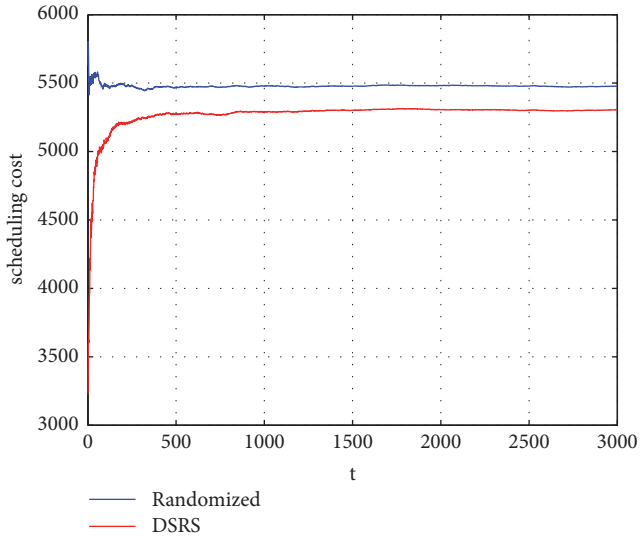


Figure 7: Scheduling cost under different algorithms.

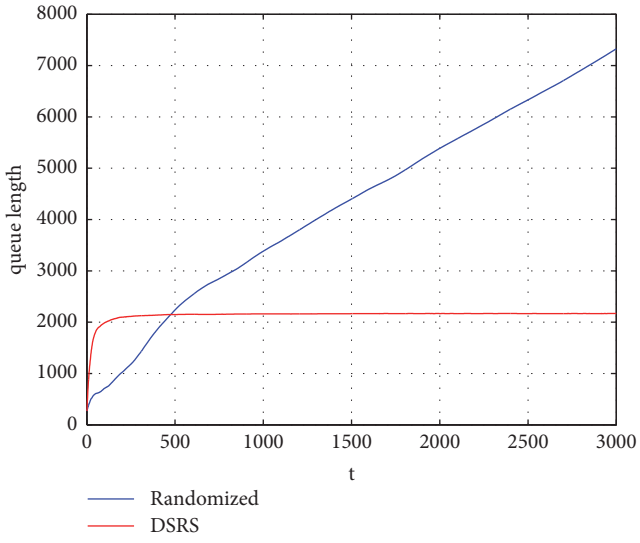


Figure 8: Queue length under different algorithms.

algorithm, which shows the effectiveness of our DSRS algorithm in reducing cost. In Figure 8, we can observe that the queue length of the Randomized algorithm is slightly smaller than our DSRS algorithm at the very beginning. However, as time goes by, the queue length of the Randomized algorithm increases continuously along with the time. The queue length of our DSRS algorithm stabilizes quickly and maintains at a small level. The reason is that our DSRS algorithm can adjust scheduling decisions dynamically according to the current queue backlog and maintain low congestion state in the MEC systems. Together with Figures 7 and 8, we can see the effectiveness of our DSRS algorithm in optimizing both scheduling cost and queue length.

6. Conclusion

In this article, we study dynamic request scheduling for MEC systems. We formulate it as an optimization problem, and the goal is to optimize scheduling cost while providing performance guarantee. We propose the DSRS algorithm to solve the optimization problem, which transforms it to a series of subproblems and solves each one efficiently in a distributed way. Mathematical analysis is presented which demonstrates that the DSRS algorithm can approach the optimal scheduling cost while bounding the queue length. Parameter analysis experiments and comparison experiments are both conducted to verify the effectiveness of the DSRS algorithm.

Data Availability

Most of the simulation experimental data used for supporting the study of this article are included within the article. Further additional information about the data is available from the corresponding author.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (no. 61370065 and no. 61502040), the Key Research and Cultivation Projects at Beijing Information Science and Technology University (no. 5211823411), Beijing Municipal Program for Excellent Teacher Promotion (no. PXM2017 014224 000028), and the Supplementary and Supportive Project for Teachers at Beijing Information Science and Technology University (no. 5111823401).

References

[1] Q. Ye and W. Zhuang, "Distributed and Adaptive Medium Access Control for Internet-of-Things-Enabled Mobile Networks," *IEEE Internet of Things Journal*, vol. 4, no. 2, pp. 446–460, 2017.

[2] M. Jia, J. Cao, and W. Liang, "Optimal cloudlet placement and user to cloudlet allocation in wireless metropolitan area networks," *IEEE Transactions on Cloud Computing*, vol. 5, no. 4, pp. 755-737, 2017.

[3] L. Tong, Y. Li, and W. Gao, "A hierarchical edge cloud architecture for mobile computing," in *Proceedings of the 35th Annual IEEE International Conference on Computer Communications, IEEE INFOCOM 2016*, pp. 1–9, April 2016.

[4] B.-G. Chun, S. Ihm, P. Maniatis, and eta, "Clonecloud: Elastic execution between mobile device and cloud," in *Proceedings of the 6th ACM EuroSys Conference on Computer Systems (EuroSys '11)*, pp. 301–314, ACM, New York, NY, USA, April 2011.

[5] S. Kosta, A. Aucinas, P. Hui, R. Mortier, and X. Zhang, "Thinkair: dynamic resource allocation and parallel execution in the cloud for mobile code offloading," in *Proceedings of the IEEE INFOCOM*, pp. 945–953, March 2012.

[6] M. S. Gordon, D. A. Jamshidi, S. Mahlke, Z. M. Mao, and X. Chen, "Comet: Code offload by migrating execution transparently," in *Proceedings of the Proceedings of the 10th USENIX Conference on Operating Systems Design and Implementation*, pp. 93–106, Berkeley, Calif, USA, 2012.

[7] J. Kwak, Y. Kim, J. Lee, and S. Chong, "DREAM: Dynamic Resource and Task Allocation for Energy Minimization in Mobile Cloud Systems," *IEEE Journal on Selected Areas in Communications*, vol. 33, no. 12, pp. 2510–2523, 2015.

[8] 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.

[9] X. Lyu, W. Ni, H. Tian et al., "Optimal schedule of mobile edge computing for internet of things using partial information," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 11, pp. 2606–2615, 2017.

[10] J. Liu, S. Wang, A. Zhou, F. Yang, and R. Buyya, "Availability-aware Virtual Cluster Allocation in Bandwidth-Constrained Datacenters," in *Proceedings of the IEEE Transactions on Services Computing*, pp. 1-1, 2017.

[11] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: vision and challenges," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637–646, 2016.

[12] Y. Yu, J. Zhang, and K. B. Letaief, "Joint subcarrier and CPU time allocation for mobile edge computing," in *Proceedings of the IEEE Global Communications Conference (GLOBECOM)*, pp. 1–6, December 2016.

[13] Y. Mao, J. Zhang, S. H. Song, and K. B. Letaief, "Stochastic joint radio and computational resource management for multi-user mobile-edge computing systems," *IEEE Transactions on Wireless Communications*, vol. 16, no. 9, pp. 5994–6009, 2017.

[14] S. Wang, J. Xu, N. Zhang, and Y. Liu, "A Survey on Service Migration in Mobile Edge Computing," *IEEE Access*, vol. 6, pp. 23511–23528, 2018.

[15] Y. Kim, J. Kwak, and S. Chong, "Dual-Side Optimization for Cost-Delay Tradeoff in Mobile Edge Computing," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 2, pp. 1765–1781, 2018.

[16] Y. Mao, J. Zhang, and K. B. Letaief, "Dynamic Computation Offloading for Mobile-Edge Computing with Energy Harvesting Devices," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 12, pp. 3590–3605, 2016.

[17] S. Wang, Y. Zhao, L. Huang, J. Xu, and C.-H. Hsu, "QoS prediction for service recommendations in mobile edge computing," *Journal of Parallel and Distributed Computing*, 2017.

[18] J. Zheng, Y. Cai, Y. Wu, and X. S. Shen, "Dynamic computation offloading for mobile cloud computing: A stochastic game-theoretic approach," in *Proceedings of the IEEE Transactions on Mobile Computing*, pp. 1-1, 2018.

[19] C. You, K. Huang, H. Chae, and B.-H. Kim, "Energy-Efficient Resource Allocation for Mobile-Edge Computation Offloading," *IEEE Transactions on Wireless Communications*, vol. 16, no. 3, pp. 1397–1411, 2017.

[20] H. Tan, Z. Han, X. Li, and F. C. Lau, "Online job dispatching and scheduling in edge-clouds," in *Proceedings of the IEEE INFOCOM 2017 - IEEE Conference on Computer Communications*, pp. 1–9, Atlanta, Ga, USA, May 2017.

[21] S. Deng, L. Huang, D. Hu, J. L. Zhao, and Z. Wu, "Mobility-Enabled Service Selection for Composite Services," *IEEE Transactions on Services Computing*, vol. 9, no. 3, pp. 394–407, 2016.

[22] Q. Ye and W. Zhuang, "Token-Based Adaptive MAC for a Two-Hop Internet-of-Things Enabled MANET," *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1739–1753, 2017.

[23] R. Urgaonkar, S. Wang, T. He, M. Zafer, K. Chan, and K. K. Leung, "Dynamic service migration and workload scheduling in edge-clouds," *Performance Evaluation*, vol. 91, pp. 205–228, 2015.

[24] Y. Nan, W. Li, W. Bao et al., "Adaptive Energy-Aware Computation Offloading for Cloud of Things Systems," *IEEE Access*, vol. 5, pp. 23947–23957, 2017.

[25] Z. Zhou, F. Liu, H. Jin, B. Li, B. Li, and H. Jiang, "On arbitrating the power-performance tradeoff in SaaS clouds," in *Proceedings of the 32nd IEEE Conference on Computer Communications, IEEE INFOCOM 2013*, pp. 872–880, Italy, April 2013.

[26] S. Ren, Y. He, and F. Xu, "Provably-efficient job scheduling for energy and fairness in geographically distributed data centers," in *Proceedings of the 32nd IEEE International Conference on*

Distributed Computing Systems, ICDCS 2012, pp. 22–31, China, June 2012.

[27] K. Ha, P. Pillai, W. Richter et al., "Just-in-time provisioning for cyber foraging," in *Proceedings of the 11th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys '13)*, pp. 153–166, New York, NY, USA, 2013.

[28] M. J. Neely, *Stochastic Network Optimization With Application to Communication and Queueing Systems*, Morgan and Claypool, 2010.

[29] E. Chlebus and J. Brazier, "Nonstationary Poisson modeling of web browsing session arrivals," *Information Processing Letters*, vol. 102, no. 5, pp. 187–190, 2007.

Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 5380481, 11 pages
https://doi.org/10.1155/2018/5380481

Research Article

A Security Situation Prediction Algorithm Based on HMM in Mobile Network

Wei Liang[1], **Jing Long**[2], **Zuo Chen**[2], **Xiaolong Yan**[2], **Yanbiao Li**[3], **Qingyong Zhang**[4] and **Kuan-Ching Li**[5]

[1]*School of Opto-Electronic and Communication Engineering, Xiamen University of Technology, Xiamen 361024, China*
[2]*College of Computer Science and Electronic Engineering, Hunan University, Changsha 410082, China*
[3]*Computer Science Department, University of California at Los Angeles, USA*
[4]*Fujian Key Laboratory of Automotive Electronics and Electric Drive (Fujian University of Technology), Fuzhou 350118, China*
[5]*Department of Computer Science and Information Engineering, Providence University, Taichung 43301, Taiwan*

Correspondence should be addressed to Kuan-Ching Li; kuancli@gm.pu.edu.tw

Received 18 April 2018; Accepted 1 August 2018; Published 14 August 2018

Academic Editor: Ao Zhou

The increasingly severe network security situation brings unanticipated challenges to mobile networking. Traditional HMM (Hidden Markov Model) based algorithms for predicting the network security are not accurate, and to address this issue, a weighted HMM based algorithm is proposed to predict the security situation of the mobile network. The multiscale entropy is used to address the low speed of data training in mobile network, whereas the parameters of HMM situation transition matrix are also optimized. Moreover, the autocorrelation coefficient can reasonably use the association between the characteristics of the historical data to predict future security situation. Experimental analysis on DARPA2000 shows that the proposed algorithm is highly competitive, with good performance in prediction speed and accuracy when compared to existing design.

1. Introduction

With the rapid development and popularization of Internet technology, mobile network becomes an indispensable communication mean [1]. Nevertheless, one of largest issues is security threat in mobile network [2–4]. People has developed many protection techniques to cope with these threats, such as mobile firewall, intrusion detection, and virus killer. Though these techniques are active and useful for already occurred threats but cannot monitor the complete situation of the mobile network. In this case, prediction technique of mobile network security situation has emerged as a trend of network security monitoring, which can rapidly acquire, understand, and display the security elements and use them to predict the future situation [5]. By analyzing the changes of mobile network security situation, network manager can predict the security situation and protect the network from illegal attacks.

In existing security situation prediction techniques, the warning is classified on the basis of the damage degree caused by network attacks and the risk level of the security vulnerability [6–8]. In such a case, the threats in mobile network can only be evaluated by the feature of attacks. Other factors such as environment are not investigated. Snort [9] classified the priority of warning into three levels. The warning of priority 1 illustrates a threat with the highest level, usually including security vulnerability with high risk, such as buffer overflow. The risk level of security vulnerability is analyzed by professional institutes, such as CVE (Common Vulnerability Exposure) [10] and Bugtraq [11], and classified into high, medium, and low. The classification result can also be obtained from a more complete quantified evaluation mechanism of vulnerability risk, namely, CVSS (Common Vulnerability Scoring System) [12].

The threat level of mobile network attack is closely related to the running condition and the vulnerability of target computer system. A typical example is worm virus Code Red II targeting at Linux computer systems. It has extremely low threat level since it is designed on the basis of Windows IIS

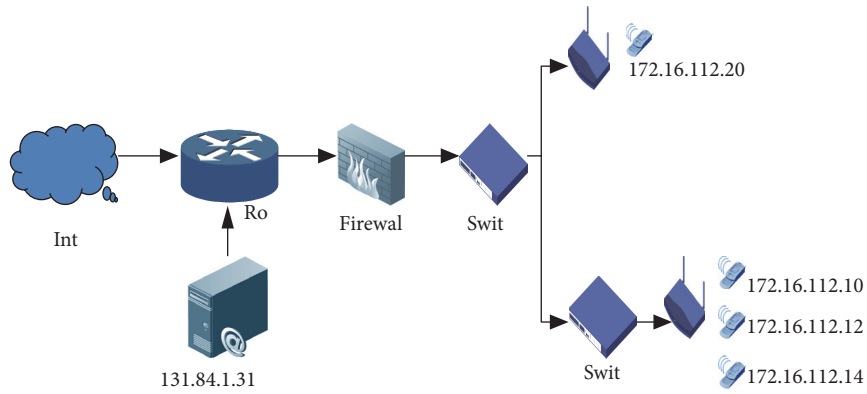


Figure 1: Topology structure of mobile network.

vulnerability, since there is no similar vulnerability in Linux systems, and therefore, it is not a threat. Many researchers proposed to evaluate the threat level by cross correlation of the mobile network warning and vulnerability scanning tool. Kruegel et al. [13] realized correlation analysis of Snort warning and Nessus [14] (a vulnerability scanning tool) and used it to confirm the warning. The target vulnerability of the attack is compared to the scanned vulnerability result of the target computer in terms of the port, service, vulnerability ID, etc. The matching result is used to confirm or filter the warning. If the target vulnerability of the attack exists in the target computer, the probability of a successful attack is larger, as well higher the threat level. Otherwise, the threat level is low, and the warning may be a false alarm. The proposed method has fully considered the network running state for the attack, by better recognizing false alarm and real threats. Nevertheless, the drawback is that it cannot evaluate the threats not targeting at vulnerability, since the information of the target vulnerability should be stored. Besides, the instantaneity of the vulnerability scanning report should be also considered.

The topology of a mobile network is depicted in Figure 1. "Mission" is the target service, port of the attack, which is determined by the manager. For example, HTTP service of 80 port on a Web server is a key Mission. The attack targeting at the Mission has the highest threat level. Porras et al. [15] calculated the probability of a successful attack using Bayesian network. Most of the SIM (Security Information Management) software provides a method to determine the warning level by making a security strategy. It is convenient for the manager to recognize the interesting attacks by custom rules and reducing the threat level of some of attacks.

2. Related Work

With the rapid development of mobile network, security situation prediction becomes an important network security technique, which plays a significant role in defending 802.11 wireless network threats. Due to its intrinsic features, the risk detection system can operate normally in wireless mobile network. There are increasing security threats of communication devices in mobile network, and therefore, it is urgent to evaluate and predict the security situation of mobile network. The network security report published by Chinese government in 2017 shows several security issues in mobile network. The Shadow Brokers leaked many 0-day vulnerabilities. There are serious security vulnerabilities in Bluetooth protocol, and a security research company in Internet-of-Thing (IoT) Armis has identified eight 0-day vulnerabilities in Bluetooth protocol. The company creates a group of network attack vectors for demonstration, where attackers can completely host the Bluetooth-supported network devices. Malicious attack software is propagated through these devices. A man-in-the-middle (MITM) connection is created, which makes lots of home cameras be intruded by Trojan, due to the fact that many intelligent cameras have weak password or vulnerabilities, which turns the main reason of camera intrusion. Security issues have arisen the concerns of government security departments in various countries, and a new generation of network security assessment prediction model emerges under this situation. The situation assessment of network communication security turned to be a hotspot in researches of network security.

In recent years, network intrusion detection systems have been replaced by security situation prediction techniques. Markov model has the nonaftereffect property and only related to the probability of its previous state. It is suitable to predict the random process which largely varies, as the future state is predicted by using the transfer probability matrix between states. The calculation of the transfer probability matrix uses statistical method and the prediction is to use the frequency to approximate the probability function. Therefore, the Markov model is suitable for security prediction of large data sample in mobile network communication. A typical Markov process includes Bernoulli process, Wiener process, and Poisson process. Based on continuous or discrete state and time parameter, Markov process can be classified into three categories: (1) discrete time discrete state Markov process, called Markov chain, (2) continuous time discrete state Markov process, called continuous time Markov chain, and (3) continuous time continuous state Markov process.

In the security prediction techniques for mobile network, Shake et al. proposed a network vulnerability evaluation method based on the electric current theory [16]. Bass

created a network situation framework using data fusion of multiple sensors. The network security situation is evaluated by deriving the identity of intruder, speed, threat level, and intrusion object [17]. There are various network attacks, which pretend to be normal data packet and perform attacks on the target computer, despite more difficult to detect the attacks. So, most of network attacks cannot be detected only be matching the features of data packet format. Meanwhile, the current detection techniques should unpack the packet, causing detection delay.

The intrusion techniques have been rapidly developed to cope with the increasing attacks in mobile network. To improve the detection accuracy, the commercial network intrusion detection system [18, 19] abstracted the characteristics of network attack using multiple algorithm and created a misuse detection model of network attacks. The network intrusion detection system based on misuse detection has high detection efficiency and low probability of false alarm. Since the detection system requires a real-time updated attack characteristic library, the characteristics of most network attacks are stored. If a new attack emerges, the detection system cannot detect it successfully since there is no information about the attack in the library. Therefore, the misuse-based detection model has higher false negative rate.

There are many situation awareness models for mobile network. Endsley [20] is the most concerned model that defines situation awareness as perception, by including the current state and future trend of the element in a space during a time period. Endsley model divides situation awareness into three parts: (1) Perception: it acquires and collects important information in network, which is the fundamental step of situation awareness model, (2) Comprehension: the collected data is integrated and analyzed and then used to evaluate the current situation, and (3) Prediction: the future trend is predicted on the basis of the results in (1) and (2). The data is collected from the management device, monitoring device and security device in the network. The collected original data is further integrated and processed to evaluate the current network security situation. On this basis, the future situation can be also predicted.

Due to its good performance in statistics, HMM (Hidden Markov Model) [21–25] technique is rapidly developed and applied in fields as voice recognition, classification, security situation prediction, intrusion detection, etc. In the field of security situation prediction, Hisham [26] proposed the first HMM model with finite state to predict the multistep attack in cloud computing system. The probability of potential attack can be calculated through an adaptive risk model, whereas such potential attack can be found through the autonomous cloud intrusion detection systems. In this case, the countermeasure can be adopted in advance. This method successfully predicts the alarm on the dataset DARPA2000 LLDDOS1.0. Luktarhan [27] proposed a HMM-based false alarm filtering algorithm that can detect the multistep attack and achieve higher detection accuracy rate. Experiments of this algorithm on DARPA2000 show good performance. As mentioned, the HMM technique usually uses Viterbi algorithm although it is applied to predict the hidden state on the basis of the observed state. Additionally, the precondition of multistep attack prediction should know all steps of multistep attack. The prediction is the upcoming step, yet not the future network security situation.

Multiscale entropy is a nonlinear characteristic analysis method. In 2002, Costa et al. [28, 29] proposed the theory of multiscale entropy on the basis of sample entropy and applied it in HRV research. Results show that the multiscale entropy can give a more reasonable explanation on the differences of CHF and AF between disease and health than the sample entropy. MSE (Mean Square Error) and RMSE (Root Mean Square Error) are metrics to evaluate the complexity and irregularity of time series with different scale factors. It is used in classification of physiological time series [30], eccentricity fault detection of motor spindle [31], dynamical change of crude oil price [32], and network traffic analysis and anomaly detection [33]. In this work, the multiscale entropy with different scale factors is used to select the scale information to process the alarm time series. HMM is used to train the security state parameters, whereas the prediction is realized by weighting the value of autocorrelation coefficient between security situations.

The remaining of this paper is organized as follows. Section 2 introduces the multiscale entropy based Markov model; the details of the prediction algorithm are illustrated in Section 3. The experimental results and analysis are presented in Section 4, and finally, summary and future work are depicted in Section 5.

3. Multiscale Entropy Based Weighted HMM

3.1. Multiscale Entropy Analysis. In mobile network, the complexity of time sequence with limit length is usually measured by nonlinear characteristic analysis. The entropy values for different scale factors are used to select suitable scale and process next the time sequence, and the sequence is further trained by the HMM. The acquired data is one-dimensional alarm time sequence. The multiscale entropy not only can be used to characterize the complexity of the one-dimensional sequence, but also can display the detailed characteristics of one-dimensional alarm sequence from different scales. Therefore, it can be used to analyze the complexity of one-dimensional alarm sequence. For the alarm time sequence, N is the length of the sequence, and the calculation process of the multiscale entropy is described in this section.

Firstly, the coarse-grained sequence is generated. The original sequence is divided into nonoverlapping window with length s. The sequence values in the window are averaged to obtain the coarse grain sequence $\{y^s\}$ on the s scale. Each element of sequence y_j^s can be obtained via the following formula:

$$y_j^s = \frac{1}{s} \sum_{i=(j-1)s+1}^{js} x_i, \quad 1 \le j \le \frac{N}{S} \tag{1}$$

The length of each coarse-grained sequence is the value divided by the original length s. Here, the time sequence of s = 1 is the original sequence.

Secondly, the sample entropy is calculated. For coarse-grained sequence $\{y(1), y(2), \ldots, y(M)\}$, M is the length of the

sequence. Detailed steps to calculate the sample entropy is described next.

Step 1. The given model dimension is m, and the m dimensional vector is composed by the original sequence.

$$Y(i) = [y(i), y(i+1), \ldots, y(i+m-1)] \tag{2}$$

where $i = 1, 2, \ldots, M + m - 1$.

Step 2. The distance between $Y(i)$ and $Y(j)$ is defined as

$$d(i, j) = \max |y(i+k) - y(j+k)| \tag{3}$$

where $k = 0, 1, \ldots, m - 1$.

Step 3. Given a threshold value r, for each i statistic value, we have the ratio of the number $d(i, j) < r$ and the number of the total number $N - m + 1$ is denoted as $B_i^m(r)$.

$$B_i^m(r) = \frac{[d(i,j) < r]}{N - m + 1} \tag{4}$$

where $1 \le j \le N - m$, $j \ne i$, seeking its average value for all i.

$$B^m(r) = \frac{1}{N - m + 1} \sum_{i=1}^{N-m+1} B_i^m(r) \tag{5}$$

Step 4. Repeat the three steps above for $m+1$, get the $B^{m+1}(r)$.

Step 5. In theory, the sequence sample entropy is as

$$\text{SampEn}(m, r) = \lim_{N \to \infty} \left[-\ln \frac{B^{m+1}(r)}{B^m(r)} \right] \tag{6}$$

As N takes a finite value, sample entropy for sequence of length n can be estimated by

$$\text{SampEn}(m, r, N) = -\ln \frac{B^{m+1}(r)}{B^m(r)} \tag{7}$$

Step 6. With the sample entropy formula, each scale factor sequence can be calculated. A function with scale factor s as argument and with sample entropy as dependent variable can be created. Thus, it can be used to analyze the complex nature of the alarm time sequence.

$$MSE(s) = SampEn(y^s, m, r) \tag{8}$$

It is obvious that the multiscale entropy is related to the scale factor s, the embedding dimension m, and the similarity coefficient r. In this paper, parameters $m = 2$ and $r = 0.2$ are set.

With the above method, multiscale entropy of one-dimensional alarm time sequence can be calculated. This result is used in next-step data processing and brought in HMM for training. Finally, the corresponding scale state transition probability matrix can be obtained.

3.2. HMM Definition. HMM is a statistical learning model in mobile network, which can describe the process of generating observed sequence. Developed on the basis of Markov chain, HMM is a double random process, including the transfer from state to state and the transfer from state to output. In this work, the transfer probability matrix between states is calculated by HMM. There is a Markov assumption of state transfer, of which the probability of state at time t shifting to state at time t + 1 is only related to the state at time t and not related to any other states. Nevertheless, the assumption is not reasonable in practice since this probability depends on the current state and the historical states. To address this issue, this work has improved the Markov assumption of the state transfer probability in traditional HMM model. A new model is realized by weighting and used to predict the future security situation of the network.

The HMM includes two random processes, transition between states and transition between state to output. Markov chain prediction for state transition at hidden layer utilizes the idea of weighted Markov chain prediction in [34]. Namely, the weight probability distribution by using multistep situation transition probability is used to predict the system's next status. The correlation coefficient in each order reflects the strength of relationship between time sequences with different step. Analysis between transition probability and time sequence makes full use of known conditions for prediction. On this basis, the proposed prediction model is described as follows.

Step 1. Calculate correlation coefficients in various steps:

$$r_k = \frac{\sum_{t=1}^{n-k} (x_t - \overline{x})(x_{t+k} - \overline{x})}{\sum_{t=1}^{n} (x_t - \overline{x})^2} \tag{9}$$

where k represents k-order (with step length of k) correlation coefficient, x_t represents the security situation at time t, $\overline{x}$ is the mean value of the security situation, and n is sequence length of security situation value.

Step 2. Normalize the self-correlation coefficient of each order:

$$\omega_k = \frac{|r_k|}{\sum_{k=1}^{m} |r_k|} \tag{10}$$

ω_k is regarded as the normalized weight of the Markov chain with various steps (M is the largest order in prediction).

Step 3. Establish security situation grading standards (to determine hidden state space). The security situation is divided into four states [35]:

(i) *Good (G)* indicates the system is in good condition with no attack.

(ii) *Probed (P)* is in the state of being attacked. Under this state, the availability of a host is slightly decreased, and the probability of being attacked increases.

(iii) *Attacked (A)* indicates that the host has been attacked and the probability of being invaded increases.

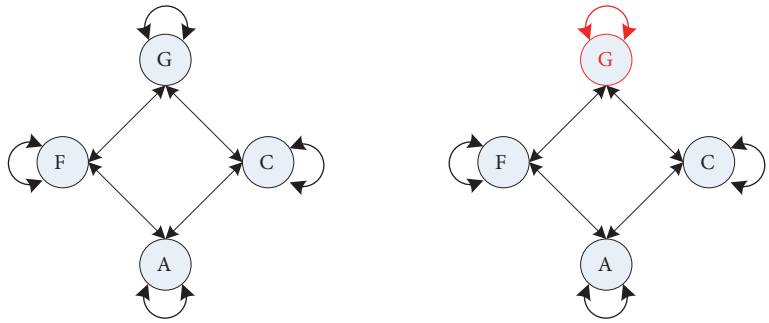


Figure 2: The state transition graph of Markov model.

(iv) *Compromised (C)* shows that the host has been invaded and is in a dangerous state. The confidentiality, integrity and availability of the host are destroyed.

The four states are, respectively, denoted by G, P, A, and C, so $S = \{G, P, A, C\}$. The transition between the host states constitutes a complete Markov chains, as shown in Figure 2.

Step 4. According to the security situation grade and alarm information, we sample security situation with different step lengths via HMM. Samples are trained to generate the transition probability matrix of different steps, determining as well the probability laws of security situation transition.

Step 5. On the basis of current state of the security situation, we can predict the probability of security status in the next time by using the security state transition matrix.

Step 6. Use the weight of prediction probability with different steps for a state as probability of secure state appeared in the next moment.

$$P_i = \sum_{k=1}^{m} \omega_k p_i^k \tag{11}$$

Step 7. With the obtained value of security state, the network situation can be analyzed.

Step 8. Repeat above Steps 4 to 7 to calculate the situation result under the scale factor and perform comparison to verify the proposed algorithm.

4. Algorithm Implementation

4.1. Algorithm Description. This work proposes a weighted hidden Markov prediction model on the basis of multiscale entropy in mobile network. The suitable scale factor is selected by using multiscale entropy, whilst the HMM is used to train the state transition probability matrix in mobile network. Changes of future security situation in mobile network are calculated by the weighting algorithm, of which entropy values of different scale data are obtained using the multiscale entropy. Two sets of data, respectively, with the maximum and the minimum entropy value which are processed to generate the multiorder state sequence matrixes are trained using HMM. The multiorder state transition probability matrix is used to calculate the security states with different scale at next time. The next security state is iteratively predicted using the obtained result, and differences between prediction value and the actual value are evaluated by MSE and RMSE. Besides, the prediction results for difference scale factors are also compared. Comparison shows the effect of multiscale entropy on weighted HMM prediction, and steps are detailed next.

Step 1. DARPA2000 dataset is processed by Snort, yielding original alarm time sequence.

Step 2. Time sequence is processed by multiscale entropy method. It generates the entropy value of different scale data. Two data sets with the maximum and the minimum entropy values are selected as the input of the next step.

Step 3. Selected data sets are used to generate the multiorder state sequence matrix, which are regarded as the parameter to train the original data.

Step 4. Multiorder alarm time sequence matrix is trained using HMM, yielding the multiorder state transition probability matrix.

Step 5. The weighting algorithm is used to calculate the autocorrelation coefficient of each order of the time sequence.

Step 6. The weighted prediction algorithm is used to calculate the network security state at next time for each scale factor.

Step 7. The prediction result is used as the input the prediction iteration to predict the next security state.

Step 8. The difference between the prediction result and the actual value is evaluated by the MSE. Besides, the prediction results for different scale factors are also compared.

4.2. Data Preprocessing. In the mobile network, the DARPA2000 dataset is collected using Snort. The alarm type is selected as the original time sequence. Based on the calculation results of multiscale entropy, nine data sets with the scale factors from 2 to 10 are used in the experiments. For each scale factor, we divide the original data on the basis of scale, yielding the multiorder data matrixes. These matrixes

```
Input: alarm data, α, β
Output: triple <signature, srcIP, dstIP>
{
    (1) Record the number N of all alarms
    (2) Foreach signature in Snort
    (3)  alarms generated by signature are written in set A;
    (4)  record the number n of alarms in set A;
    (5)   If (n/N > α)
    (6)   all the srcIPs and dstIPs in A respectively construct set S and set D;
    (7)     Foreach srcIP in S
    (8)       If (the ratio of srcIP > β)
    (9)       Return <signature, srcIP, any>
    (10)        Endfor
    (11)     Foreach dstIP in D
    (12)       If (the ratio of dstIP > β)
    (13)       Return <signature,any, srcIP>
    (14)       Endfor
    (15)    endif
    (16)    Endfor
}
```

Algorithm 1: The extraction of alarm stream.

will be further trained by HMM, and the corresponding state transition probability matrixes are generated next.

The different order of matrix that corresponds to the different scale factor is used as HMM training data. Next, the corresponding state transition probability matrix is obtained, following formula (12), where s is scale factor and k (k=1,2,3,4) is multiorder transition probability.

$$P_s^k = \begin{bmatrix} p_{11}^{(k)}(t') & p_{12}^{(k)}(t') & p_{13}^{(k)}(t') & p_{14}^{(k)}(t') \\ p_{21}^{(k)}(t') & p_{22}^{(k)}(t') & p_{23}^{(k)}(t') & p_{24}^{(k)}(t') \\ p_{31}^{(k)}(t') & p_{32}^{(k)}(t') & p_{33}^{(k)}(t') & p_{34}^{(k)}(t') \\ p_{41}^{(k)}(t') & p_{42}^{(k)}(t') & p_{43}^{(k)}(t') & p_{44}^{(k)}(t') \end{bmatrix} \quad (12)$$

4.3. Implementation of Prediction Algorithm. The distribution of various attributes of alarm in mobile network is not uniform. By analysis, there are lots of alarm streams generated by a few attributes.

The alarm stream is represented by triple <Signature, src IP, dst IP>, where "Signature" is the rule to generate the alarm and "src IP" and "dst IP" are, respectively, original IP address and the destination IP address. First, the one-day alarms of intrusion detection system in mobile network are collected, as the ratio threshold α of "Signature" to generate the alarm and the ratio threshold β of IP to generate alarm are determined. If the ratio threshold of "Signature" exceeds the value of α, the corresponding src IP and dst IP should be recorded. Moreover, if the ratio threshold of IP to generate alarm exceeds the value of β, the triple will be further analyzed as shown in Algorithm 1.

The returned triple in algorithm will be recorded each hour. The alarm sequence of a certain triple can be denoted by a random process $\{x(t) : t \in N\}$, where x(t) is the number of alarms generated by the triple within t hours. The period of the sequence will be further analyzed with Algorithm 2.

5. Results and Discussion

5.1. Dataset. DARPA2000 is an offline intrusion detection dataset, available and maintained by MIT's Lincoln Laboratory [36]. It has become a standard dataset to verify effectiveness of intrusion alert correlation algorithm, scene construction algorithm, and trend prediction algorithm.

DARPA2000 dataset includes all monitored data package in demilitarized zone (DMZ) and the internal (INSIDE) by using TCPDUMP. The dataset contains instances in attack scenarios of LLDOS 1.0 and LLDOS2.0.2. In LLDOS1.0 scenario, the attacker captures and controls 3 hosts in Eyrie airbase via vulnerability of Solaris Sadmind, uploads Mstream tool, and performs DDOS attack on a government website. Different with LLDOS 1.0, LLDOS2.0.2 utilizes a more concealed method to seek host with vulnerability and setup Mstream.

In this research, we make use of DARPA2000 dataset to verify the effectiveness of the proposed algorithm. It contains two multistep DDOS attack scenarios. There are lots of data, redundancies, and false alarms in both scenarios. To verify the effectiveness of the method, we just take attack alarm data in five steps to complete the experiments: the attack scene time is about three hours to LLDDOS1.0 and about 1.5 hours for LLDDOS2.0.

5.2. Calculation of Multiscale Entropy. With corresponding formula, the original alarm time sequence can be calculated, and the scale factor is selected from 2 to 20. The histogram of entropy values is depicted in Figure 3.

In this figure, samples of entropy values change with the scale factor, where the minimum and the maximum values

Input: the alarm sequence $\{x(t) : t \in N\}$ generated by triple <*Signature, srcIP, dstIP*>
Output: alarm removal rule <*Signature,srcIP,dstIP*>
{
(1) select an alarm sequence $\{x(t) : 1 \le t \le N,\ N > 5p\}$ generated by a triple;
(2) determine the autocorrelation sequence $\{R(m) : 1 \le m \le N\}$;
(3) while $(1 \le f \le N)$
(4) calculate the energy value of frequency f;
(5) endwhile
(6) calculate the frequency f_{max} with the maximum energy;
(7) calculate the cycle of alarm sequence;
(8) calculate the value of F;
(9) if $F \ge F_\alpha (b - 1, nb - b)$
(10) generate the alarm removal rule <*Signature,srcIP,dstIP*>;
(11) Endif
}

Algorithm 2: Period analysis algorithm.

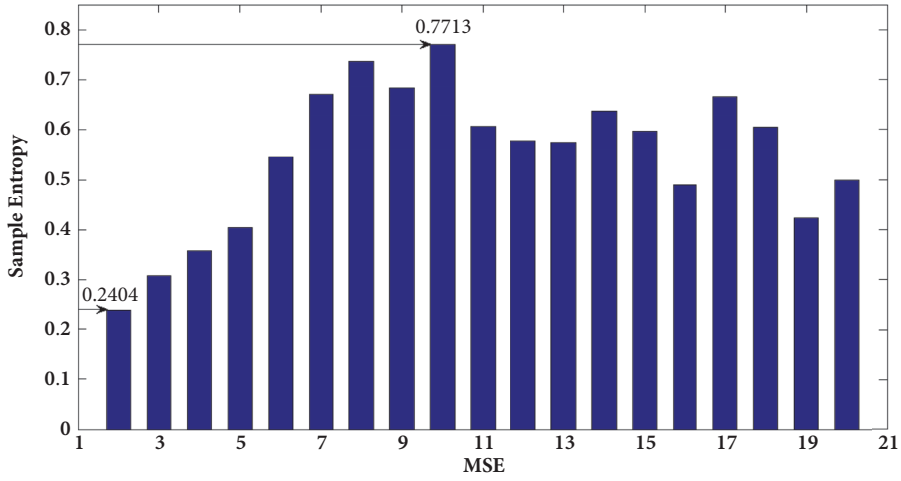


Figure 3: Multiscale entropy histogram sequence of the original data of the alarm time.

are obtained when the scale factors are 2 and 10, respectively. The value of entropy reflects complexity of time sequence and frequent changes of alarm data. Thus, the security situation can be easily reflected by entropy value. We assume that the entropy value has effect on prediction results, whose assumption will be further proven in experiments that follow next.

5.3. Weighted Prediction Performance. After calculating the formulas (5) and (6), the autocorrelation coefficients of the training data are obtained and normalized, as shown in Table 1.

The normalized autocorrelation coefficients are calculated as the weighted values, and the results are compared with the calculated results. The maximum probability of state is used as the prediction result at the next time.

We added the multiscale entropy method on the basis of method introduced in [35]. The overall changes of network security situation cannot be clearly observed due to dramatic changes, and therefore, we introduce Bezier curve to describe the overall trend of network security situation. Advantages using Bessel curve include smoothing the change curve of network security situation. The actual security situation prediction graph for LLDDOS2.0 is shown in Figure 4.

As depicted in this figure, the network security situation value is not high at the beginning. During this period, the network is attacked by Ipsweep scanning and the host is probed by Sadmind Ping. After 3:00 hours, the trend of value decreases during a period, which shows that the attacker has completely controlled the host. The rising trend reflects the attackers' launch DDOS attacks; the decline shown next reflects that the attack is gradually weakened. As a result, the trend curve is more vivid to describe the trend of the network security situation. It is helpful for administrators to manage the network security situation. After the preprocessing of the data, the multiorder state transition probability matrix is obtained. As listed in Tables 2 and 3 the state transition probability matrix when the scale factors are 2 and 10.

As the original data is preprocessed to generate the matrix of multiorder state transition probability with different scales,

Table 1: The autocorrelation coefficients training data and the normalized results.

Evaluation	Order 1	Order 2	Order 3	Order 4
r_k	0.5628	0.4658	0.4507	0.2757
w_k	0.3207	0.2654	0.2568	0.1571

Table 2: The state transition probability matrix when scale factor =2.

Prediction step	State	G	P	A	C
P=1	G	0.9168	0.0012	0.0385	0.0435
	P	0.0615	0.9259	0.0126	0.0000
	A	0.0000	0.0000	1.0000	0.0000
	C	0.0000	0.0000	0.0015	0.9985
P=2	G	0.8156	0.0000	0.1755	0.0086
	P	0.0042	0.9956	0.0000	0.0000
	A	0.0000	0.0000	0.9861	0.0138
	C	0.0021	0.0000	0.0018	0.9961
P=3	G	0.7271	0.0032	0.2697	0.0000
	P	0.1689	0.8311	0.0000	0.0000
	A	0.0000	0.0866	0.8864	0.0269
	C	0.0000	0.0000	0.0000	1.0000

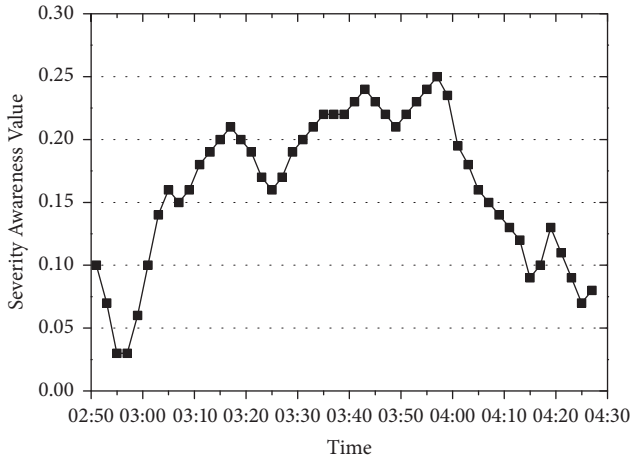


Figure 4: The actual value of network security situation.

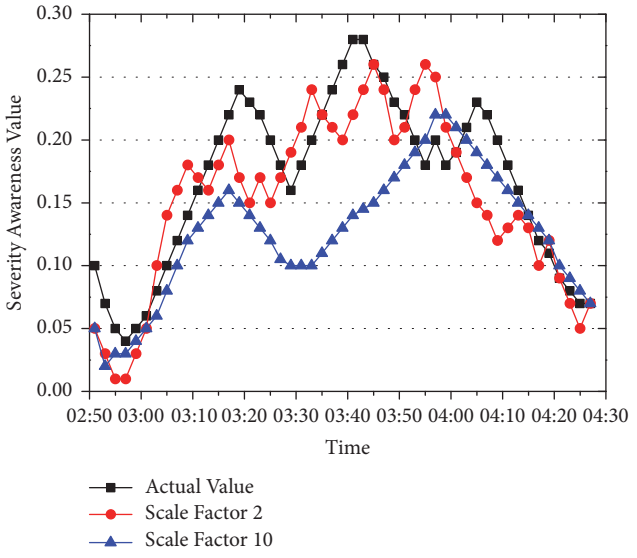


Figure 5: The one-step prediction result.

the scale factor as 10 and the order as 4 are chosen for evaluation. The prediction results are listed in Table 4.

In Table 4, C is shown to be the secure state at the next time with the largest probability, representing that the host is intruded under a dangerous state. The confidentiality, integrity, and availability are damaged. In this case, corresponding measures should be adopted and result in the fact that Table 4 can be used as the input of the next iteration for future prediction.

The probability of each state at the next moment of time sequence is obtained through this proposed method. The value of the probability is used to determine the status of the security situation. Figure 5 shows the actual value and the predictive value of the different scale factors in the time period from 2:51 to 4:25 hours. The horizontal axis represents time and the vertical axis represents the security situation. It shows the actual value of security situation; curves with other color represent the prediction results of different scales. The overall trend predictions and actual values are almost the same. It demonstrates the effectiveness of the weighted prediction method proposed in this paper.

The MSE is introduced to illustrate the different scaling factors of prediction and the difference between actual and predictive value. The multiscale entropy and the MSE values in one-step prediction are listed in Table 5.

The different scale factor corresponding to RMSE and MSE in one-step prediction is shown in Table 5. As the scale factor is 2, it gets smaller entropy and larger RMSE, and with the scale factor 10, the results are opposite ones. Therefore,

Table 3: The state transition probability matrix when scale factor =10.

Prediction step	State	G	P	A	C
P=1	G	0.8357	0.0657	0.0801	0.0185
	P	0.0487	0.8778	0.0736	0.0000
	A	0.0015	0.0187	0.9763	0.0035
	C	0.0139	0.0108	0.0000	0.9753
P=2	G	0.7155	0.0311	0.2238	0.0295
	P	0.1148	0.8192	0.0660	0.0000
	A	0.0076	0.0271	0.9578	0.0075
	C	0.0257	0.0214	0.0000	0.9529
P=3	G	0.5375	0.0698	0.3236	0.0691
	P	0.1557	0.7434	0.1009	0.0000
	A	0.0015	0.0802	0.8802	0.0381
	C	0.0396	0.0000	0.0101	0.9503

Table 4: Results of security situation prediction at next time.

Initial state	delay		G	P	A	C
P	1	0.321	0.0487	0.8778	0.0736	0.0000
C	2	0.265	0.0257	0.0214	0.0000	0.9529
C	3	0.257	0.0396	0.0000	0.0101	0.9503
A	4	0.157	0.0000	0.0000	0.9888	0.0112
	Weighting sum P_i		0.03262	0.2874	0.1815	0.4985

Table 5: The multiscale factor and the MSE /RMSE values in one-step prediction.

Scale factors	2	3	4	5	6	7	8	9	10
MSE	0.2404	0.3082	0.3586	0.4051	0.5456	0.6720	0.7372	0.6839	0.7713
RMSE	0.1849	0.1885	0.1887	0.1877	0.1865	0.1857	0.1771	0.1849	0.1849

Table 6: The multiscale factor and the MSE /RMSE values in two-step prediction.

Scale factors	2	3	4	5	6	7	8	9	10
MSE	0.2404	0.3082	0.3586	0.4051	0.5456	0.6720	0.7372	0.6839	0.7713
RMSE	0.1989	0.1978	0.1975	0.1971	0.1981	0.1976	0.1891	0.1968	0.1968

Table 7: The multiscale factor and the MSE /RMSE values in three-step prediction.

Scale factors	2	3	4	5	6	7	8	9	10
MSE	0.2404	0.3082	0.3586	0.4051	0.5456	0.6720	0.7372	0.6839	0.7713
RMSE	0.1919	0.1986	0.1982	1980	1973	1946	0.1829	0.1919	0.1919

scale factor with larger entropy entails prediction with higher accuracy, what verifies the assumption presented in previous section. In addition, RMSE and MSE are verified between predicted value and real value under scale factor from 2 to 10 with the step length of 2 and 3. Weighted prediction method can be used to predict security state at the next moment. It can also predict the results of iterations to obtain the future network security conditions within period. The prediction results and root mean square error values with different scale factors and different steps are given in Tables 6 and 7 and Figures 6 and 7.

The prediction curves show the predictive values are close to actual values, what demonstrates the effectiveness of the proposed weighted HMM algorithm. At the same time, it is possible to note that the change on prediction steps will not affect the effectiveness of multiscale factor predicted results.

6. Conclusions

Current HMM-based prediction algorithms predict the unknown state via a known state. It has limitations and the prediction accuracy is not high enough. To address this issue,

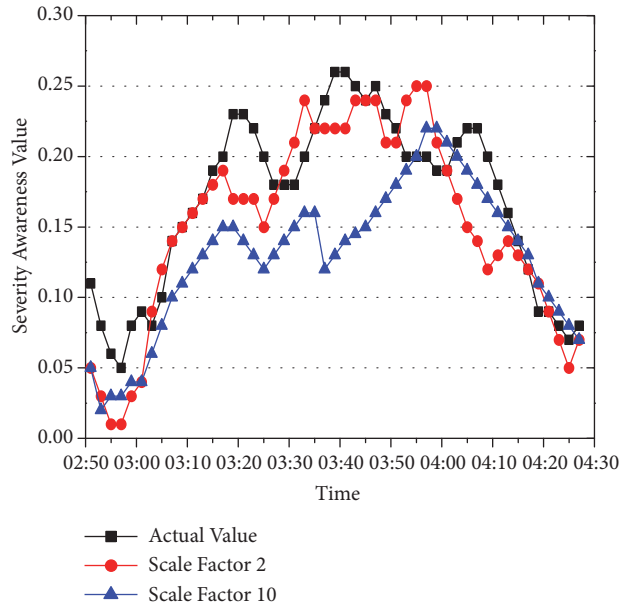


Figure 6: The two-step prediction result.

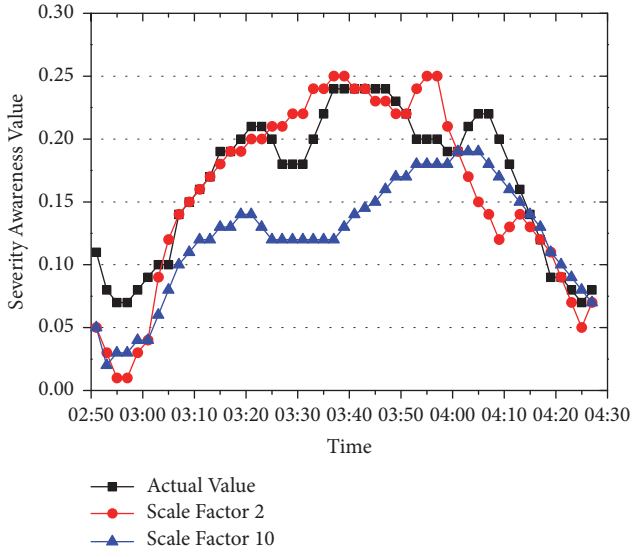


Figure 7: The three-step prediction result.

this work presents the application of the weighted HMM model to predict the security situation of mobile networks, where the multiscale entropy method is used to select the appropriate scale factor of the data. It will be regarded as training data for HMM model to obtain the state transfer probability matrix, whilst the correlation coefficients are considered as weights to predict changes of network security situation. In the proposed model, we use the alarm time sequence as original data, and with the multiscale entropy method, an appropriate scale factor is selected as training data to train the parameters in HMM model. Finally, the predictive results are evaluated and compared to real security situation. Experimental results have shown that the proposed method is accurate and effective. Weighted method can not only use current network state, but also consider the historical security situation. The dependent relationship between the security situations is fully used. It could predict network security at the next state. The network security administrator could take measures to maintain network security and avoid more attacks in advance.

Based on the prediction issues in mobile network, several security situation prediction algorithms are analyzed as follows: (1) there is causal relationship between attack behaviors, (2) the probability to occur different attacks is different, (3) the evidence of future attacks includes the attack itself, (4) the purpose of attack can be inferred, and (5) the evidence of future attacks has relationship with the change of network security situation. These features provide a thought to predict the security situation by extracting evidence. Besides, other aspects can be also researched, such as real-time data extraction and data fusion of multisource heterogeneous sensors.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work is supported by the National Science Foundation of China (Grant 61572188), Xiamen Science and Technology Foundation (Grant 3502Z20173035), Scientific Research Program of New Century Excellent Talents in Fujian Province University, Fujian Provincial Natural Science Foundation of China (Grant 2018J01570), the CERNET Innovation Project (Grant NGII20170411), the National Nature Science Foundation of Fujian Province (Grant 2018J01544), the Scientific Research Program of Outstanding Young Talents in Universities of Fujian Province, the Key Project of Natural Foundation for Young in Colleges of Fujian Province (Grant JZ160466), and Hunan Provincial Natural Science Foundation of China (Grant 2016jj2058).

References

[1] S. Touboul, "System and method for providing network security to mobile devices," *Yoggie Security Systems*, 2017.

[2] W. Liang, Y. Xie, and X. Chen, "A two-step MF signal acquisition method for wireless underground sensor networks," *Computer Science & Information Systems*, vol. 13, no. 2, pp. 623–638, 2016.

[3] W. Liang, Y. Huang, J. Xu, and S. Xie, "A distributed data secure transmission scheme in wireless sensor network," *International Journal of Distributed Sensor Networks*, vol. 13, no. 4, pp. 1–11, 2017.

[4] W. Liang, Z. Ruan, Y. Wang, and X. Chen, "RESH: a secure authentication algorithm based on regeneration encoding self-healing technology in WSN," *Journal of Sensors*, vol. 2016, 11 pages, 2016.

[5] R.-F. Wu and G.-L. Chen, "Research of network security situation prediction based on multidimensional cloud model," in *Proceedings of the 6th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS '12)*, pp. 409–414, July 2012.

[6] J. Wang, M. Ouyang, W. Liang, and J. Hou, "Device-to-device relay cooperative transmission based on network coding," *KSII Transactions on Internet and Information Systems*, vol. 11, no. 7, 2017.

[7] J. Wang, W. Luo, W. Liang, X. Liu, and X. Dong, "Locally minimum storage regenerating codes in distributed cloud storage systems," *China Communications*, vol. 14, no. 11, pp. 82–91, 2017.

[8] C. Jiang, T. Li, J. Liang, and H. Wu, "Low-latency and energy-efficient data preservation mechanism in low-duty-cycle sensor networks," *Sensors*, vol. 17, no. 5, p. 1051, 2017.

[9] Snot, https://www.snort.org/, 2017-12-21.

[10] CVE, http://cve.mitre.org/, 2017-12-23.

[11] Bugtraq, http://www.bugtraq-team.com/, 2017-12-23.

[12] CVSS, http://resources.infosecinstitute.com/common-vulnerability-scoring-system/, 2017-12-23.

[13] C. Kruegel and W. Robertson, "Alert Verification - Determining the Success of Intrusion Attempts," pp. 1-14, 2004.

[14] Nessus, https://www.tenable.com/products/nessus/nessus-professional, 2017-12-23.

[15] P. A. Porras, M. W. Fong, and A. Valdes, "Mission-impact-based approach to INFOSEC alarm correlation," in *Proceedings of the International Conference on Recent Advances in Intrusion Detection*, pp. 95–114, Springer-Verlag, 2002.

[16] H. Shake, B. Hazzard, and D. Marquis, "Assessing network infrastructure vulnerabilities to physical layer attacks," in *Proceedings of the 22nd National Information Systems Security Conference*, 1999.

[17] T. Bass, "Intrusion detection systems and multisensor data fusion," *Communications of the ACM*, vol. 43, no. 4, pp. 99–105, 2000.

[18] F. Hock and P. Kortiš, "Commercial and open-source based Intrusion Detection System and Intrusion Prevention System (IDS/IPS) design for an IP networks," in *Proceedings of the 13th IEEE International Conference on Emerging eLearning Technologies and Applications (ICETA '15)*, pp. 1–4, November 2015.

[19] D. A. Bhosale and V. M. Mane, "Comparative study and analysis of network intrusion detection tools," in *Proceedings of the 1st International Conference on Applied and Theoretical Computing and Communication Technology (ICATCCT '15)*, pp. 312–315, October 2015.

[20] W. Hu, J. Li, X. Chen, X. Jiang, and M. Zuo, "Scalable model for network situational awareness based on Endsley's situation model," *High Technology Letters*, vol. 13, no. 4, pp. 395–401, 2007.

[21] A. Årnes, F. Valeur, G. Vigna, and R. A. Kemmerer, "Using hidden Markov models to evaluate the risks of intrusions system architecture and model validation," *Lecture Notes in Computer Science*, vol. 4219, pp. 145–164, 2006.

[22] W. Liang, Z. Chen, X. Yan, X. Zheng, and P. Zhuo, "Multi-scale entropy-based weighted hidden markov network security situation prediction model," in *Proceedings of the 2nd IEEE International Congress on Internet of Things (ICIOT '17)*, pp. 97–104, June 2017.

[23] K.-L. Yap and Y.-W. Chong, "Optimized access point selection with mobility prediction using hidden Markov model for wireless network," in *Proceedings of the 9th International Conference on Ubiquitous and Future Networks (ICUFN '17)*, pp. 38–42, July 2017.

[24] X. Tan, W. Wang, and X. Hong, "A hidden markov model used in intrusion detection," *Journal of Computer Research & Development*, 2003.

[25] G. Zhu, K. Song, P. Zhang, and L. Wang, "A traffic flow state transition model for urban road network based on Hidden Markov Model," *Neurocomputing*, vol. 214, pp. 567–574, 2016.

[26] H. A. Kholidy, A. Erradi, S. Abdelwahed, and A. Azab, "A finite state hidden markov model for predicting multistage attacks in cloud systems," in *Proceedings of the 12th IEEE International Conference on Dependable, Autonomic and Secure Computing (DASC '14)*, pp. 14–19, August 2014.

[27] N. Luktarhan, X. Jia, L. Hu, and N. Xie, "Multi-stage attack detection algorithm based on hidden markov model," in *Web Information Systems and Mining*, vol. 7529 of *Lecture Notes in Computer Science*, pp. 275–282, Springer Berlin Heidelberg, 2012.

[28] M. Costa, A. L. Goldberger, and C.-K. Peng, "Costa, goldberger, and peng reply," *Physical Review Letters*, vol. 92, no. 8, 2004.

[29] M. Costa, A. L. Goldberger, and C. Peng, "Multiscale entropy analysis of biological signals," *Physical Review E: Statistical, Nonlinear, and Soft Matter Physics*, vol. 71, no. 2, 2005.

[30] R. Istenič, P. A. Kaplanis, C. S. Pattichis, and D. Zazula, "Multiscale entropy-based approach to automated surface EMG classification of neuromuscular disorders," *Medical & Biological Engineering & Computing*, vol. 48, no. 8, pp. 773–781, 2010.

[31] P. Xie, G.-Q. Jiang, X. Wu, and X.-L. Li, "Rolling bearing fault diagnosis based on multiscale entropy and distance evaluation," *Acta Metrologica Sinica*, vol. 34, no. 6, pp. 548–553, 2013.

[32] E. Martina, E. Rodriguez, R. Escarela-Perez, and J. Alvarez-Ramirez, "Multiscale entropy analysis of crude oil price dynamics," *Energy Economics*, vol. 33, no. 5, pp. 936–947, 2011.

[33] Y. Ruo-Yu and Z. Qing-Hua, "Multi-scale entropy based traffic analysis and anomaly detection," in *Proceedings of the 8th International Conference on Intelligent Systems Design and Applications (ISDA '08)*, pp. 151–157, Kaohsuing, Taiwan, November 2008.

[34] Y. Feng and W. Han, "The application of weighted Markov-chain to the prediction of river runoff state," *Systems Engineering—Theory & Practice*, vol. 19, no. 10, pp. 89–94, 1999.

[35] J. Lei, *Research on Network Security Threats and Situation Assessment*, Huazhong University of Science and Technology, 2008.

[36] MIT Lincoln Laboratory, https://www.ll.mit.edu/r-d/datasets.

Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 8056195, 12 pages
https://doi.org/10.1155/2018/8056195

Research Article

CEPTM: A Cross-Edge Model for Diverse Personalization Service and Topic Migration in MEC

Hongchen Wu,[1] **Huaxiang Zhang**,[1] **Lizhen Cui**,[2,3] **and Xinjun Wang**[3]

[1]*School of Information Science and Engineering, Shandong Normal University, Jinan 250014, China*
[2]*Shandong Provincial Key Laboratory of Software Engineering, Shandong University, Jinan, China*
[3]*School of Computer Science and Technology, Shandong University, Jinan 250101, China*

Correspondence should be addressed to Hongchen Wu; wuhongchen@sdnu.edu.cn

Received 17 April 2018; Accepted 2 August 2018; Published 12 August 2018

Academic Editor: Shangguang Wang

For several reasons, the cloud computing paradigm, e.g., mobile edge computing (MEC), is suffering from the problem of privacy issues. MEC servers provide personalization services to mobile users for better QoE qualities, but the ongoing migrated data from the source edge server to the destination edge server cause users to have privacy concerns and unwillingness of self-disclosure, which further leads to a sparsity problem. As a result, personalization services ignore valuable user profiles across edges where users have accounts in and tend to predict users' potential purchases with insufficient sources, thereby limiting further improvement of QoE through personalization of the contents. This paper proposes a novel model, called CEPTM, which (1) collects mobile user data across multiple MEC edge servers, (2) improves the users' experience in personalization services by loading collected diverse data, and (3) lowers their privacy concern with the improved personalization. This model also reveals that famous topics in one edge server can migrate into several other edge servers with users' favorite content tags and that the diverse types of items could increase the possibility of users accepting the personalization service. In the experiment section, we use exploratory factor analysis to mathematically evaluate the correlations among those factors that influence users' information disclosure in the MEC network, and the results indicate that CEPTM (1) achieves a high rate of personalization acceptance due to the availability of more data as input and highly diverse personalization as output and (2) gains the users' trust because it collects user data while respecting individual privacy concerns and providing better personalization. It outperforms a traditional personalization service that runs on a single-edge server. This paper provides new insights into MEC diverse personalization services and privacy problems, and researchers and personalization providers can apply this model to merge popular users' like trends throughout the MEC edge servers and generate better data management strategies.

1. Introduction

With the proliferation of the Internet of things and the burgeoning of the cloud computing paradigm network, mobile edge computing (MEC) has been widely adopted at a tremendous speed. A large collection of computing resources, such as mobile devices, application servers, and storage units, are utilized to serve users in all types of tenant models. The presentation and wide application of MEC have changed people's daily work and are utilized in the data centers of computing companies. A recent report has demonstrated that extensible infrastructure and processing engine technology have greatly impacted the methods of running applications and websites across multiple edge servers [1]. A large number of data centers are built by venders to serve a very large number of users, and these large-scale, commodity-computer data centers have sufficient resources to provide personalization to a large number of users. As the de facto centralized big data platform, the cloud computing paradigm supports QoE with a fruitful number of benefits—convenience, pay for each use, and ubiquity—which has given birth to a worldwide range of industry companies [2]. The widely applied denominator in MEC is the deployment of cloud computing-like capabilities at the edge of the IoT network. Figure 1 demonstrates the

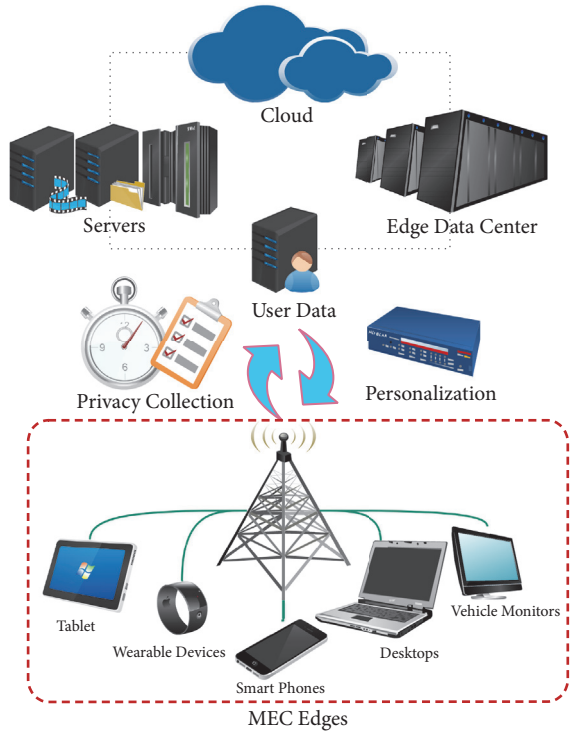


Figure 1: The structure of mobile edge computing in the cloud environment.

structure of the personalization service provided by MEC. Edge data centers, which are owned and deployed by infrastructure providers, implement a multitenant virtualization infrastructure. Edge devices automatically cooperate with one another for service collaboration in the cloud environment, and personalization activity in edge servers collects users' data and enhances end users' experiences by matching the items and their potential buys. These services have forged into various support mechanisms and management platforms, creating an open ecosystem where a multitude of customers can be served.

Despite the achievements, MEC is still far from a panacea. The massive growth of MEC edge servers [3], including websites and applications, has led to an explosive volume of data and has caused the problem of "information overload" [4]. An unprecedented number of information resources are available, but users are often unsure of what to do [4, 5]. Personalization services can apply information-filtering methods to eliminate the items that users would not click on according to their browsing histories and preferences [6, 7]. However, a prerequisite for a personalization service is that it must collect significant amounts of personal data for precise recommendations [8]. This approach introduces three main issues into the research field. First, although a tremendous volume of data is available on the Internet, the valuable data that are useful for analysis constitutes only a small percentage, which is called "data sparsity" [9]. Second, the collection of large amounts of data is in conflict with today's users' high level of privacy concern [10]. In addition, users often have difficulty in deciding on information disclosure and do not possess sufficient knowledge to evaluate the risks and benefits [11]. As a result, users usually deny most information requests. This action leads to a "worse-to-worse" loop, as users do not trust the personalization service and do not share much private information as input for the information-filtering algorithm. The personalization service cannot generate good personalization using such a small amount of data, and the users will, in turn, trust the personalization service even less. Third, users' standards for accepting personalization are not low, and the users may no longer accept personalization in one type of item on an MEC edge server. Instead, users fall prey to various types of "innovative" personalization [12]. This user-tailored personalization approach is difficult to measure, and the trigger that leads a user to accept personalization is unknown. These problems have limited the further development of personalization services in MEC networks, but the available data from users' multiple accounts on various MEC edge servers have provided a solution from a new perspective. The related papers had made some progress on personalization but lack consideration on privacy issues, which indicate the motivation of our work.

This paper establishes a cross-edge model, called CEPTM, which integrates valuable resources for analyzing users' "like" behaviors from multiple MEC edge servers for diverse personalization and for merging real-time popular buys from a source edge server into all destination edge servers. This model is equipped with a personalization engine for producing user-tailored suggestions of what items to view. CEPTM is applied in the experiment section and achieves higher prediction accuracy than a baseline personalization service that is running on a single-edge server. This finding is attributed to it determining the factors that influence users' decisions on accepting personalization. Our study also focuses on reconciling the tension between user acceptance of a personalization and various other factors, e.g., personalization diversity and personalization quality, by using exploratory factor analysis (EFA), which is a common modeling method for explaining the correlations among the variables in terms of fundamental factors. From both the user and managerial perspectives, the results of the model on real-life datasets from the MEC edge servers enabled us to answer the following question: what should personalization providers do to cause mobile users to be more likely to accept the personalization and more willing to disclose their privacy? Our suggestions are as follows:

(1) Collect as many user profiles as possible. Taking this action gives the personalization provider access to more available information for generating high-accuracy personalization, which could further optimize user QoE and increase user satisfaction with the personalization service.

(2) The scope of personalization should be diverse and, especially, not restricted to the types of products with which users are very familiar, as this could generate repeated items. Repeated items are those that users have already liked, and showing them will cause users to lose interest in the personalization service and MEC edge server. Compared with a traditional personalization service method, our model uses common tags from various types of items by similarity computing.

(3) Respect users' privacy concerns. An effective user-tailored personalization approach should not only produce a personalization that the user will like according to his or her preferences but also collect the user's privacy in a volume that will not raise his or her privacy concerns. It is noted that a user's tolerance of information disclosure differs from person to person. Once users feel that their privacy has been compromised by data collection, their trust in the personalization service will decrease. Thus, they will decrease their amount of disclosure. However, we observed that most MEC managers still seek an agreement with a data collection policy from whoever signs into their edge servers.

2. Related Work

This study builds on existing research on personalization services and privacy in MEC networks and addresses several theoretical and practical gaps in prior studies. Here, we briefly introduce some related work and describe the manner in which the problems of personalization service and privacy were solved. Gaps were identified in these studies, which we subsequently discuss in this work.

2.1. Personalization Service versus Privacy Concern. MEC aims to offer computing resources and information technology services at the network edge, where mobile users face a bewildering array of options in an MEC edge server, such as which movie to watch [13], which news to read [14], and which people to add as friends [15]. Personalization services have been applied to help users find the items that best fit their interests by learning the past item-user relationships and performing information-filtering methods. Common methods of personalization services are collaborative filtering methods [16], content-based methods [17], and hybrid methods [18]. Collaborative filtering methods plot the user-item relationship as a metric and use similarity computing, such as Pearson correlations, to filter the items that have large gaps with users' preferences in the distance function. Content-based methods integrate the information that best describes the items, such as manually tagged annotations from experts and other users, and rank the items according to how well they match the users' preferences. A hybrid method is a mixture of the previous two categories.

However, there remain unsolved problems in this body of work. All of the methods necessitate the availability of users' rudimentary data in such a way that the personalization can know the users' needs at a basic level. One general rule is that the quality of the personalization improves with the available volume of personal data [19], but users usually do not agree to this method of data collection [20], which could lead to unwanted exposure of information. Conversely, users could disclose their long-term privacy for temporary benefits, such as filling in questionnaires in exchange for coupons and discounts [21]. This issue is called the "personalization-privacy paradox" [22], and it has attracted the interest of several researchers. Some researchers have made progress by extending this paradox, and they believe that the amount of privacy that is disclosed to personalization services depends on how much users trust the system. Meyffret argued that user anonymity allows users to feel less concerned about their privacy [23]. Letting users decide what privacy information to share with the personalization service is an option [24], but users often rely on default settings when considering their privacy controls [25]. Most users lack sufficient knowledge with regard to personalization services and privacy and are not certain about privacy control [26]. As a result, recent studies have proposed the idea of "usable privacy" [27], which states that users shall be assisted with decision support for information disclosure. This topic aims at kindly informing users that a proper method of data collection would not harm their personal information. Personalization providers should make use of personal data and convince users that good data collection generates more benefits than potential risks.

2.2. Single-Edge versus Cross-Edge. Personalization service techniques aim at predicting missing ratings for an item based on previous ratings, which are collected from user-item connections. However, the sparsity problem in single-edge personalization services has become a major bottleneck for prediction algorithms. There are various reasons for this problem. For example, users' privacy concerns may cause anxiety, thereby limiting the amount of shared information; each user may only view a small number of items, which yields limited useful data for similarity computing in personalization services; and data for new users are very limited, which leaves the personalization service with a "cold start" problem [28]. To alleviate data sparsity, one solution is to integrate the data and knowledge using cross-edge rather than single-edge solutions. The joint method has several advantages. First, there is no need to learn new users' behaviors and rules from the very beginning. Although globalization could lead to communication and coordination delays, empirical studies have suggested that the representation of user-item relationships can be efficiently accomplished [29]. Second, cloud computing make it feasible to provide cross-edge personalization services, thereby increasing the number of sites that allow users to sign into multiple accounts. A joint approach that combines user data from various sites will become a major trend in personalization services and MEC. Providing heterogeneous cross-edge personalization could shed light on brand new and promising models, which could tremendously increase the interaction between users and items using both single-edge and cross-edge methods. Users' interests in items could also be discovered in a wider scope, thereby leading to diverse personalization from which users can choose [30]. As a result, the likelihood that users would accept the personalization would increase. Finally, users' privacy concerns could be alleviated. Data collection conflicts with users' high level of privacy protection, so they may share very little information with each edge server. Cross-edge methods could integrate users' privacy to better predict their potential buys [31]. By establishing relatedness across multiple rating matrices, Li constructed a user-item joint mixture model and the corresponding ratings and showed that their proposed algorithm for effective cross-domain collaborative filtering personalization service can outperform individual models that are trained on a single domain [32]. Pan observed that, in different domains, the

user feedback could be heterogeneous, such as ratings versus clicks. To determine a solution using this feedback, both user and item knowledge from auxiliary data sources are integrated through a principled matrix-based transfer learning framework, and the discovered principle can be used to coordinate both users and items in the auxiliary data matrices and transfer them to the target domain to reduce the effect of data sparsity [33]. The sparsity problem can be addressed by jointly considering multiple heterogeneous link prediction tasks, such as the links between users and different types of items, including books, movies, and songs, and the learned intertask similarity was applied to correct the biases and skewness of the distribution from the single domain.

In summary, the current shortcomings of single-edge methods and single-domain algorithms have limited the further development of personalization services in MEC due to limited data integration, and new problems are encountered when incorporating a cross-edge personalization service. The following question was posed: Is it possible to migrate useful knowledge by establishing bridges between the resource edge server and the destination edge servers, to optimize the performance? This question still remains unsolved in recent personalization service studies. Before setting up a personalization service from multiple MEC edge servers, we must determine what mobile users truly want from personalization, what factors influence the likelihood that they will accept personalization from a cross-edge personalization service, and what we should do to make them more willing to disclose their privacy in the MEC network. This paper is aimed at these three targets and designs a cross-edge model prototype, which is focused on personalization service and data management for users from a cross-edge MEC environment. This model, CEPTM, provides an informing function of what users' friends are doing on various MEC edge servers and is equipped with a personalization engine to produce user-tailored suggestions of what items to view. By integrating the resources of user-item relationships from multiple MEC edge servers, we also discover the factors that affect users' willingness to accept personalization with exploratory factor analysis.

3. Methods and Materials

3.1. Model for Cross-Edge MEC. In this paper, we propose a prototype of the model, called CEPTM, which consists of the interactions between users and diverse items from multiple MEC edge servers, personalization for new items, and privacy collection strategies. CEPTM has two basic functions: (a) informing users about what their friends are doing in all MEC servers and (b) generating personalization for items by calculating the possibility that one user may like the personalization, while focusing on examining the factors that affect users' decisions on accepting the personalization. The prototype of the model is depicted in Figure 2, where Johnson has friends across different MEC devices; e.g., Mike is his friend on photo and music sites, and Nancy is his friend on a music and movie sites. Friends share favorite items of diverse types but usually have common tags, such as fiction and entertainment, which make personalization of various types of items feasible.

3.1.1. Basic Data Structure. The data structures of users and items that are loaded to the model are as follows.

User (Id, Profile, Friends $[\boldsymbol{id}_1, \boldsymbol{id}_2, \ldots, \boldsymbol{id}_x]$**, ViewHistory, Servers** $[\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots, \boldsymbol{s}_y]$**)**: a user may have multiple names on different MEC servers to which (s)he has registered. However, the model should use only one **Id** to identify a user. This data structure also includes private user information, e.g., age, gender, phone number, postage code, and affiliation, which can be collected from all of the servers under the users' permissions and should be stored in the user **Profile**. A user can add other users as his/her friends in the list of **Friends**, while **ViewHistory** records users' browsing histories for all types of items in the following format:

Item (Id, Category, Tags $[\boldsymbol{t}_1, \boldsymbol{t}_2, \ldots, \boldsymbol{t}_i]$**, Weight, Servers** $[\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots, \boldsymbol{s}_y]$**)**: an item must be named with an **Id** that is not equal to any existing items, while **Category** records the groups to which this item belongs, e.g., books, flowers, and stories. When an item is uploaded by a user to a **Server** for the first time, the user can add descriptions about this item with several **Tags**, e.g., romantic, comedy, and tragedy. Users can rate an item with a five-scale range, and the average rate can be calculated as **Weight**.

3.1.2. Personalization Engine. CEPTM is equipped with a personalization algorithm that calculates the recommended items in three steps. First, the algorithm determines the friends and computes their similarities. Specifically, the similarities among friends are given by the common "ratings", as defined by Pearson's correlation coefficient [34–36], which could indicate the linear correlation between two variables:

$$S_{friends} = \frac{Weight_{x,y}}{\sqrt{Weight_x^2 \times Weight_y^2}} \tag{1}$$

where $Weight_{x,y}$ is the total **Weight** of the rated items that users x and y have in common, while $Weight_x$ is the total weight of the items that are rated by user x. The **Weight** of each item is initialized to 1. Second, personalization is generated by assigning **Weight**s to items that are rated by friends:

$$\textbf{Weight} = \sum_{friends\ rate\ items} S_{friends} \tag{2}$$

where the **Weight** of a recommended item is the sum of the rates given by friends. All of the recommended items are presented to users in decreasing order of **Weight**. Finally, items that are rated by the users are removed, and an item is recommended in the first priority that has common **Tag**(s) with the removed items.

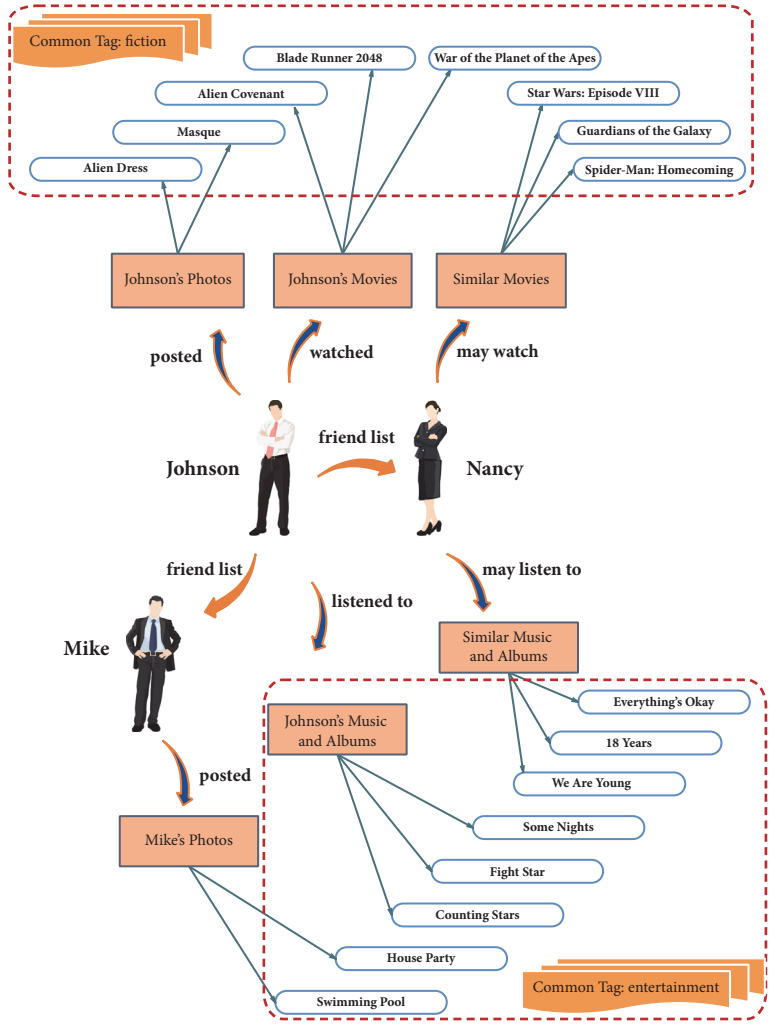


FIGURE 2: Prototype of the cross-edge model CEPTM.

3.1.3. Informing Function. The informing function can inform users about their friends' activities on all MEC edge servers, such as new posts and relevant personalization. The following is an example of a message that could be provided by CEPTM:

"Dear **Mike** (**User Id**$_{Mike}$), your friend **Nancy** (**User Id**$_{Nancy}$) has posted 'Surf all the way to the North with my paddle boat!' on **Sina Weibo** at 7:21 p.m. on January. 21, 2018."

The message contains the user's name and **Id**, his friend's name and **Id**, the post status and relevant item **paddle boat**, and the name of the **Sina Weibo**. The hyperlinks are also provided, as users may click them for message details, e.g., **Mike** (**User Id**$_{Mike}$) will direct the user "Mike" to his homepage, "Surf all the way to the North with my paddle boat!" will direct the user to the real poster on his friend Nancy's homepage, and **Weibo** will direct the user to Weibo. As a result, all of the messages that are presented by the model will contain the following information: the users' **Ids**, friends' **Ids**, friends' posts, the item **Id**, and the **Server**.

A nearby stranger could be suggested as an addable friend to the users. A user must disclose relevant private information to CEPTM to receive messages from this stranger and may add him/her as a friend later. For example, a user could provide his current location to see a message from a nearby stranger, or the model could display an upcoming motor show that will be held at the university at which a user studies if the user discloses the campus on which he is studying. A concert tour event could be displayed by CEPTM if the user is willing to reveal which style of music he likes the most. Furthermore, some information will be grayed out in the message if the person is not a friend of the user. This approach is a privacy protection strategy, but it also encourages users to add more friends. The differences between the formats of the messages from a friend and a stranger can be identified from the following:

Friend (User name & Id, **Friend name & Id**, poster & personalization)

Stranger (User name & Id, **Stranger Id**, poster & personalization)

where the stranger's hyperlink will direct the clicker to a page that asks whether the user would like to add the stranger

as a friend with the poster's information as the reference. The stranger and the user may have no overlapping MEC server, so the model may suggest that the user register first by clicking **Server**. Before the stranger accepts the user as a friend, the details of the stranger's homepage and the comments and likes will not be available to the user. All of the descriptions of recommended items, persons (users, friends, and strangers), and posters are available when clicking a corresponding link to the homepage. The user can adjust his threshold for eliminating strangers who have low similarity with him and avoiding recommended items with low scores.

3.2. Exploratory Factor Analysis. CEPTM can not only provide personalization and informing services but also analyze the latent variables that could affect a user's willingness to accept personalization. Here, we introduce the exploratory factor analysis (EFA) that will be applied in the experiment section. When the latent variables that lead to an observed item (e.g., users could accept personalization) are unknown, one solution for determining the variables is to apply an EFA to primitively identify the number of factors and their associated correlations. An EFA is used to determine the minimum volume of factors that explain the correlations among the latent variables. The dimensions are reduced when possible. The factors include common factors, unique factors, and errors. There is no correlation among the latent variables when the common factors are extracted. Factor analysis mainly focuses on the common factors. As a result, the error is usually combined with a unique factor.

$$B_{mn} = f_{n1} \times w_{n1} + f_{n2} \times w_{n2} + \cdots + f_{nx} \times w_{nx} + d_n \times u_{mn} \tag{3}$$

where B_{mn} is the standard deviation of sample m on observed item n, f_{nx} is a common factor with corresponding **Weight** w_{nx}, and d_n is the unique factor with corresponding u_{mn}. An EFA has to cover as many aspects as possible to find the factors that affect the observed item. Our method was to invite EMC mobile users to participate in an online survey, which required each person to consider "on what condition I will accept the personalization" for as much personalization as possible. The natural language processing technique was applied to the dataset of their answers to determine which content was mentioned. The code was written in Python and selected nouns as keywords in such a way that we could identify the primary concerns when a personalization was accepted. Keywords were ranked based on the number of times they were mentioned.

$$|Factor| = |keyword_1| + |keyword_2| + \cdots + |keyword_x| \tag{4}$$

For example, the respondents mentioned "satisfaction" 47 times and "interesting" 83 times, and the technique categorized both of these keywords under "user experience". Then, the score for "user experience" was calculated as 47 + 83 = 130. The participants also mentioned "security" 52 times, "fraud" 73 times, and "illegal usage" 20 times. Therefore, these three keywords were categorized under "data collection aspect", and its score was calculated as 52 + 73 + 20 = 145. If "data collection aspect" was not identified as a factor in EFA, then "satisfaction" would not be a factor either. The load factor, namely, Cronbach's Alpha, should be more than .70 for a good fit. The top-ranking (e.g., i_1-i_6 and j_1-j_4) keywords would be regarded as the indicators of the factor; see step 1 in Figure 2. Kaiser's Rule is applied in EFA to determine the number of factors, where the eigenvalues of each factor were visualized according to its **Weight** contribution to the observed item. Kaiser's rule is to drop all of the necessary components with eigenvalues that are less than 1.0 or, in other words, to consider information that is accounted for by an average single item while recommending the use of a sole cut-off criterion for estimating the number of factors as overextracted factors. The calculation is stopped when the confidence interval for each eigenvalue is greater than 1.0. Once the number of factors has been confirmed, GEOMIN, which is an oblique rotation method in EFA, is used to determine the factor structure, which should look similar to the model formation in step 2 and step 3 of Figure 3. For example, in step 2, i_3 is not defined as a key indicator of factor f_1, j_2 is not defined as a key indicator of factor f_2, *and* i_6 is not defined as a key indicator of factor f_3. Once all factors have found their determining indicators, we should know that the effects from f_3 on f_2 and f_2 on f_1 are all positive, shown in step 3.

The final structure consists of the latent variables, including the top-ranking factors and indicators, combined with the analysis methods, e.g., maximum likelihood parameter estimation and GEOMIN rotation. The output should consist of Cronbach's Alpha of each factor with the indicators and the model fitting information, e.g., values of log likelihood, Chi-Squared Test with the p value, RMSEA (Root Mean Square Error of Approximation), and GEOMIN rotated loadings and correlations.

4. Experiments

4.1. Prestudies before Setting Up CEPTM. This survey addresses the users' behaviors when managing their accounts. We regard users who have accounts in more than three MEC edge servers equal to users who are involved in more than three servers with a joint account. The total number of users who completed our survey was 860, and 89.77% of them (N = 772) indicated that they have multiple accounts in at least three MEC servers. We further asked these users for the reasons that they have so many accounts, and their answers cover five aspects: to keep in touch with friends (N = 353, 41.05%), e.g., "My best friend Nancy had registered a new account in another app, and she posts more updates than in the site to which we both registered"; to meet new people (N = 246, 28.60%), e.g., "Tim Cook mostly releases his posts on Weibo" and "I want to know my classmates better from Weibo"; to be well known (N = 394, 45.81%), e.g., "I want to find a good internship opportunities"; to try something new (N = 95, 11.05%), e.g., "The interface of the old website is so hard to use" and "So many ads in my old account"; to gain access to available resources

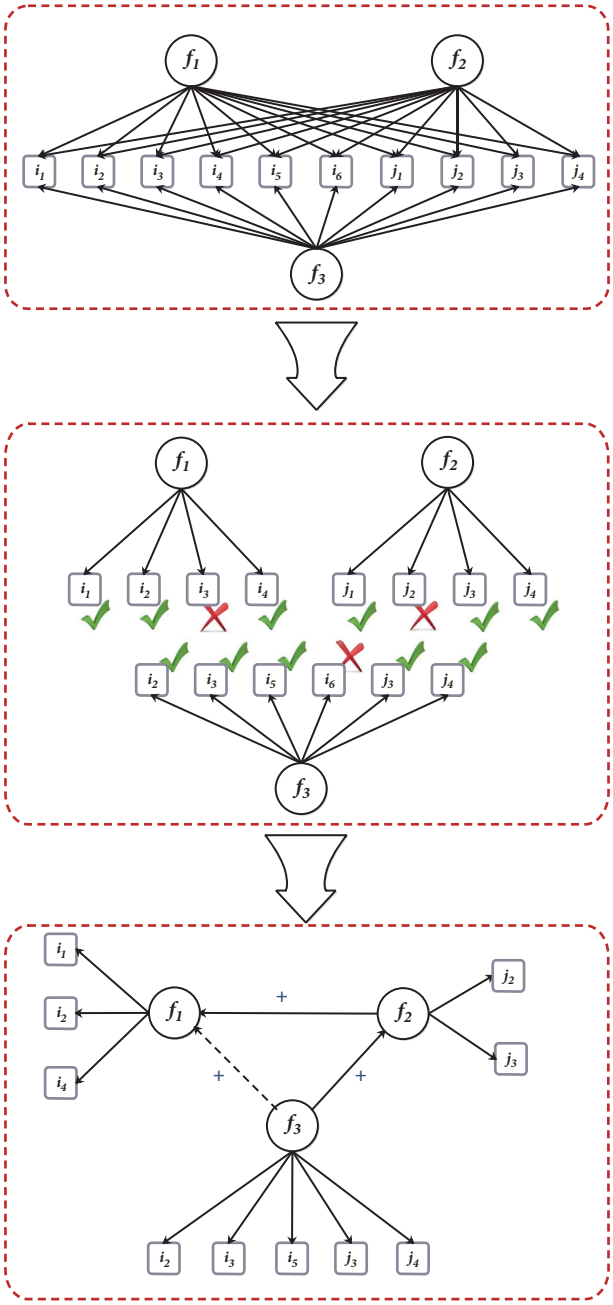


Figure 3: Structure formation and factor determination using EFA.

(N = 503, 58.49%), e.g., "Weibo provides a tremendous amount of knowledge and news for reference, and I can simply sign in with my Tencent account"; and for other reasons (N = 85, 9.88%), e.g., "For fun." From these aspects, the general reason for having multiple MEC server accounts was summarized as "being connected as much as possible". In the circumstances of cloud computing, users require several types of information for supporting daily decisions in work, study, etc. For example, to fix a car, users can browse an autocare site for suggestions. Visiting a new campus requires users to visit its college website beforehand. However, the users also gave some feedback on the flaws of having so many accounts (N = 612, 71.16%). They mainly stated that they did not have enough time to browse so many sites, e.g., "The site that I often visit is different from the one where my friends visit and post mostly", "I will not post my updates repeatedly in all of my accounts", and "I post my updates on one site only, and I sometimes miss the updates that my friend post on the other site." We conclude that users have difficulty in managing multiple accounts. Designing the CEPTM that could distribute and post users' information from all of their MEC server accounts in a cross-edge manner is truly necessary.

From the managerial perspective, a personalization service would increase its prediction accuracy if it could collect enough user information, especially when the collected information has a common **Tag**. This rule is also applicable for multiple edge servers. Compared with collecting user data from a single-edge server, collecting user data from cross-edge would increase the likelihood of locating popular topics through MEC networks, thereby guaranteeing higher personalization accuracy. For example, suppose that a user had viewed cake on a picture site, listened to romantic and peaceful songs on a music site, viewed roses and lilies on a flower app, watched romance and fiction films on a movie website, added her family members and bridesmaids as friends on Weibo, and read love and friendship tales on a book app. All of the viewed items share the common **Tag** "romantic". The personalization engine could generate an ad for wedding services accordingly, and this user would be more likely to accept this suggestion. In contrast, if the personalization engine only collected this user's actions on the flower and book sites, where the common **Tags** are "friendship" and "romantic", the prediction could miss the appropriate ad personalization. Another advantage of integrating the users' data from cross-edge is collecting more information while keeping the users' privacy concerns low. Users are likely to disclose their information only on a specific MEC site, e.g., a patient may only post his hypertension status on a medical website, rather than sharing this information with his friends on Weibo, or a person may only share her monthly income with Wells Fargo bank for a loan, while choosing not to disclose this information elsewhere. Table 1 shows that users' volumes of information disclosure in response to daily requests depend on how much they trust the information recipient. Users mostly disclose the sensitive privacy to their trusted people as expected, e.g., family members. However, the interesting thing is that users mostly disclose low sensitivity information to the people they work alongside or see only occasionally, e.g., friends and companies, regardless of whether the information was collected from a phone, a computer, or a car. A good model collects the users' data across MEC edge servers to maximize the available information for generating precise personalization. All of the items were collected under the users' permission. Another 137 MEC mobile users were invited online to register for accounts, including WeChat (Photos, http://www.wechat.com/), Kuwo Music (Music and Albums, http://www.kuwo.cn/), Youku (Movies, http://www.youku.com/), and Weibo (People, http://www.weibo.com/). Each user must view the items on each site for at least 15 minutes. The goal was to predict which

Table 1: Disclosure rates of users privacy with different recipient.

ITEMS\OPTIONS	Family	Friend	Stranger	None
Sensitive Information on portable device	153(52.76%)	115(39.66%)	13(4.48%)	106(36.55%)
Low-Sensitive Information on portable device	121(41.72%)	133(45.86%)	35(12.07%)	102(35.17%)
Low-Sensitive Information on computer	77(26.55%)	88(30.34%)	15(5.17%)	156(53.79%)
Sensitive Information on computer	88(30.34%)	62(21.38%)	18(6.21%)	162(55.86%)
Sensitive Information on connected vehicles	165(56.9%)	85(29.31%)	14(4.83%)	88(30.34%)
Low Sensitive Information on portable device	149(51.38%)	160(55.17%)	51(17.59%)	66(22.76%)

ad generated by the personalization engine users would like to view. Once a user had clicked the recommended ad, the prediction accuracy is calculated as the number of clicks multiplied by the total number of personalization. The statistics in Table 1 also show that the more servers from which user profiles were loaded as training data, the higher the probability that users accepted the generated ad personalization. This survey revealed that a model that could help them manage multiple MEC accounts is truly needed. The prestudy used approved datasets from many MEC edge servers. Thus, it could guarantee higher prediction accuracy by learning users' preferences through their browsing histories. Establishing a cross-edge model could provide mutual benefits from both the users' perspective and the managerial perspective.

4.2. Participants and Procedure in Main Experiment. We recruited 860 people from Sojump, which is a website that provides online survey services that connect more than 2 million members throughout China and enables individuals and businesses to coordinate the use of human intelligence to perform tasks that computers are currently unable to complete. For quality purposes, only adult online users who had signed into at least three of the four sites, e.g., WeChat, Kuwo, Youku, and Weibo, and spent more than 2 hours per day on these websites were allowed to participate in the main experiment. Since there is little difference among which tools users applied in browsing sites in Table 1, users can utilize any MEC device through the procedure. At least 1/3 of their friends must have multiple accounts on these websites. Of the MEC mobile users, 561 were male, 329 were between the ages of 18 and 26, 413 were between the ages of 26 and 45, and 118 were older than 45. Additionally, 278 users were working in or studying computer science or used their mobile device and computer very often, e.g., programmers and network engineers; 539 participants had received a B.S. degree or less education, 253 had received an M.A. degree, and 68 had received a Ph.D. degree or above. Each participant was required to have at least 10 common ratings with his friends, and otherwise it was suggested that he perform more ratings and attempt to join the study again to avoid the cold start problem.

The experiment was conducted for 4 days, and the performance of CEPTM demonstrated that hot topics that were raised on one site could spread to the other 3 sites over time in the MEC network. The keywords that had been clicked the most were considered to be hot top-10 topics. As shown in Figure 4, topics were segmented into 4 styles, e.g., romance, comedy, adventure, and thriller. Each style came from only one site, e.g., romance in WeChat in Figure 4(a), comedy in Kuwo Music in Figure 4(b), adventure in Youku in Figure 4(c), and thriller in Weibo in Figure 4(d). Hot topics in the other 3 sections in each site were removed in the first 24 hours. Users can update their status and forward friends' statuses within the site or across the sites. During the second 24 hours, the number of clicks on the hot topics in all 4 sites started rising. On the third day, hot topics started to migrate to the other 3 sites. During the last 24 hours, the number of clicks remained high in each section of each site and stayed equally high as the clicks on the same topics in the other 3 sites. A baseline personalization service (BLPS) was also applied in each site, which only allows forwarding friends' statuses within the site.

During the 4 days, we also compared the number of clicks between CEPTM running on the four sites (red squares and lines) and BLPS running on each site (black squares and lines). The BLPS utilizes users' profiles from a single site and recommends items by similarity computing, and items from friends who share the most common interests in hot topics are provided. The results indicate that the clicks increased much faster in CEPTM than the clicks in each BLPS, and the average number of clicks on personalization provided by CEPTM was 87.17% higher than 58.09% provided by BLPS. Therefore, CEPTM outperforms BLPS.

4.3. Results and Discussion. We examine the possible reasons behind the experimental results. First, all qualified users were asked about their personal traits, including item expertise: how much they are familiar with the item category, trust propensity: how much they trust the BLPS or CEPTM, and familiarity with online privacy and personalization: how well they understand the privacy collection. Second, they were briefly introduced to the functions of CEPTM and informed of how the personalization service works. Afterward, we administered a quiz to ensure that they understood our research target and had a basic understanding of what we were trying to accomplish. Finally, we requested 30 privacy items that belonged to one type in Table 1 from the users, and they could choose to agree to or deny the disclosures. The personalization service produced an item by similarity computing, and when a user finished the 30th request, we asked them 8 questions regarding their experience. These questions involved the users' understanding of CEPTM, their satisfaction with CEPTM, the quality of the recommended items, privacy concerns, etc. The responses to the 30 requests and eight questions on personal traits, privacy collection

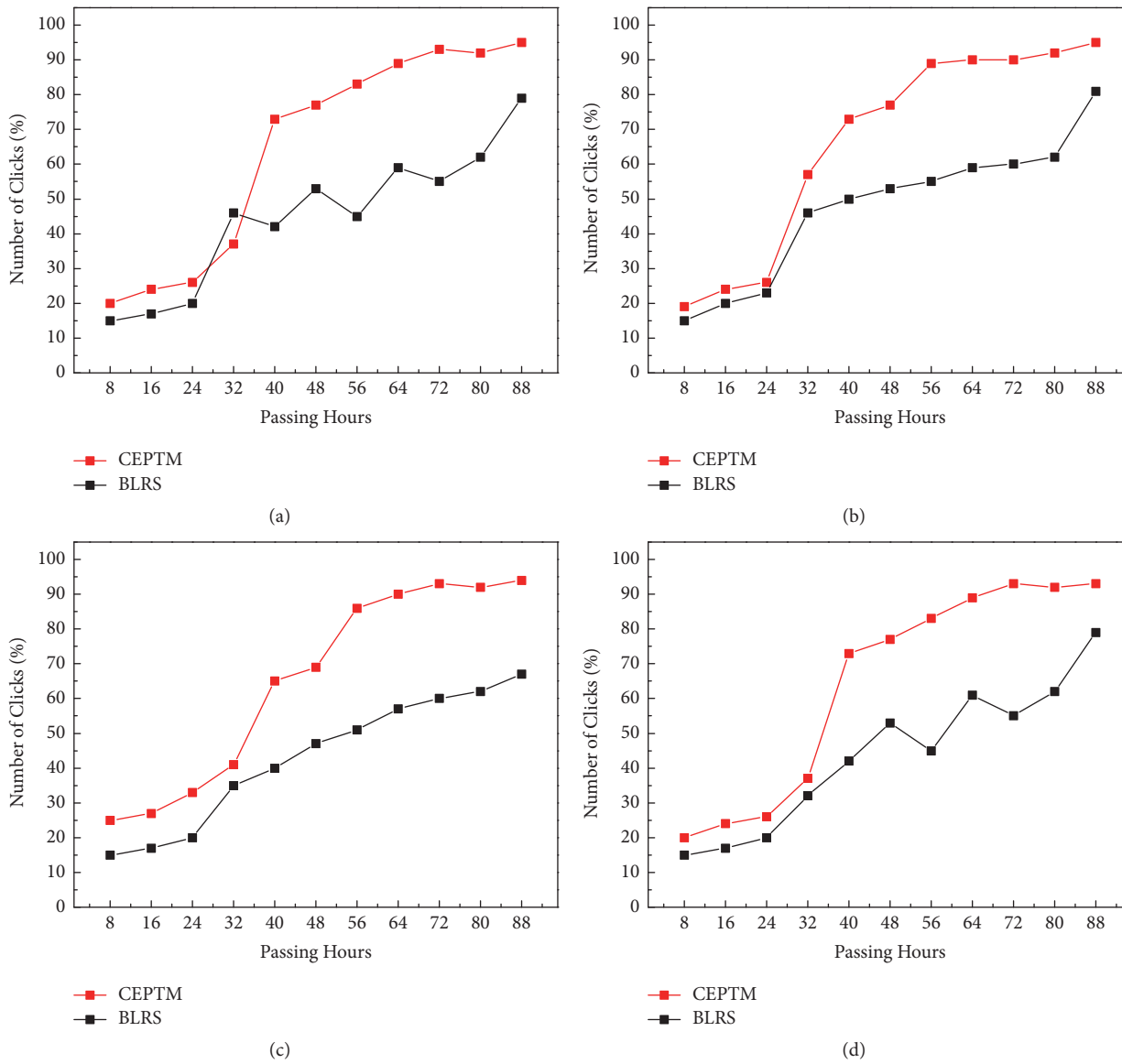


Figure 4: Four comparisons between CEPTM and BLRS running on each MEC.

aspects, user experience, and personalization quality were inputted into an exploratory factor analysis. Table 2 provides details on the results.

Table 2 lists the eight factors in the experiment that were loaded to fit the factor measurement model and the structural relations between the factors and variables in the exploratory factor analysis. The model achieved a good fit: $p < 0.01$, CFI = 0.985, TLI = 0.99, RMSEA = 0.035, SRMR = 0.631, and it revealed interesting findings among the factors, e.g., **item expertise** and **personalization diversity** had a positive effect on the users' **understandability** (+0.471, +0.273), which in turn raised the **personalization quality** (+0.371) and lowered the users' privacy concerns (-0.239). Furthermore, users' **trust propensity** had a negative effect on **satisfaction** (-0.351), while **satisfaction** increased the acceptance rate of a recommended item or stranger (+0.291). Users' **item expertise** and **familiarity with privacy and PS** both positively affected users' understanding of the cross-edge model (+0.418, +0.227). According to the above results, we describe why those factors influenced one another and briefly discuss how to manage users' information disclosure and the quality of the personalization engine. We strongly argue that users' knowledge that supported their decisions on information disclosure played an important role in this study. **Item expertise** and **familiarity with privacy and PS** determined users' **understandability**, which in turn increased the probability of accepting recommended items. This aspect further lowered users' privacy concerns in such a way that the users decided to disclose more information, and they increased their satisfaction with the personalization of the model. Users could benefit from the personalization service and the informing function, so they disclose more privacy for increased personalization quality. In this study, if users should know the underlying strategy of the privacy collection and

Table 2: Eight factors in the users' feedback on personalization services (PS) and the related parameters.

Factors	Questions	User Feedback Sample	Alpha	AVE
item expertise	4	I know the artists very well, and my friends always accept my music suggestions.	0.83	0.832
trusting propensity	4	I seldom accept a personalization from a stranger.	0.73	0.681
familiarity with privacy & PS	3	I understand that the personalized needs people's private information to generate better personalization.	0.89	0.893
understandability	3	The model provides better management of friends' posts across multiple sites.	0.75	0.709
satisfaction	5	The informing function of the model saved me time in viewing my friends' status updates on all sites.	0.90	0.738
PS quality	4	I like the recommended item.	0.78	0.712
privacy concern	5	I don't like requests that are too sensitive.	0.75	0.709
PS diversity	3	I like the personalization of a new type of item.	0.81	0.731

personalization, they might decrease their privacy concerns and increase their propensity for information disclosure. Users chased after the benefits of using our model, such as receiving high-quality personalization for unknown items, as well as being informed of their friends' real-time activities on other edge servers in MEC network. This trend was reflected in their satisfaction being directly determined by the personalization quality.

This study reveals that personalization should collect users' data while respecting their privacy concerns. This approach would allow users to share their information without experiencing worry or unnecessary anxiety. However, current privacy policy has not been designed well enough to manage and collect the users' information based on their personal traits or background knowledge. For example, a site in the MEC network will post the same set of terms and data policy to whoever signs in, even though most users do not read it. From a managerial perspective, why not design a personalized data collection policy in such a way that the users' satisfaction could be maximized? That should be an important consideration when adjusting the interactions between MEC edge servers and their users.

5. Conclusion

In this paper, we proposed a novel cross-edge model, called CEPTM, for better personalization service, and we reveal how famous topics in one resource edge server can emerge on several other destination edge servers in MEC. The prototype of the model consists of the interaction between the users and diverse items, the personalization of new items, and privacy collection and distribution strategies. Our study indicates that CEPTM (1) achieves a high rate of personalization acceptance due to the availability of a large amount of data as inputs and highly diverse personalization as outputs and (2) gains users' trust because it collects user data while respecting an individual's privacy concerns. It outperforms a baseline personalization service that is running on a single MEC edge server. The experiment results, which were performed on several real-life datasets, indicated that the trends of hot topics on a resource edge server could migrate to other edge servers in an MEC environment. An exploratory factor analysis revealed that personalization of diverse types is more likely to be accepted than single-type personalization and that helping users gain more knowledge and understanding can increase their disclosure of personal information, which enables the personalization service to predict items better, with higher accuracy. Additionally, CEPTM has provided new insight into handling the cold start problem in cloud computing, to which scholars and personalized providers can refer. In future work, we will continue to focus on user-tailored personalization, which covers both providing items that fit users' preferences and requesting users' disclosures according to their individual privacy concerns in MEC networks. Hopefully, additional interesting results will be discovered.

Data Availability

Users' data for the study are collected from our previous work [37, 38], and users who have at least 3 server accounts in EMC can join the study in Section 5.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 61702312), the Natural Science Foundation of Shandong Province of China (No. ZR2017BF019), and the project of Shandong Province Higher Educational Science and Technology Program (No. J17KB178); and Hongchen Wu acknowledges the financial support from the China Scholarship Council (CSC, File No. 201306220132).

References

[1] R. Roman, J. Lopez, and M. Mambo, "Mobile edge computing, Fog et al.: A survey and analysis of security threats and challenges," *Future Generation Computer Systems*, vol. 78, part 2, pp. 680–698, 2016.

[2] Z. Wang, Q. Zhao, F. Xu, H. Dai, and Y. Zhang, "Detection performance of packet arrival under downclocking for mobile edge

computing," *Wireless Communications and Mobile Computing*, vol. 2018, Article ID 9641712, 7 pages, 2018.

[3] Z. Ruobin, Z. Weihua, and H. Jiang, "Scalable multiple description coding and distributed video streaming in 3G mobile communications," *Wireless Communications and Mobile Computing*, vol. 5, no. 1, pp. 95–111, 2005.

[4] T. Verhagen, J. van Nes, F. Feldberg, and W. van Dolen, "Virtual customer service agents: Using social presence and personalization to shape online service encounters," *Journal of Computer-Mediated Communication*, vol. 19, no. 3, pp. 529–545, 2014.

[5] P. M. Herceg, T. B. Allison, R. S. Belvin, and E. Tzoukermann, "Collaborative exploratory search for information filtering and large-scale information triage," *Journal of the Association for Information Science and Technology*, vol. 69, no. 3, pp. 395–409, 2018.

[6] J. Teevan, S. T. Dumais, and E. Horvitz, "Personalizing Search via Automated Analysis of Interests and Activities," *ACM SIGIR Forum*, vol. 51, no. 3, pp. 10–17, 2018.

[7] B. Loepp, K. Herrmanny, and J. Ziegler, "Blended recommending: Integrating interactive information filtering and algorithmic recommender techniques," in *Proceedings of the 33rd Annual CHI Conference on Human Factors in Computing Systems, CHI 2015*, pp. 975–984, Seoul, Republic of Korea, April 2015.

[8] Y. Li, C. Zhai, and Y. Chen, "Exploiting rich user information for one-class collaborative filtering," *Knowledge and Information Systems*, vol. 38, no. 2, pp. 277–301, 2014.

[9] G. Guo, J. Zhang, and D. Thalmann, "Merging trust in collaborative filtering to alleviate data sparsity and cold start," *Knowledge-Based Systems*, vol. 57, pp. 57–68, 2014.

[10] S. Guizani, "Relay attacks concerns in wireless ad hoc, sensors, and RFID networks," *Wireless Communications and Mobile Computing*, vol. 16, no. 11, pp. 1431–1435, 2016.

[11] G. Xu, Q. Wu, M. Daneshmand, Y. Liu, and M. Wang, "A data privacy protective mechanism for wireless body area networks," *Wireless Communications and Mobile Computing*, vol. 16, no. 13, pp. 1746–1758, 2016.

[12] B. Paudel, F. Christoffel, C. Newell, and A. Bernstein, "Updatable, accurate, diverse, and scalable recommendations for interactive applications," *ACM Transactions on Interactive Intelligent Systems (TiiS)*, vol. 7, no. 1, 2016.

[13] D. Margaris, C. Vassilakis, and P. Georgiadis, "Query personalization using social network information and collaborative filtering techniques," *Future Generation Computer Systems*, vol. 78, pp. 440–450, 2018.

[14] X. Bai, B. Barla Cambazoglu, F. Gullo, A. Mantrach, and F. Silvestri, "Exploiting search history of users for news personalization," *Information Sciences*, pp. 125–137, 2017.

[15] Z. Yu, C. Wang, J. Bu, X. Wang, Y. Wu, and C. Chen, "Friend recommendation with content spread enhancement in social networks," *Information Sciences*, vol. 309, pp. 102–118, 2015.

[16] S. Wang, Y. Zhao, L. Huang, J. Xu, and C.-H. Hsu, "QoS prediction for service recommendations in mobile edge computing," *Journal of Parallel and Distributed Computing*, 2017, In press.

[17] T. Spyropoulos and P. Sermpezis, "Soft cache hits and the impact of alternative content recommendations on mobile edge caching," in *Proceedings of the the Eleventh ACM Workshop*, pp. 51–56, New York City, NY, USA, October 2016.

[18] D. Kim, C. Park, J. Oh, and H. Yu, "Deep hybrid recommender systems via exploiting document context and statistics of items," *Information Sciences*, vol. 417, pp. 72–87, 2017.

[19] L. Yang and X. Amatriain, "Recommending the world's knowledge: Application of recommender systems at Quora," in *Proceedings of the 10th ACM Conference on Recommender Systems, RecSys 2016*, p. 389, Boston, Mass, USA, September 2016.

[20] E. Aguirre, D. Mahr, D. Grewal, K. de Ruyter, and M. Wetzels, "Unraveling the personalization paradox: The effect of information collection and trust-building strategies on online advertisement effectiveness," *Journal of Retailing*, vol. 91, no. 1, pp. 34–49, 2015.

[21] Y. Sun, N. Wang, X.-L. Shen, and J. X. Zhang, "Location information disclosure in location-based social network services: Privacy calculus, benefit structure, and gender differences," *Computers in Human Behavior*, vol. 52, pp. 278–292, 2015.

[22] J.-M. Lee and J.-Y. Rha, "Personalization-privacy paradox and consumer conflict with the use of location-based mobile commerce," *Computers in Human Behavior*, vol. 63, pp. 453–462, 2016.

[23] S. Meyffret, L. Médini, and F. Laforest, "Trust-based local and social recommendation," in *Proceedings of the 4th ACM RecSys Workshop on Recommender Systems and the Social Web, RSWeb 2012*, pp. 53–60, Dublin, Ireland, September 2012.

[24] G. Liu, Y. Wang, M. A. Orgun, and E.-P. Lim, "Finding the optimal social trust path for the selection of trustworthy service providers in complex social networks," *IEEE Transactions on Services Computing*, vol. 6, no. 2, pp. 152–167, 2013.

[25] E. Toch, N. M. Sadeh, and J. Hong, "Generating default privacy policies for online social networks," in *Proceedings of the 28th Annual CHI Conference on Human Factors in Computing Systems, CHI 2010*, pp. 4243–4248, Atlanta, Ga, USA, April 2010.

[26] T. De Pessemier, K. Vanhecke, and L. Martens, "A hybrid strategy for privacy-preserving personalizations for mobile shopping," in *Proceedings of the the 8th ACM Conference on Personalized System*, pp. 22–25, 2014.

[27] C. Brodie, C.-M. Karat, J. Karat, and J. Feng, "Usable security and privacy: A case study of developing privacy management tools," in *Proceedings of the Symposium on Usable Privacy and Security, SOUPS 2005*, pp. 35–43, Pittsburgh, Pa, USA, July 2005.

[28] Y. Li and Y. Guo, "Cultural distance-aware service recommendation approach in mobile edge computing," *Scientific Programming*, vol. 2018, Article ID 2181974, 8 pages, 2018.

[29] J. Yang, H. Wang, Z. Lv, and H. Song, "Multimedia personalization and transmission system based on cloud platform," *Future Generation Computer Systems*, vol. 70, 2016.

[30] Y. Zhang, "Browser-oriented universal cross-site recommendation and explanation based on user browsing logs," in *Proceedings of the the 8th ACM Conference*, pp. 433–436, Foster City, Silicon Valley, Calif, USA, October 2014.

[31] A. Barth, C. Jackson, and J. C. Mitchell, "Robust defenses for cross-site request forgery," in *Proceedings of the 15th ACM conference on Computer and Communications Security, CCS'08*, pp. 75–87, Alexandria, Va, USA, October 2008.

[32] B. Li, "Cross-domain collaborative filtering: A brief survey," in *Proceedings of the 23rd IEEE International Conference on Tools with Artificial Intelligence, ICTAI 2011*, pp. 1085-1086, Boca Raton, Fla, USA, November 2011.

[33] W. Pan, E. W. Xiang, N. N. Liu, and Q. Yang, "Transfer learning in collaborative filtering for sparsity reduction," in *Proceedings of the 24th AAAI Conference on Artificial Intelligence and the 22nd Innovative Applications of Artificial Intelligence Conference, AAAI-10 / IAAI-10*, pp. 230–235, Atlanta, Ga, USA, July 2010.

[34] H. Zhou, Z. Deng, Y. Xia, and M. Fu, "A new sampling method in particle filter based on Pearson correlation coefficient," *Neurocomputing*, vol. 216, pp. 208–215, 2016.

[35] Q. Zou, J. Zeng, L. Cao, and R. Ji, "A novel features ranking metric with application to scalable visual and bioinformatics data classification," *Neurocomputing*, vol. 173, part 2, pp. 346–354, 2016.

[36] N. Altman and M. Krzywinski, "Points of Significance: Association, correlation and causation," *Nature Methods*, vol. 12, no. 10, pp. 899-900, 2015.

[37] H. Wu, B. P. Knijnenburg, and A. Kobsa, "Improving the prediction of users' disclosure behavior… by making them disclose more predictably?" in *Proceedings of the SOUPS2014 Workshop on Privacy Personas and Segmentation*, Menlo Park, Calif, USA, 2014.

[38] H. Wu, H. Zhang, L. Cui, and X. Wang, "A heuristic model for supporting users decision-making in privacy disclosure for recommendation," *Security & Communication Networks*, vol. 4, pp. 1–13, 2018.

Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 3610482, 13 pages
https://doi.org/10.1155/2018/3610482

Research Article

A Novel Real-Time Image Restoration Algorithm in Edge Computing

Xingmin Ma,[1] Shenggang Xu,[1] Fengping An,[2] and Fuhong Lin[1]

[1]*School of Computer and Communication Engineering, University of Science and Technology Beijing, Beijing 100083, China*
[2]*Huaiyin Normal University, Huai'an 223001, China*

Correspondence should be addressed to Fuhong Lin; fhlin@ustb.edu.cn

Received 18 April 2018; Accepted 9 July 2018; Published 9 August 2018

Academic Editor: Ao Zhou

Owning to the high processing complexity, the image restoration can only be processed offline and hardly be applied in the real-time production life. The development of edge computing provides a new solution for real-time image restoration. It can upload the original image to the edge node to process in real time and then return results to users immediately. However, the processing capacity of the edge node is still limited which requires a lightweight image restoration algorithm. A novel real-time image restoration algorithm is proposed in edge computing. Firstly, 10 classical functions are used to determine the population size and maximum iteration times of traction fruit fly optimization algorithm (TFOA). Secondly, TFOA is used to optimize the optimal parameters of least squares support vector regression (LSSVR) kernel function, and the error function of image restoration is taken as an adaptive function of TFOA. Thirdly, the LLSVR algorithm is used to restore the image. During the image restoration process, the training process is to establish a mapping relationship between the degraded image and the adjacent pixels of the original image. The relationship is established; the degraded image can be restored by using the mapping relationship. Through the comparison and analysis of experiments, the proposed method can meet the requirements of real-time image restoration, and the proposed algorithm can speed up the image restoration and improve the image quality.

1. Introduction

With the development of science and technology, more and more users use their mobile phones for photography. During imaging of the digital image, it is easy to produce image blur due to jitter, light, and many other factors [1], and this phenomenon is called image degradation. Therefore, an effective parameter setting needs to be performed in image restoration processing, which consumes a large amount of resources. The current solution is to use offline servers and other types of processing equipment, which cannot be done in real time. The emergence of edge computing makes real-time processing of such applications possible. Edge computing can meet the user's real-time computing service requests and provides users with low-latency image processing services. The local image is uploaded to edge nodes for real-time processing. We can deploy the Traction Fruit fly Optimization Algorithm (TFOA) and Least Squares Support Vector Regression (LLSVR) to edge nodes. However, the processing capacity of the edge node is limited, so it is necessary to propose a lightweight image restoration algorithm. In this way, the photos taken by the mobile phone can be processed in real time in the edge node.

Image restoration is the reverse process of image degradation. And its ultimate goal is to remove blur and interference, so that the image is restored as close as possible to the original one [2]. One of the most typical image blur models is Gaussian blur. Gaussian point spread functions are common in many image acquisition, measurement, and transmission systems. So, many degradation processes can be approximated by Gaussian blur models. The study of Gaussian blurred image restoration is of great significance to other types of degraded image restoration [3]. Most of the traditional restoration methods are deconvolution processes, such as inverse filtering, wiener filtering, least square method, and Lucy-Richardson (LR), which are commonly referred to as nonblind restoration methods [4, 5]. These methods all need to know the point spread function of imaging system,

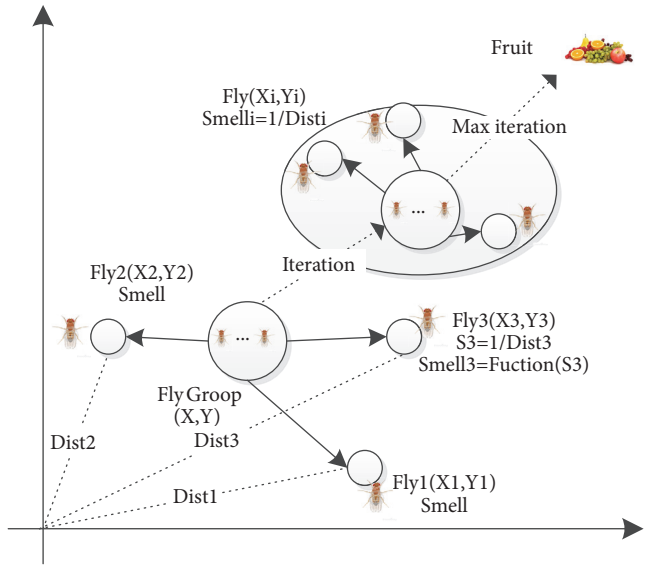


Figure 1: Population iterative search for food in fruit fly.

which is very difficult in practical application. In contrast, blind restoration method uses the information carried by the image itself or the degradation model estimation under the condition that the point spread function is unknown and combines the additional conditions to solve the problem [6]. This kind of method is widely used at present, but the result is not stable, and the calculation is large and the convergence is slow.

With the rise of in-depth learning in recent years, various machine intelligence models, such as BP neural network and support vector machine, have been widely used in the field of image restoration. In 2011, Mukherjee et al. used a Multilayered Quantum Backpropagation Neural Network for Image Restoration [7]; Sethy et al. proposed an image restoration method based on BP neural network, which has achieved good results, but there is still room for improvement [8]. References [9, 10] use fuzzy neural network model for image restoration and the image quality has been improved to some extent, but there is still room for optimization. Reference [11] proposes image restoration based on support vector regression (SVR), which improve convergence accuracy and achieves better restoration effect. However, in the optimization process of SVM parameters using FOA, the two internal parameters are not optimized, and only empirical values are given. Based on the existing work, this paper proposes an image restoration method based on TFOA-LSSVR. Through the comparison and analysis of experiments, the proposed method can meet the requirements of real-time image restoration.

2. Background Materials

2.1. Fruit Fly Optimization Algorithm (FOA). FOA is an algorithm for finding food sources based on fruit fly, which is a globally optimized algorithm [12]. The olfactory and visual organs of fruit fly are very developed. It first collects the smell of food while foraging, the position where food and others companions congregate can also be visually detected after approaching the food location and fly in this direction. A schematic diagram of the iterative foraging process for fruit fly populations is shown in Figure 1. The algorithm combines the global search of population with individual information exchange. On that basis of a global search, global optimal solution is updated through individual information exchange. And finally the algorithm is terminated under the condition that either the maximum iteration number or the convergence target precision is met. The steps of FOA are as follows.

Step 1. The population size is expressed by *sizepop* and the maximum iteration number is expressed by max *gen*. The random initial position of fruit fly is expressed by X_axis, Y_axis.

Step 2. Give fruit fly individuals a random direction and distance to use their sense of smell to search for food; *StepLength* is the search distance, namely, the iterative step value; that is,

$$X(i) = X\_axis + StepLength \tag{1}$$

$$Y(i) = Y\_axis + StepLength \tag{2}$$

Step 3. Since the initial position of the food is unknown, we first estimate the distance $Dist_i$ between individual and the origin and then set the smell concentration S_i of individual, which we define here as the reciprocal of the distance.

$$Dist_i = \sqrt{X_i^2 + Y_i^2} \tag{3}$$

$$S_i = \frac{1}{Dist_i} \tag{4}$$

Step 4. The smell concentration value S_i brought into the smell concentration determination function to calculate the smell concentration $Smell_i$;

$$Smell_i = Function(S_i) \tag{5}$$

Step 5. Find the individual with the best odor concentration in the population;

$$[bestsmell, bestindex] = \min(bestsmell) \tag{6}$$

Step 6. The optimum smell concentration value of *bestsmell* and coordinates X, Y are recorded and retained; this time, the fruit fly population uses vision to fly to that location;

$$X\_axis = X(bestindex) \tag{7}$$

$$Y\_axis = Y(bestindex) \tag{8}$$

$$smellbest = bestsmell \tag{9}$$

Step 7. Enter iteration optimization, repeatedly execute Steps 2 to 5, and judge whether the optimal taste concentration is better than the optimal smell concentration of the previous iteration or not and the current iteration number is less than the maximum iteration number $\max gen$; if yes, Step 6 is executed.

This paper introduces a new FOA, which is TFOA [13]. The optimization algorithm has no change on the two main parameters. Instead, the direction of iteration is strategically optimized, and it solves the problem of falling into a local optimal solution in the solving process. The results of each iteration are recorded. When the algorithm is locally optimal, *optimalpop* represents the best individual [13, 14], and *badPop* represents the worst individual in each iteration. The size of the Drosophila population is represented by *sizepop*. The final direction of the iteration is represented by the following formula:

$$direction = \begin{cases} 1 & optimalpop^j > badPop_i^j \\ -1 & optimalpop^j < badPop_i^j \\ RandomDirection & optimalpop^j = badPop_i^j \end{cases} \tag{10}$$

The new individual generation method is as shown in (11).

$$individual_i^m = optimalpop^m + (randius * RandDirection) \tag{11}$$

$$individual_0 = rand(1, \dim ension)$$

$$individual_i = individual_{i-1} * (E - individual_{i-1}) * 4 \tag{12}$$

In (12), the entire vector is 1.

The main steps of the TFOA [13] are as follows.

Step 1. The Drosophila population is produced according to (11), that adaptability of the Drosophila individual is calculated, and the worst individual is recorded.

Step 2. The optimal smell concentration and initial position of the population were changed. In this process, if the smell concentration is consistent, the iteration is continued and the count is increased.

Step 3. During the iteration, if the algorithm ends prematurely, (11) is executed to jump out of local optimization. When a better fruit fly is discovered, the population's location is updated and the search continues around it. If it ends too quickly, expand the search radius and continue iterative optimization. When the traction operation is complete, the worst person is rerecorded in each iteration process.

Step 4. Steps 1 to 3 are repeatedly executed, and whether the optimal taste concentration is better than the optimal taste concentration of the previous iteration is judged, until meeting the requirements of the algorithm.

2.2. Support Vector Machine (SVM). SVM is developed from statistical theory [15]. LSSVR is developed from SVM through special treatment [11]; LSSVR is suitable for regression model of static nonlinear function estimation. SVM algorithm has been applied to classification problems since it was proposed; now in the classification direction is very mature, the classification method is effectively extended, and the current extension is also very effective in fitting curves and nonlinear regression estimation.

Trained array $\{(x_i, y_i), i = 1, 2, \cdots, l\}$, the $i'th$ value in this array combination, is representing $x_i \in R^d$, it is known that there are vectors $x_i = (x_i^1, x_i^2, \cdots, x_i^d)$ of d dimensions at the same time, and that output result is set to $y_i \in R$. The missing parameter ε is

$$|y - f(x)|_\varepsilon = \begin{cases} 0 & |y - f(x)| \le \varepsilon \\ |y - f(x)| - \varepsilon & |y - f(x)| > \varepsilon \end{cases} \tag{13}$$

When the loss parameter is 0, assuming that the difference between the estimated value and the actual value of the final result obtained by training the $f(x)$ into the support vector machine is smaller than the lost parameter, finding a proper kernel function $k(x_i, x_j) = \varphi(x_i) \cdot \varphi(x_j)$ and mapping the kernel function $k(x_i, x_j) = \varphi(x_i) \cdot \varphi(x_j)$ to a linear high-dimensional space for regression prediction, assume here that the function is

$$f(x) = \omega \cdot \varphi(x) + b \tag{14}$$

In formula (14), $\omega \in R^d$ is expressed as the weight value of the vector, $b \in R$ represents the threshold calculated for the data, and $(\cdot)$ is expressed as the inner product operation of the data. The value of b and ω is finally obtained. Error function under the premise that meets $\omega^T\omega/2$ minimizes ε, ξ_i, ξ_i^* are added to the optimization calculation, and the final expression is

$$\min_{w,b,\xi_i,\xi_i^*} \quad \frac{\|w\|^2}{2} + C\sum_{i=1}^{l}(\xi_i + \xi_i^*) \tag{15}$$

$$\text{s.t.} \quad y_i - (w \cdot \varphi(x_i) + b) \le \varepsilon + \xi_i \tag{16}$$

$$(w \cdot \varphi(x_i) + b) - y_i \le \varepsilon + \xi_i^* \tag{17}$$

$$\xi_i, \xi_i^* \ge 0, \ i = 1, 2, \cdots, l \tag{18}$$

$$\max \quad \left[-\frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}(a_i - a_i^*)(a_j - a_j^*)k(x_i, x_j) - \varepsilon\sum_{i=1}^{l}(a_i + a_i^*) + \sum_{i=1}^{l} yi(a_i - a_i^*) \right] \tag{19}$$

$$\text{s.t.} \quad \sum_{i=1}^{l}(a_i - a_i^*) = 0 \tag{20}$$

$$0 \le a_i, a_i^* \le C \tag{21}$$

It is assumed here that the penalty parameter C is a normal number. The above problems are solved optimally, support vector $(a_i - a_i^*) \neq 0$, and $w = \sum_{i=1}^{l}(a_i - a_i^*)\varphi(x_i)$ Then, there is

$$b = \frac{1}{N_{Nsv}}\left\{ \sum_{0\le a_i\le C}\left[y_i - \sum_{x_j \subset sv}(a_j - a_j^*)k(x_j, x_i) - \varepsilon \right] + \sum_{0\le a^*{}_i\le C}\left[y_i - \sum_{x_j\subset sv}(a_j - a_j^*)k(x_j, x_i) + \varepsilon \right] \right\} \tag{22}$$

The numb of standard support vectors in (22) is N_{Nsv}; obtain a b threshold and a regression function:

$$f(x) = \sum_{x_i \in sv}(a_i - a_i^*)k(x_i, x) + b \tag{23}$$

There are three kernel functions commonly used in support vector machines:

First: kernel function polynomial

$$K(x_i, x_j) = \left[(x_i \cdot x_j{}') + 1\right]^q \tag{24}$$

The coefficient of the polynomial in (24) is q, the value is determined by the application itself, and the parameter is adjusted according to the actual situation.

Second, the radial basis function (RBF kernel function)

$$K(x_i, x_j) = \exp\left\{ -\frac{|x_i - x_j|^2}{2\sigma^2} \right\} \tag{25}$$

The core parameter in (25) is σ; it uses Gaussian kernel function of mean square error; the larger the value of σ, the wider the width of the Gaussian function.

Third, Sigmoid kernel function is as follows:

$$K(x_i, x_j) = \frac{1}{\left(1 + \exp\left(-|x_i - x_j|\right)\right)} \tag{26}$$

The LSSVR represents the following:

$$f(X) = \sum_{i=1}^{n} a_i K(x, x') + b \tag{27}$$

$$K(x_i, x_j) = \exp\left\{ -\frac{|x_i - x_j|^2}{2\sigma^2} \right\} \tag{28}$$

We need to optimize the selection of parameter C and δ before conducting sample training.

3. The Proposed Methodology

3.1. Process of Image Restoration Processing. Assuming that the original image is $f(x, y)$, the degraded image may be expressed as

$$g(x, y) = H[f(x, y)] + \eta(x, y) \tag{29}$$

In (29), H denotes a degraded image, and $\eta(x, y)$ denotes additive noise.

If H is a linear, spatially invariant process, the degraded image may be represented in the spatial frequency domain as

$$g(x, y) = f(x, y) * h(x, y) + \eta(x, y) \tag{30}$$

In (30), $*$ is a convolution operation, $f(x, y)$, $h(x, y)$, $\eta(x, y)$ are original image, degraded image, and additive noise. Image restoration is a parameter estimation process. When the fuzzy kernel function is Gaussian, the degeneration process is called Gaussian blur. This type of blurred image can be seen as a result of convolution operation between the convolution kernel of Gaussian distribution and the clear image. Namely, that pixel value of a certain point of the image

and the surrounding pixel value are weighted and averaged according to the weight of the Gaussian distribution to obtain the pixel value after the position is degraded.

3.2. Optimization Algorithm of TFOA Based on LSSVR. In this paper, TFOA-LSSVR is introduced into the mapping relationship model between images. In the process of image restoration, the selection criteria of the original image and the blurred image are consistent. Assuming that the size of the image is $m \times n$, a $i-1$ window of an original image corresponds to the corresponding center pixel in the process of constructing a training sample. If there is a degraded image completely recorded by the near window, we can get $(m-i+1)\times(n-i+1)$ support vectors in the process of degradation, and vector dimension is $i^2 \times 1$. We represents this vector with X_i and the pixel i with Y_i, then image restoration steps are as follows [12].

Step 1. We initialize the TFOA.

Step 2. We set up a two-dimensional space. According to Section 2.1, we have the following formula:

$$\begin{aligned} D(i,1) &= \sqrt{X(i,1)^2 + Y(i,1)^2} \\ D(i,2) &= \sqrt{X(i,2)^2 + Y(i,2)^2} \end{aligned} \tag{31}$$

$$\begin{aligned} S(i,1) &= \frac{1}{D(i,1)} \\ S(i,2) &= \frac{1}{D(i,2)} \end{aligned} \tag{32}$$

Step 3. The mean square error (MSE) function is used as the odor concentration function of TFOA when a mapping relationship is established between the original image and the degraded image.

$$MSE = \frac{\sum_{i=1}^{mn} \left(\hat{t} - t\right)^2}{mn} \tag{33}$$

In (33), $\hat{t}$ represents the original image, t represents a restored image, the size of the image is $m \times n$, and the parameter $[C, \delta]$ in the least squares support vector machine is represented by the parameter $[S, (i,1), S(i,2)]$.

Step 4. The odor concentration value is calculated by comparing the calculated minimum odor concentration and bringing this value into the LSSVR model, while preserving the optimal location.

Step 5. Carrying out iteration optimization, repeating Steps 2-4, and stopping iteration optimization when the iteration number reaches the maximum, the parameters $[S, (i,1), S(i,2)]$ are obtained, and then $[C, \delta]$ are obtained.

After the training sample is completed, the corresponding picture attribute is created by using the same picture size when a new degraded image needs to be restored. Through the LSSVR model, the pixel value of the image is predicted, and finally the pixel value of the image is displayed by pixels.

4. Experiments and Applications

4.1. Parameter Optimization Analysis of TFOA. The server was simulated in Windows 7 operating system (i5-3210M, 2.5 GHz CPU, 8.00 GB), and the TFOA algorithm was implemented using Matlab2014a. The simulation experiment is carried out by taking the minimum value of 10 benchmark functions [14, 16] as an example. The effects of parameters on the convergence accuracy and speed of the algorithm were analyzed and compared. Finally, the appropriate *sizepop* and max *gen* were determined. Name, function form, search interval, and function optimal value of 10 benchmark functions are in Table 1. The picture material used in this paper comes from [17].

4.1.1. Effect of Population Size on Algorithm Performance. In the experiment, we use progressive approaches to population size *sizepop*. The final experimental results were analyzed to find out the relationship among the population size, the convergence accuracy, and time complexity of the algorithm. The parameters were set as follows: population size separate values (5,10,15,20); max *gen* is 150 times; optimization iterative step is valued by TFOA algorithm. The search interval is shown in Table 1. The optimal mean value, average running time, and relative change rate of convergence accuracy of TFOA global optimization function are taken as evaluation indexes. The experimental results of 10 test functions after 50 consecutive runs are shown in Table 2.

In Table 2, optimized mean represents the arithmetic mean of the minimum value of the global optimization function; it reflects the convergence accuracy of the algorithm: the smaller the value, the higher the convergence accuracy of the algorithm. Average operation time is the arithmetic average of the algorithm's run time, which is the average time required for the algorithm to run once. Convergence condition refers to optimal mean corresponding to low population size-optimal mean corresponding to high population size/optimal mean corresponding to low population size. It shows that the convergence accuracy of the algorithm increases with the increase of population size, and the larger the value, the greater the convergence accuracy of the algorithm. At the same time, as we can see from Table 2, with the increase of *sizepop*, the convergence accuracy of the algorithm is improved, and the average running time is proportional to the increase; however, with the increase of population size, the relative rate of change of convergence accuracy shows a downward trend. The trend line of optimized mean of 10 test functions with increasing population size is shown in Figure 2. In the figure, the ordinate is expressed by optimized mean and the abscissa is population size.

As we can see from Figure 2, at the initial stage of population size increase, the optimized mean decreases monotonically with the increase of population size, and the relative rate of change is the largest; that is, the convergence accuracy of the algorithm is the largest. In the middle stage of population size increase, the optimized mean continues to decrease monotonically with increasing population size, but the relative rate of change decreases; at the later stage of population increase, on functions F4, F5, and F8, optimized

Table 1: The classical test functions.

Function ID	Function Name	Equation	Function Type	Dimension	Bounds of X
F1	Classic	$f_1(x,y) = -x\cos(2py) - y\sin(2px)$	Multimodal	2	UB(2) LB(-2)
F2	Bohachevsky	$f_2(x,y) = x^2+y^2-0.3*\cos(3\pi x)+0.3*\cos(4\pi y)+0.3$	Multimodal	2	UB(10) LB(-10)
F3	Step	$f_3(x) = \sum_{i=1}^{n}(\lfloor x_i + 0.5\rfloor)^2$	Unimodal	30	UB(100) LB(-100)
F4	Easom	$f_4(x) = -\cos x \times \cos y \times e^{-(x-\pi)^2-(y-\pi)^2}$	Multimodal	2	UB(10) LB(-10)
F5	Sphere	$f_5(x) = \sum_{i=1}^{n} x_i^2$	Unimodal	30	UB(100) LB(-100)
F6	Rastrigin	$f_6(x) = \sum_{i=1}^{n}(x_i^2 - 10\cos(2\pi x_i) + 10)$	Multimodal	30	UB(5.12) LB(-5.12)
F7	Quartic	$f_7(x) = \sum_{i=1}^{n} i x_i^4 + \text{rand}()$	Unimodal	30	UB(1.28) LB(-1.28)
F8	Sum squares	$f_8(x) = \sum_{i=1}^{n} i x_i^2$	Unimodal	30	UB(100) LB(-100)
F9	Ackley	$f_9(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e$	Multimodal	30	UB(32) LB(-32)
F10	Griewank	$f_{10}(x) = \frac{1}{4000}\sum_{i=1}^{n}x_i^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	Multimodal	30	UB(100) LB(-100)

mean continues to decrease with increasing population size, but the relative rate of change is also smaller. Therefore, the synthesis of the above analysis can draw the following two conclusions: Firstly, with the increase of population size, the time cost of the algorithm also increases. Secondly, with the increase of population size, the convergence accuracy of the algorithm is improved; however, with the increase of population size, the convergence accuracy of the algorithm cannot be improved significantly, because it is easy to fall into local optimum at the later stage of algorithm iteration. Therefore, the appropriate population size can find the best balance between algorithm performance and running time. In this paper, 15 is chosen as the population size of TFOA.

4.1.2. Influence of Maximum Iteration Number on Algorithm Performance. In the experiment, we use progressive approaches to population size max *gen*; the parameters were set as follows: max *gen* separate values are 10, 50, 150, 500, and 1000, *sizepop* is 15, and the search interval is shown in Table 1. The optimal mean value obtained from the minimum value of the global optimization function and the running times of the algorithm are taken as evaluation indexes. The experimental results of 10 test functions after 50 consecutive runs are shown in Table 3, the larger max *gen*, the higher the accuracy of the optimized mean and the higher the average running time. The trend line of the optimized mean values of the 10 test functions as max *gen* increases is shown in Figure 3, with the ordinate indicated by optimized mean, and the abscissa is maximum iteration number.

In Figure 3, we can draw the following two conclusions: Firstly, increasing the max *gen* will inevitably increase the computation time consumed by the algorithm. Secondly, the convergence accuracy of the algorithm can be improved by increasing the max *gen*, but that continuous increase of the maximum iteration numb does not lead to the continuous and obvious improvement of the convergence precision of the algorithm; this is because it is easy to fall into local optimum at the later stage of algorithm iteration. Therefore, the appropriate maximum number of iterations can find the best balance between algorithm performance and runtime. The max *gen* used in this paper is 500.

4.2. Image Restoration Analysis of LSSVM- TFOA. The server was simulated in edge computing. In order to make the algorithm convincing, we add different types of blur and noise to the standard image for testing. In the experiment, we use the size 512 × 512 for both the training image and the test image through normalization processing. Use the optimization results in the previous section that *sizepop* is 15 and max *gen* is 500. The initial position of Drosophila is random. In order to evaluate the image restoration effect quantitatively, the peak signal-to-noise ratio (PSNR) and

Table 2: Effect of population size on algorithm performance.

Function	Evaluation index	*sizepop*			
		5	10	15	20
F1	Optimized mean	4.450	4.398	4.381	4.380
	Average operating time (s)	0.617	0.953	1.385	1.671
	Convergence condition	-	1.1709	0.3843	0.0137
F2	Optimized mean	8.6098	8.6059	8.6046	8.6045
	Average operating time (s)	0.543	0.969	1.421	1.606
	Convergence condition	-	0.0453	0.0151	0.0012
F3	Optimized mean	0.002541	0.002436	0.002412	0.002408
	Average operating time (s)	0.586	0.849	1.287	1.599
	Convergence condition	-	4.1322	0.9852	0.1658
F4	Optimized mean	1.34003E-05	1.29E-05	1.28E-05	1.26E-05
	Average operating time (s)	0.614	1.265	1.68	1.55
	Convergence condition	-	3.5632	2.1095	0.8419
F5	Optimized mean	0.0733	0.0409	0.03784	0.0301
	Average operating time (s)	0.726	1.1	1.233	1.807
	Convergence condition	-	44.2019	7.4817	20.4545
F6	Optimized mean	8.8113	6.5458	5.44	4.728
	Average operating time (s)	0.76	0.985	1.233	1.586
	Convergence condition	-	25.7113	16.8933	13.0882
F7	Optimized mean	15.6435	15.1187	14.2415	14.1232
	Average operating time (s)	0.579	0.863	1.321	1.592
	Convergence condition	-	3.3547	5.8021	0.8307
F8	Optimized mean	2.5632	2.1998	1.6826	1.4576
	Average operating time (s)	0.564	0.886	1.253	1.605
	Convergence condition	-	14.1776	23.5112	13.3722
F9	Optimized mean	0.2044	0.2028	0.1475	0.1382
	Average operating time (s)	0.603	1.061	1.302	1.793
	Convergence condition	-	0.7828	27.2682	6.3051
F10	Optimized mean	1.008	1.004	1.002	1
	Average operating time (s)	0.652	1.705	1.72	2.95
	Convergence condition		0.3968	0.1992	0.1996

normalized mean square error (NMSE) are used to evaluate the image restoration performance.

Experiment 1 (establishing a mapping relation of image restoration). In order to establish the mapping relationship between original image and blurred image, we add Gaussian blur (5×5, $\delta = 0.8$) and Gaussian random noise (variance=0.01) to Barbara image. Blurred image is generated by convolution of original image and Gaussian filter. Barbara's image restoration model is constructed from degraded and original Barbara images. After 500 iterations of training, the optimization results for parameters C and δ are 5.2 and 0.6231. In order to verify the image restoration effect of the algorithm in this paper, the support vector regression [15] and BP neural network algorithm [10] are selected to compare, and the final image restoration effect is shown in Figure 4 and Table 4.

From the final training results, we can know that the mapping approach can heighten the image quality of blurred image. The PSNR of BP neural network algorithm is low, while the NMSE is higher, which is because its initial parameters need to be selected and corrected. SVR method is better than BP neural network in image restoration, but it takes more time in cross-validation of parameters. However, this method takes less time and has higher PSNR and lower NMSE, which shows better image restoration performance.

Experiment 2 (comparison of degraded image restoration using the trained mapping). The TFOA-LSSVR mapping model proposed in this paper is applied to different blurred images. In order to make the algorithm convincing, there are three blurred ways, namely, motion blur ($L = 20, \delta = 30$), disk blur ($R = 10$), and passivation blur. They are,

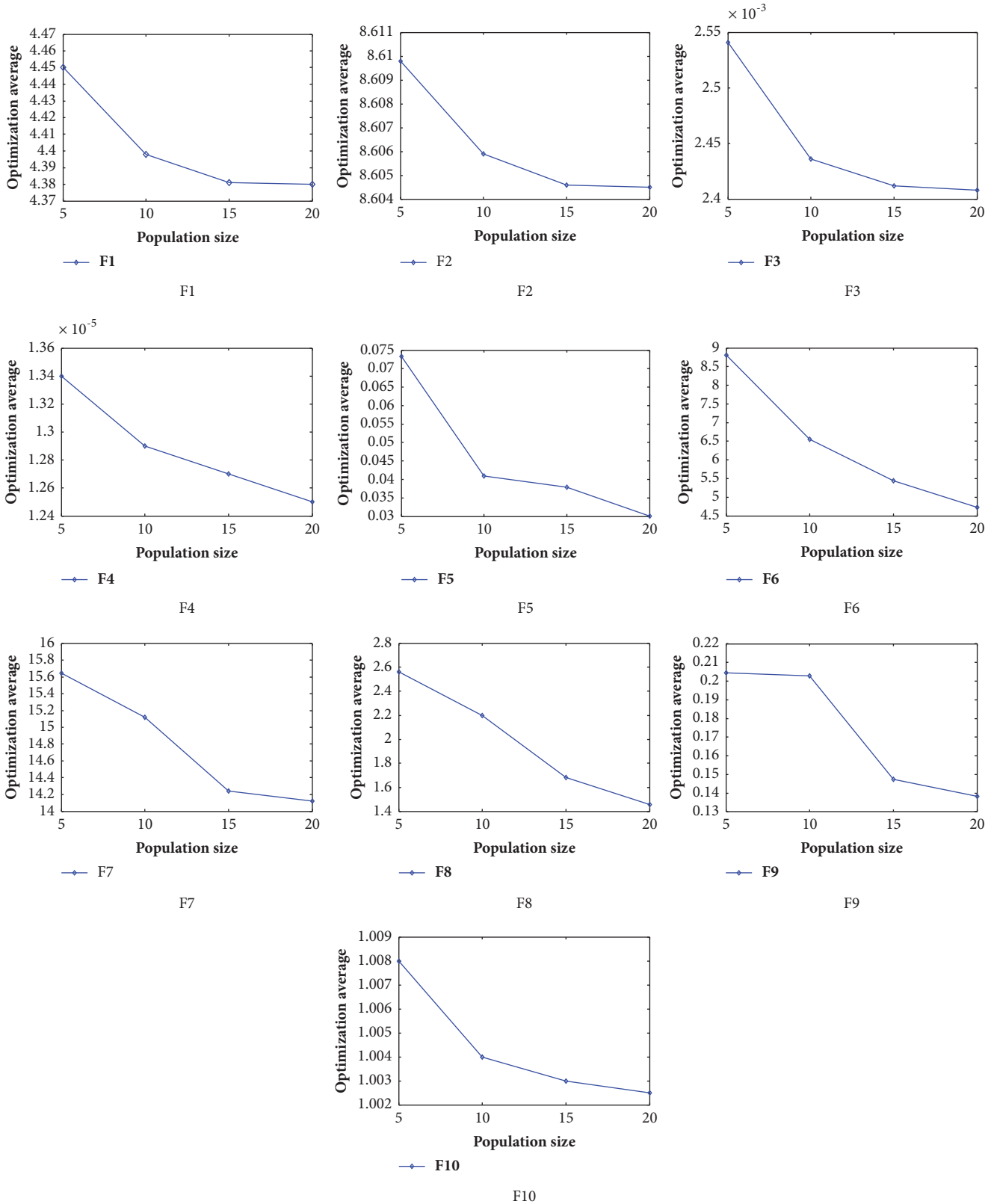


Figure 2: The relationship between population size and optimized mean of F1-F10.

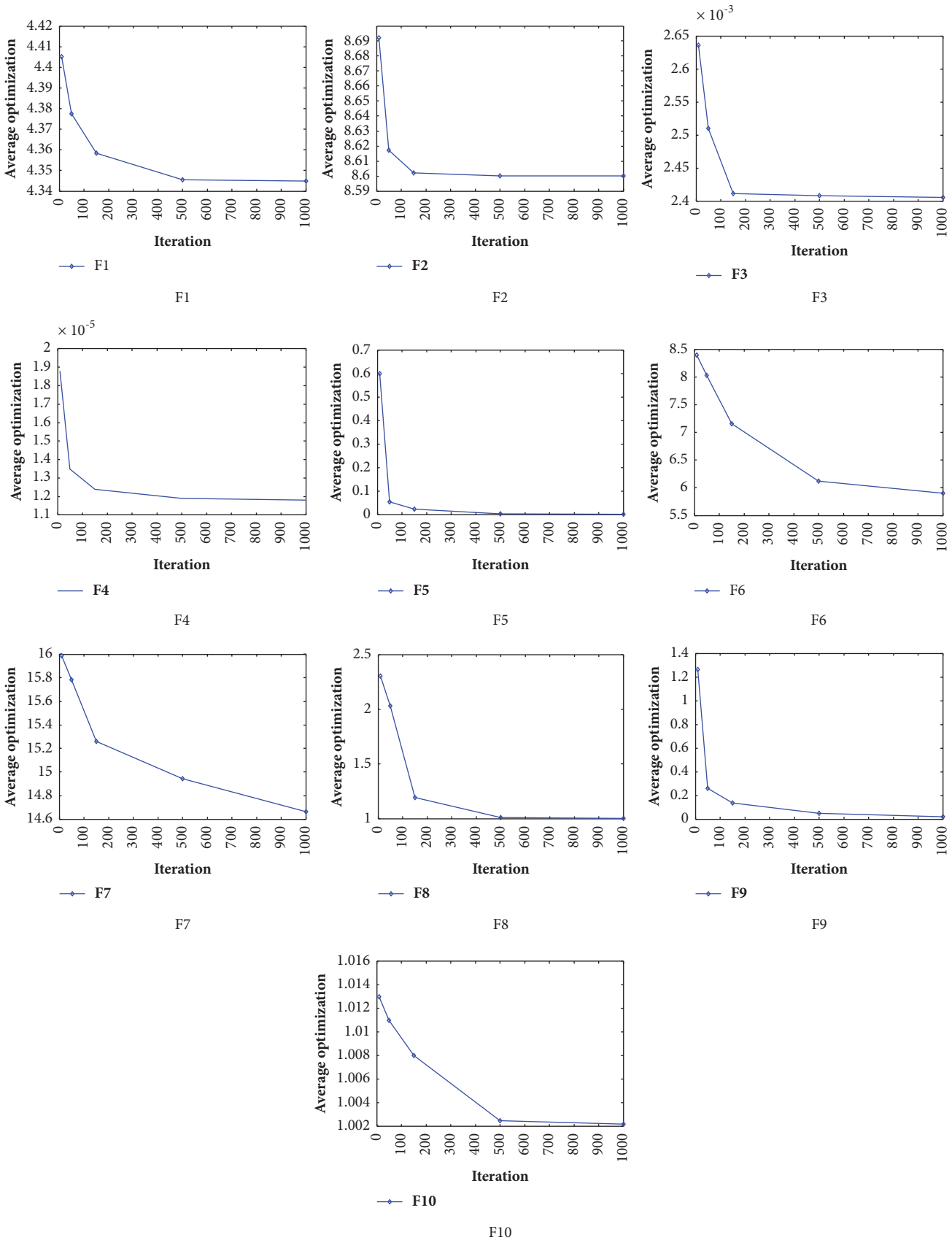


Figure 3: The relationship between max *gen* and optimized mean of F1-F10.

Table 3: Effect of maximum iteration number on algorithm performance.

Function	Evaluation index	max*gen*				
		10	50	150	500	1000
F1	Optimized mean	4.4052	4.3775	4.3584	4.3456	4.3448
	Average operating time (s)	0.311	0.943	1.771	6.14	11.16
F2	Optimized mean	8.6921	8.6172	8.602	8.6002	8.6001
	Average operating time (s)	0.275	0.658	1.751	5.871	10.983
F3	Optimized mean	0.002637	0.002511	0.002412	0.002409	0.002406
	Average operating time (s)	0.283	1.078	1.805	5.399	11.393
F4	Optimized mean	1.88E-05	1.35E-05	1.24E-05	1.20E-05	1.18E-05
	Average operating time (s)	0.246	0.662	1.861	5.345	11.085
F5	Optimized mean	0.5993	0.0533	0.022	0.002	0.000858
	Average operating time (s)	0.366	0.872	1.755	5.382	11.543
F6	Optimized mean	8.3914	8.0295	7.1533	7.12	7.1
	Average operating time (s)	0.351	0.704	1.733	5.417	11.641
F7	Optimized mean	15.9877	15.7825	15.2618	14.9422	14.6644
	Average operating time (s)	0.317	0.741	2.386	5.962	11.726
F8	Optimized mean	2.304	2.0313	1.1926	1.0082	1.002
	Average operating time (s)	0.248	0.684	1.798	5.516	11.231
F9	Optimized mean	1.2648	0.2589	0.1368	0.04961	0.0209
	Average operating time (s)	0.237	0.734	1.74	5.982	11.317
F10	Optimized mean	1.013	1.011	1.008	1.0025	1.0022
	Average operating time (s)	0.314	1.384	2.837	8.16	17.394

Table 4: Experimental results of different approach.

Approach	PSNR	NMSE	Time Cost
Blur image	18.1292	0.1495	-
BP neural network	23.3441	0.0914	13.2354
Support vector regression	26.9224	0.0758	20.1126
TFOA-LLSVR	28.6251	0.0652	9.1128

respectively, applied to three original standard images for blur processing simulation, and Gaussian noise is added to the motion blur images and disk blur images. In order to further explain the image restoration effect, Lucy-Richardson (LR) algorithm [18] is used to restore the different degraded image. The specific results are shown in Figure 5 and Table 5.

In this experiment, TFOA-LSSVR can improve image quality in PSNR and NMSE. But LR algorithm's recovery result is not very satisfactory. For motion blurred images added with Gaussian noise and disk blurred images added with Gaussian noise, the final image clarity of LR algorithm is not improved, and a large number of spots appear. For passivation blurred images, LR results showed some blurring and a large number of spots too. All TFOA-LSSVR algorithms recover images that are clearer than blurred images, and most of the noise is reduced. Therefore, the TFOA-LSSVR model in this paper has better application effect in image restoration.

5. Conclusion

This paper proposes an image restoration method in edge computing environment which achieves good image real-time restoration results. In this paper, TFOA and LSSVR are combined to establish a nonlinear mapping model for image restoration. Firstly, the population size and the maximum iteration number of TFOA are determined by using 10 test functions. Then, the LSSVR error function is optimized as an adaptive function of TFOA. The LSSVR with optimized parameters is used to establish the degradation relationship between the original image and the blurred image. Finally, the image restoration effect is verified. In order to verify the validity of the mapping method, the BP network and the SVR algorithm were compared, and the feasibility of the combination of TFOA and LSSVR was verified. At the same time, the nonlinear mapping model constructed by this algorithm can be used to recover the degenerate

Table 5: Comparison of different restored methods.

Blurs & Noise	Blur image		LR		TFOA-LLSVR	
	PSNR	NMSE	PSNR	NMSE	PSNR	NMSE
Motion Blur	19.2455	0.1489	19.3155	0.1326	24.3776	0.0689
Disk blur	17.9981	0.0986	18.3864	0.0997	22.1244	0.0891
Passivation blur	18.9987	0.0834	21.9166	0.0792	25.7311	0.0667

(a) Original Barbara image (b) Blurred Barbara image (c) Restored by BP neural network

(d) Restored by SVR (e) Restored by TFOA-LLSVR

Figure 4: The final image restoration effect.

images in the training set and have better image restoration than the LR algorithm. The experimental results show that the combination of TFOA and LSSVR can achieve a satisfactory recovery effect. This method has advantages in other areas of competition, especially in terms of CPU time cost.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work is supported by National Key R&D Program of China (2017YFC0820700); the Foundation of Science and Technology on Information Assurance Laboratory (no. KJ-17-101); and the National Science Foundation Project of China (no. 61701188).

(a) Motion blurred images

(b) Disk blurred images

(c) Passivation blurred images

(d) Restored by LR

(e) Restored by the method proposed in this paper

Figure 5: The specific results.

References

[1] M. R. Banham and A. K. Katsaggelos, "Digital image restoration," *IEEE Signal Processing Magazine*, vol. 14, no. 2, pp. 24–41, 1997.

[2] J. G. Triana, L. C. Romero, and G. D. Roldán, "Restauración digital de imágenes mediante ecuaciones diferenciales parciales," *Revista UDCA Actualidad & Divulgación Científica*, vol. 16, no. 2, pp. 511–518, 2013.

[3] A. Kumar, M. F. Hassan, and P. Raveendran, "Learning based restoration of Gaussian blurred images using weighted geometric moments and cascaded digital filters," *Applied Soft Computing*, vol. 63, pp. 124–138, 2018.

[4] G. Xu and E. Tang, "Blind deconvolution for image restoration based on text characteristic," in *Proceedings of the 2011 International Conference on Control, Automation and Systems Engineering (CASE)*, pp. 1–3, IEEE, Singapore, Singapore, July 2011.

[5] J. Yao, S. Tian, X. Wang et al., "Blind restoration method of three-dimensional microscope image based on RL algorithm," in *Proceedings of the ISPDI 2013 - Fifth International Symposium on Photoelectronic Detection and Imaging 2013: Optical Storage and Display Technology*, vol. 8913, p. 89130Z, International Society for Optics and Photonics, Beijing, China, 2013.

[6] J. Y. Zhao, Y. H. Wu, J. L. Jia et al., "Image restoration based on real time wave-front information," *Guangxue Jingmi Gongcheng (Optics and Precision Engineering)*, vol. 20, no. 6, pp. 1350–1356, 2012.

[7] S. S. Mukherjee, R. Chowdhury, and S. Bhattacharyya, "Image restoration using a multilayered quantum backpropagation neural network," in *Proceedings of the 2011 International Conference on Computational Intelligence and Communication Networks, CICN 2011*, pp. 426–430, IEEE, India, October 2011.

[8] P. K. Sethy, L. Panda, and S. K. Behera, "Ann based image restoration in approach of multilayer perceptron," in *Proceedings of the 2016 International Conference on Inventive Computation Technologies, ICICT 2016*, vol. 2, pp. 1–4, IEEE, Coimbatore, India, August 2016.

[9] H. Takeda, S. Farsiu, and P. Milanfar, "Kernel regression for image processing and reconstruction," *IEEE Transactions on Image Processing*, vol. 16, no. 2, pp. 349–366, 2007.

[10] Y. Liu, C. Li, and H. Xu, "A BP neural network model for image restoration based on area around pixel and edge information," in *Proceedings of the 2nd Annual Conference on Electrical and Control Engineering, ICECE 2011*, pp. 3761–3764, IEEE, China, September 2011.

[11] G. Sharma, F. Zhou, J. Liu et al., "Fruit fly optimization based least square support vector regression for blind image restoration," in *Proceedings of the International Symposium on Optoelectronic Technology and Application 2014: Image Processing and Pattern Recognition*, pp. 9301–93011W, International Society for Optics and Photonics, Beijing, China, 2014.

[12] W.-T. Pan, "A new fruit fly optimization algorithm: taking the financial distress model as an example," *Knowledge-Based Systems*, vol. 26, pp. 69–74, 2012.

[13] X. Guo, J. Zhang, W. Li, and Y. Zhang, "A fruit fly optimization algorithm with a traction mechanism and its applications," *International Journal of Distributed Sensor Networks*, vol. 13, no. 11, Article ID 1550147717739831, 2017.

[14] F. Ye, X. Y. Lou, L. F. Sun, and W. Du, "An improved chaotic fruit fly optimization based on a mutation strategy for simultaneous feature selection and parameter optimization for SVM and its applications," *PLoS ONE*, vol. 12, no. 4, p. e0173516, 2017.

[15] D. Li, R. M. Mersereau, and S. Simske, "Blind image deconvolution through support vector regression," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 18, no. 3, pp. 931–935, 2007.

[16] Z.-B. Jiang and Q. Yang, "A discrete fruit fly optimization algorithm for the traveling salesman problem," *PLoS ONE*, vol. 11, no. 11, Article ID e0165804, 2016.

[17] Z. He, Q. Wang, Y. Shen, and Y. Wang, "Discrete multivariate gray model based boundary extension for bi-dimensional empirical mode decomposition," *Signal Processing*, vol. 93, no. 1, pp. 124–138, 2013.

[18] X. Guo, Y. Li, T. Suo, H. Liu, and C. Zhang, "Dynamic deformation image de-blurring and image processing for digital imaging correlation measurement," *Optics and Lasers in Engineering*, vol. 98, pp. 23–30, 2017.

Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 9354273, 12 pages
https://doi.org/10.1155/2018/9354273

Research Article

DS-Harmonizer: A Harmonization Service on Spatiotemporal Data Stream in Edge Computing Environment

Weilong Ding[1,2] **and Zhuofeng Zhao**[1,2]

[1]*Data Engineering Institute, North China University of Technology, 100144 Beijing, China*
[2]*Beijing Key Laboratory on Integration and Analysis of Large-Scale Stream Data, Beijing 100144, China*

Correspondence should be addressed to Weilong Ding; dingweilong@ncut.edu.cn

Received 18 April 2018; Accepted 20 June 2018; Published 5 August 2018

Academic Editor: Shangguang Wang

Abundant sensors in various types are widely used in modern cities to comprehend the current situations in real time. The raw data in open conditions is always in low quality and is hard to employ directly due to its imperfect or missing records. Traditional data preprocessing methods focus on the offline historical data and remain a dilemma between the efficiency and the overhead. In this paper, a data harmonization service *DS-Harmonizer* is proposed on spatiotemporal data stream in the edge computing environment. Through the online cleaning and complementing steps of the hierarchical service instances, the records' validity and continuity can be guaranteed in an efficient way. On the simulated data in a practical project, the service shows high performance, low latency, and acceptable precision in extensive conditions.

1. Introduction

Various sensors of smart cities are adopted in practice nowadays, such as recognition cameras on the trunk roads, smart-card readers in the buses, inductive loops at the toll stations, and transducer in the power plants. Those sensors locating at the edge of the network gather their data hierarchically in a timely manner for further data analysis through either offline batch tasks or online real-time tasks [1]. The diversity and the adequacy of sensory data make it possible to comprehend the current urban situation and the future trends from different perspectives even in real time. In the emerging infrastructure of edge computing environment, leveraging resources closer to the edges would be more efficient for data analysis [2] than that of traditional ways at the data centers in the core network.

The raw data in such open conditions is always in low quality and hard to employ directly for further usage. In a general statistics, 15% of the data gathered directly from sensors is incomplete or contains errors [3]. Due to the data noise, communication error, or device failure, it is common for the processing to encounter critical data distortion or records missing. Take the data in transportation domain as an example. Williams B M et al. in their research [4] show that approximately 20% of the traffic flow data in the investigation were missing. In the data of seven years in Alberta, Canada, that proportion is more than 50% and even 90% at some periods. In the official statistic of Minnesota Department of Transportation, more than 40% of traffic flow data has missing values [3]. The missing data or errors among data are usually caused by three reasons: the data would be lost due to a broken communication link; the data would be abandoned actively by the backend system when the cache is overflow without compensation; the data would be dropped by the dispatching thread in an overloaded parallel or competing environment.

Against the imperfect or missing records, the preprocessing on sensory data is necessary but still remains inherent challenges in edge computing environment especially when the data volume and rate grow. On the one hand, low latency of preprocessing is hard to guarantee on continuous sensory data. Such data stream would be accumulated concurrently and fast from massive sensors at second-level frequency. After the data has been stored, traditional methods face the intrinsic shortage due to the physical limitation of disk

Table 1: The structure of traffic flow data.

Attribute	Notation	Type
Time_start	Start time of an interval	Time
Time_end	End time of an interval	Time
Exit_station	Identity of exit station	Space
Traffic_flow	Amount of exit vehicles	Aggregation

I/O. Although leveraging edge resources can alleviate costs associated with data processing, it tends to be constrained by the capabilities in edge servers. Accordingly, it has to balance the data quality and processing latency with computation overheads. On the other hand, the complementing on the missing values is always blamed on the accuracy. The missing records of the sensory data always lead to unreasonable zero values of business statistics. For example, the travel time being zero at given periods or on given road segments may come from missing records of ALPR (Automatic License Plate Recognition) data. In practice, it is ought to be recognized and repaired, because the aggregation on data stream reflects the continuous situation [5] and such zero or missing is not comprehensible. Moreover, no general standard exists yet due to the diversity of data or business. The spatiotemporal characteristics of sensory data have not paid enough attentions in current works, which must be the very key to improve the practical effects.

In this paper, the typical traffic flow data in highway domain is taken into consideration, and a data harmonization service DS-Harmonizer is proposed in edge computing environment. Through the steps of online cleaning and complementing in the hierarchical service instances, the records' validity and continuity can be guaranteed. Our contributions are listed as follows. (1) On spatiotemporal data stream, the latency of service can hold second-level even when fusing the auxiliary data; (2) through the business constraints and enhanced ARIMA model, the service can distinctly improve the accuracy for the data cleaning and complementing; (3) by the evaluation in a practical project, the benefits of service is convincing in extensive conditions. The rest of this paper is organized as follows: Section 2 shows the motivation and related works; Section 3 elaborates our data harmonization service including two key steps; Section 4 quantitatively demonstrates performance and effects in the experiments; Section 5 summarizes the conclusion.

2. Background

2.1. Motivation. The research in this paper is driven by *Highway Big Data Analysis System* in Henan, one of the provinces in China. The goal of this system is to improve the routine business analysis through Big Data technologies for public travelling and official management. Operated by *Henan Transport Department*, the system has been online since October 2017. On one billion records of the years 2016 and 2017, the analyses in the system involve 660 million vehicles, 37 highway lines, and 279 toll stations. In the system, the traffic flow defined as follows is one of the most significant business metrics.

Definition 1 (traffic flow of toll station(s)). On toll stations L_s, traffic flow is the volume of vehicles exiting L_s at a time period between the start time T_s and the end time T_e, where $|L_s| \geqslant 1$ and $T_e > T_s$. It is counted periodically with a frequency of interval length $\sigma = |T_e - T_s|$. In practice, σ is always 5 minutes, 15 minutes, or 30 minutes as short-time period.

In the system, a record of traffic flow data is typical spatiotemporal and contains 4 attributes as in Table 1. Here, the two temporal attributes present either the start or end of a time period; the spatial attribute is the identifier of a toll station where vehicles exit and pay the toll; the aggregative attribute is the traffic flow value at a given toll station and a given time period.

Such traffic flow data is generated continuously by inductive loops embedded in the ground at a toll station and would be transferred to hierarchical data centers regularly. The procedure of data transmission is illustrated as in Figure 1. The sensory data is first transferred from devices to a road side server through direct wires, and then to servers in section center, region center, and province center in batches step by step through private network. Typical tree topology is composed of those servers: a server connects only an upper one, while it could gather the data from various lower ones. Here, the road side servers and the section centers servers can be regarded as the edge servers in the hierarchy due to their limited computation capacity.

During the procedure above, the error or missing records exist in the records of raw data, which would be disseminated to the subsequent servers. On the one hand, the temporal attributes in the records sometimes are inconsistent. For example, both 2001-01-01 and 2015-05-31 appear in some continual records from the same device. It is always caused by the devices in troubles whose data is considered as untrustworthy. It is not easy to discriminate such false data if no business constraints or conditions are employed. On the other hand, the records of traffic flow are missing at given periods of some toll stations. Due to the high traffic density, the communication may be broken and the device may be in a malfunction, and the sensory data fails to transfer to the servers. In practice, such missing data has to be substituted by the reasonable values for reasonable statistics. In fact, the data preprocessing is difficult due to the scalability and time-sensitiveness guarantee especially when the data size or rate grows. Current methods to tackle imperfect records

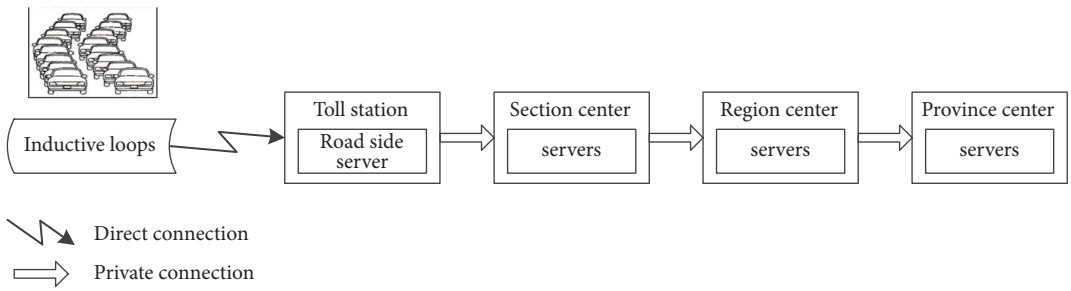


Figure 1: Traffic flow data transmission in a hierarchy.

are centralized and completed in data centers at the core network, which would consume heavy IO on GigaByte data and endures hour-level latency.

Therefore, it is required to handle the continuous spatiotemporal data before further usage in a low latency and cost-efficient way. That is just our original motivation.

2.2. Related Work. Data harmonization combines data from different sources with a comparable view and implies various intentions in different studies [6]. In this paper, this term is extended to the necessary preprocessing services working before the core business calculations. It includes two key steps: the data cleaning for imperfect records and data complementing for the missing ones. Accordingly, the related works can be classified in such two perspectives.

The first perspective is data cleaning. Data cleaning is the process to identify unreasonable data and fix possible errors [7] to improve data quality and keep the semantic consistency [8]. It always involves steps about identifying errors and repairing data. First, errors ought to be identified as the inconsistency, duplication, invalidness, and nonintegrity in the data. It is also the procedure to capture the violations based on business constraints [9] through the technologies like similarity join, clustering, and the rule-based fixing [10]. However, the spatiotemporal characteristics are not exploited enough especially for the continuous data stream [11]. An analogous work [12] concerns the problem about real-time data in wireless sensor network, but it only focuses on the redundant duplication. Second, data repairing aims to revise the records' errors after they have been identified. The heuristic repairing algorithms like functional dependency [13] or denial constraints [14] often employ confidence values [15] to alter possible errors or missing attributes. Commodity cleaning systems like NADEEF [16] even require consulting business professionals for more domain knowledge. However, in highway domain, the general principle for data cleaning still lacks. In this paper, on the continuous data stream, our method concerns temporal inconsistency and business constrains' violation according to typical spatiotemporal characteristics. The efficiency and scalability are guaranteed in edge computing environment.

The second perspective is data complementing. To improve the low quality from the missing records, the data complementing is the manipulation techniques to substitute the missing one [17] with the approximate value. Against the null or distorted value from the missing records, data complementing always refers to the domain experiences. First, an intuitive way is to substitute missing values through domain specific threshold. Such a method of highway domain is proposed using the upper and lower bound on traffic flow data [18]: the distorted value larger than the upper bound (or smaller than the lower bound) would be revised as that bound. It overemphasizes the value's validity but neglects the real rationality on spatial or temporal factors. Second, another typical idea is to generate values from the historical trends by offline processing. Such a method [19] is proposed for traffic flow data complementing from the values of the same period in the previous day and that of the previous period in the same day. The complementing value $y(t) = a * y(t\text{-}1) + (1\text{-}a) * y^{(k-1)}(t)$, where a is a predefined weight for the current day, $y(t\text{-}1)$ is the traffic flow value of the previous period in the same day, and $y^{(k-1)}(t)$ is the value of the same period in the previous day. It can reduce the influence of the flow fluctuation, but the offline manner could not fit the continuous condition. Third, the value can be generated from the neighbors of the missing one by online or offline processing. Such a method [19] is used for traffic flow data complementing through the values of the adjacent periods in the same day. The complementing value $y(t) = [y(t\text{-}n) + y(t\text{-}n{+}1) + \ldots + y(t\text{-}1)]/n$, where n is the neighbor volume and $y(t\text{-}n)$ is the traffic flow of the nth previous period. That makes it possible to work on real-time data stream, but such arithmetic average cannot always achieve accurate value due to the outlier values. Fourth, the data complementing can be regarded as a short-term prediction problem on the recent data. The prediction idea is widely used in fields like QoS evaluation, process management, and service recommendation [20]. In highway domain, the traffic flow prediction is one of the hottest topics [21] and current work can be classified in different perspectives like predictive period, predictive range, and implemented technology. Such methods for data complementing can acquire more precise results than that of others, but they cost more due to the calculation complexity. Current technical mainstream is the Big Data [22] solutions on popular system like Hadoop, Storm, or Spark running at core data centers [2]; while in edge computing environment only limited computation capacities of edge servers are available. Accordingly, those methods cannot be applied directly for the data complementing in this paper. As the recent trends exploiting spatiotemporal characteristics, our method adopts enhanced prediction technology for the better accuracy with economy consumption.

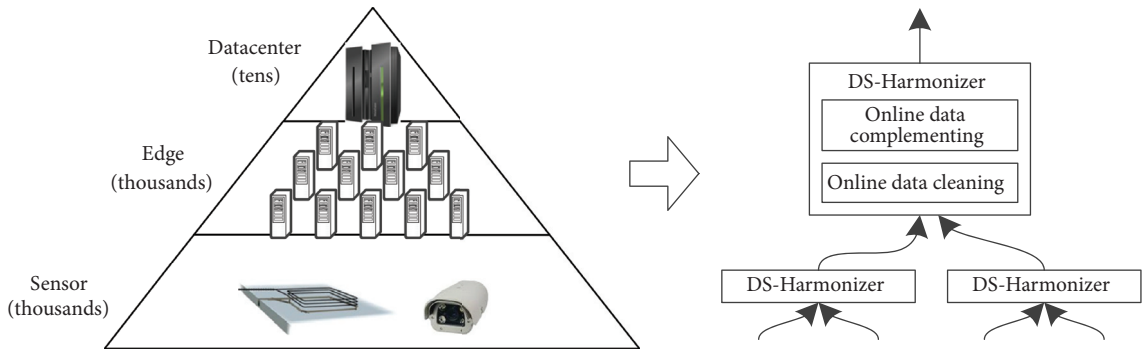


Figure 2: DS-Harmonizer in hierarchical edge environment.

In brief, on continuous data with spatiotemporal characteristics, current works still lack effective approaches to leverage edge resources for either data cleaning or data complementing. Taking the traffic flow data in highway domain as an example, we introduce our harmonization service on data stream against the imperfect or missing records.

3. Data Harmonization on Spatiotemporal Data Stream

3.1. Methodology. The DS-Harmonizer (Data Stream Harmonizer service) is a lightweight streaming processing service running in hierarchical edge servers as Figure 2.

In highway domain like the scenario of Section 2.1, the edge environment can be abstracted as three layers. (1) The bottom sensor layer maintains the thousands of sensors in multiple types. The raw sensory data generates from sensors and transfers to the edge layer. For example, at a toll station, the inductive loops embedded in the ground and the recognition cameras installed on the lane gantry would work collaboratively to yield traffic flow data. As the description in Table 1, the traffic flow data includes the time, station identifier, and an aggregative value. (2) The middle edge layer owns thousands of hierarchical edge servers. When gathering data from sensors, the layer would handle imperfect or missing records and then transfer the validated data to the data center layer. As mentioned before, those edge servers make up the typical tree hierarchy: one server could gather data from multiple lower servers and transfer its results to a certain upper one. Each edge server provides execution environment for a DS-Harmonizer instance with limited computation capacity like CPU, memory, storage, and bandwidth. (3) The top data center layer is a private Cloud for massive business calculations on sensory data. Compared with the servers in edge layer, the data center owns enough resources for Big Data analysis, such as *daily traffic flow of ETC vehicles*, *weekly mileage of MTC vehicles*, and *monthly proportion of vehicle types*.

The sensory data floats from bottom to the top, and it would be hierarchically handled and transferred in the edge layer through the instances of DS-Harmonizer service. The service is a lightweight streaming processing job including two consecutive steps: the data cleaning step against the imperfect records and the data complementing step for the missing ones. Each service instance runs independently in an edge server and collaborates with its direct neighbors through the message communication. Moreover, such service collaboration provides somewhat guarantee of data availability when an instance or an edge server crashes: the lost data in that case can be regarded as the missing records and would be complemented by DS-Harmonizer in an upper edge server. The two steps for data harmonization in the service would be elaborated in the following parts.

3.2. Online Data Cleaning. When the data is gathered from sensors or lower servers, DS-Harmonizer in the target edge server would carry out the online cleaning first to handle the imperfect records in the data. We take the scenario in Section 2.1 as an example and illustrate the procedure by Figure 3. The input here is the continuous sensory data in two different types, and the output is the cleaned data stream of traffic flow. The goal of this step is to revise some errors in attributes and calibrate the semantic inconsistency among data.

As in the left part of Figure 3, it is a typical streaming processing procedure. Each record of traffic flow data in the stream is read at a time and then its attributes are extracted for verification. If the station does not exist or neither of temporal attributes is on the current day, the record is regarded as meaningless and would be abandoned directly, because no hints are available for the repair. Otherwise, some temporal errors can be corrected: if the start time is less than the end one, these twos would be replaced by each other; the false date in either temporal attribute can be substituted by the current day. More business constraints can be adopted here. After an optional calibration procedure with the data in other types, a revised valid record would be emitted to the downstream.

The right part of Figure 3 demonstrates how the data in different types would fuse for the data cleaning. At a toll station in highway, besides the inductive loops, the recognition cameras are deployed, whose data could be used for the traffic flow calibration. The ALPR (Automatic License Plate Recognition) data generated by those cameras has the structure in Table 2, which includes several temporal, spatial, and entitative attributes. Compared with the traffic flow data, the ALPR data contains more information about vehicles

Table 2: The structure of ALPR (automatic license plate recognition) data.

Attribute	Notation	Type
Time	Timestamp when vehicle is passing	Time
Exit_station	Identity of exit station	Space
Exit_lane	Lane number of exit station	Space
Vehicle_license	Vehicle identity	Entity
Vehicle_type	Vehicle type	Entity
Card_id	vehicle passing card identity	Entity
ETC_id	vehicle ETC card identity	Entity
ETC_cpu_id	ETC card chip identity	Entity

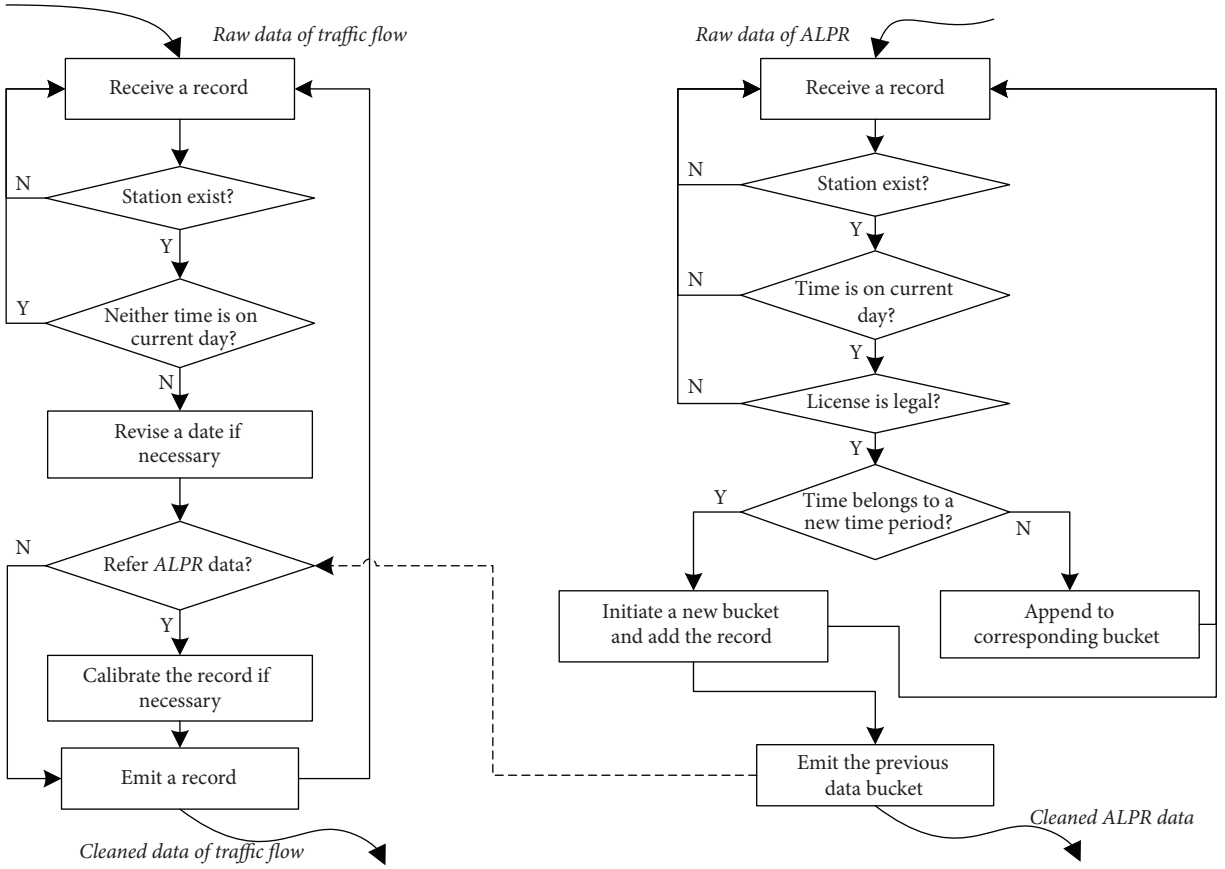


Figure 3: Online data cleaning procedure.

and roads, so that it can be employed for wider analysis in more perspectives. As the streaming processing procedure, each record of ALPR data in the stream is read at a time. Besides the verification for the spatial and temporal attributes like that of traffic flow data, the vehicle licenses would be inspected for its legality. For some occasions, a vehicle cannot be recognized correctly in its plate, type, or payment mode. As the key of ALPR data, the license is a vehicle's identifier and its illegality would make a record invalid. The licenses are examined by the filtering on regular expressions. For example, the expression in Figure 4 is used to examine four aspects: plate color, belonging provincial region, license number, and license type. The business constrains on those factors are beyond this paper and would not be elaborated here. After that, each cleaned record would be cached in a bucket structure according to the time attribute. Besides being transferred continuously to the next DS-Harmonizer instance, the result as bucket can be referred by traffic flow data for the calibration: the value of traffic flow during the same period has to be consistent with the count of valid ALPR records.

Therefore, DS-Harmonizer can fuse the continuous sensory data in different types to complete the online data cleaning. The cleaning procedure for any type of data above is an independent streaming processing job and works collaboratively with the others by the message communication in the same edge server. Here, the output is the continuous data stream of the traffic flow data and ALPR data: the former

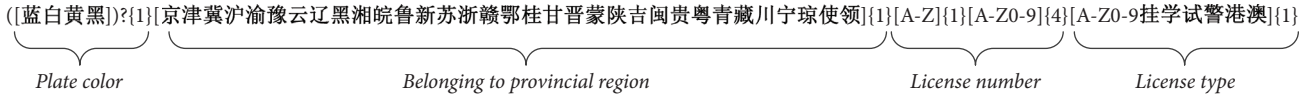


Figure 4: Regular expression for a valid vehicle license.

would be the input of the data complementing of the same service instance; the latter would be the input of other service instance in an upper edge server. With the help of data cleaning, the data quality is improved in a certain extend because the imperfect records cannot disseminate further.

3.3. Online Data Complementing. The output data stream of the data cleaning triggers the online data complementing, which becomes more feasible when some irreparable records are abandoned. Although the traffic flow seems to be fluctuating randomly, it appears to be spatiotemporal correlation [19] because an ongoing vehicle on a specific line has certain speed limitation and imperative distances between others. That makes the traffic flow predictable in short term. Moreover, such prediction to complement the missing data in edge environment has to be low latency in cost-efficient manner, because only limited computation capacity is available in edge servers to balance the effect and cost. Accordingly, we extend the ARIMA (AutoRegressive Integrated Moving Average) model on spatiotemporal data stream to achieve preferable accuracy with comprehensible parameters. The procedure is illustrated by Figure 5.

As the left part of Figure 5, the input is the cleaned traffic flow from data cleaning, and the output is the complemented data stream. It is illustrated as follows, where the "missing" refers to a lost record of traffic flow data at station s of time interval l.

(i) A record at s of time interval l+1 in the traffic flow stream is read.

(ii) The records at the same station are handled by the same hierarchical DS-Harmonizer instances, and their temporal sequence are guaranteed. A record at s of the previous interval l is regarded as a missing one, if it has not been read from the input stream yet. Otherwise, go to (iv).

(iii) The missing record at s of l is substituted by a cached one, which is predicted during the previous time interval l-1. After the record complementing and result emitting, the very record of l+1 would be cached further.

(iv) The cache updates: the record of l+1 would append to cache when the record of l is missing or replaces the one in the cache otherwise.

(v) With this record and the ones at s of recent K intervals in the cache, the predictive value at s of l+2 would be calculated through our enhanced ARIMA model. It would be elaborated in the right part of Figure 5.

(vi) After the predictive record of l+2 is cached, the record of l+1 would be emitted to the output stream.

The cache is maintained in the memory of edge server where the service instance is resident. ARIMA model is employed in the (v) step to predict the traffic flow of the next interval as a potential complement. As a classical short-term prediction model on offline time-series data, ARIMA (p, d, q) process can be expressed as $(1 - \sum_{i=1}^{p} \phi_i L^i)(1 - L)^d X_t = (1 + \sum_{i=1}^{q} \theta_i L^i)\varepsilon_t$, where L is the lag operator, X_t is the traffic flow value at time t, and ε_t is the white noise at time t. It can be regarded as three steps with a respective algorithmic parameter as autoregressive (abbr. AR(p)), differencing (abbr. I(d)), and moving average (abbr. MA(q)). We extend it on spatiotemporal data stream like the right part of Figure 5, where the record is just read and the recent K ones are used as the input.

(a) When the record is just read and the previous K ones at s are ready, the stationary of time-series would be verified. To guarantee low latency of the verification, the ADF (Augmented Dickey-Fuller test) [23] is employed: if the hypothesis is rejected by test, those values are proved stationary and step(c) is triggered with the parameter d=0.

(b) Otherwise, on those nonstationary traffic flow values, differencing operation I(d) would be evaluated iteratively d times until ADF test rejects the hypothesis (i.e., their differential values are stationary). Considering the latency for data stream processing and the fast convergence for the hypothesis testing, the differencing parameter d is restricted not larger than 5 according to the domain experience. In fact, it is always not greater than 3 in practice. Therefore, the d value is found by I(d) and ADF test here.

(c) On the stationary differential values of traffic flow, the AR(p) and MA(q) steps are evaluated. Traditionally, it is a long-term iterative calculation through Least Squares method, while a trick is adopted here for the online processing. Parameter q is not more than 3 according to the domain experience, and parameter $p <= K$+1 (K is a build-time parameter above) implies the sample size. Due to the finite combination of p and q, the minimum of AIC (Akaike Information Criterion) [24] can be found by the enumeration instead of the iteration with the time complexity O($p*q$). For example, it is sufficient for short-term prediction when K=200, because the recent data lies in more than 16 hours (200∗5=1000 minutes) even if the minimal interval length of 5 minutes is used. Therefore, the p and q values are found by the minimal AIC metric.

(d) Model is learned to predict the complementing value. Through the certain algorithmic parameters p, d, and

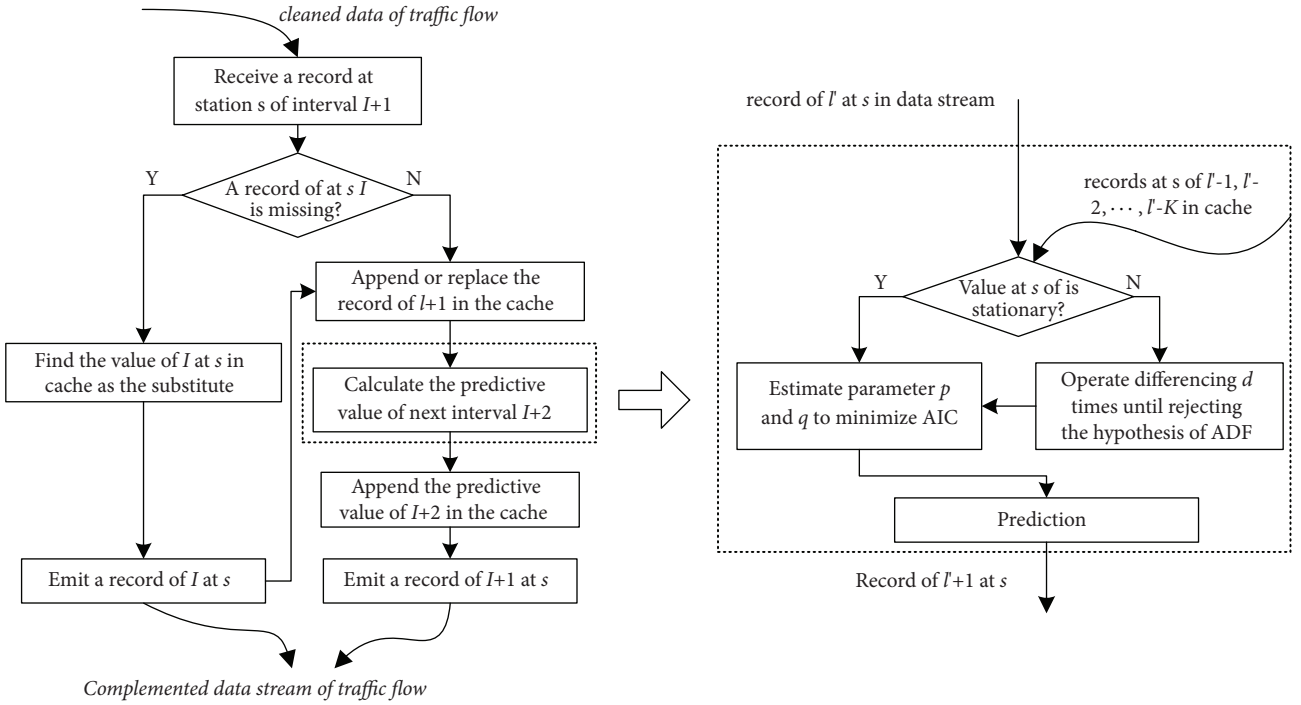


Figure 5: Online data complementing procedure.

q, the ARIMA model can be determined by learning the optimized parameters of model as the traditional way. The traffic flow value at the same station of next time interval can be predicted by that model.

Accordingly, DS-Harmonizer realizes the online data complementing through continuous prediction on data stream. The trade-off of the low latency stream processing is considered in the limited computation capacity of edge servers. Common domain experience is adopted to speed the parameter tuning to avoid the heavy iterations. The output is the traffic flow of current period as data stream and would transfer to the service instance in an upper edge server. Their predictive value of next period is cached for potential data complementing in the future. It makes a cost-efficient solution.

4. Evaluation

4.1. Experiment Setting. In the project mentioned in Section 2.1, our service is evaluated by extensive experiments. For the data center layer, tens of Acer AR580 F2 rack servers via Citrix XenServer 6.2 are utilized to build a private Cloud, each of which owns 8 processors (Intel Xeon E5-4607 2.20GHz), 64 GB RAM, and 80 TB storage. For the edge layer, tens of personal computers are used as edge servers, each of which owns 2-core CPU, 4 GB RAM, and 500 GB storage installing CentOS 6.6 x86_64 operating system. For the sensor layer, the data stream is simulated through our dedicated data generator [25, 26]. The data imported were generated in Henan province since Feb 1st 2017 to Apr 30th 2017. The concurrency and velocity of the simulated stream could be configured by defined scripts and settings in the data generator. By default, simulated traffic flow data is generated from virtual inductive loops with the rate of 1 record per 5 minutes in a loop; each toll station contains 10 loops for 200 toll stations (i.e., concurrency is $10*200=2000$); simulated ALPR data is generated from recognition cameras with the rate of 1 record per second in a camera; each toll station contains 10 cameras for 200 toll stations (i.e., concurrency is identical to loops); the parameter K (sample size of each station) is set as 200. That is, the number of the inductive loops is identical to that of cameras and equals that of bottom DS-Harmonizer instances in a hierarchy.

In any edge server, several tools are deployed. (1) A tailored *Apache Storm* 0.9.4 is installed as a single-machine cluster. All daemons (e.g., *nimbus* and *supervisor*) run in one machine, message acknowledgement is disabled, and some components are removed to reserve more resource for data harmonization. (2) The *ZeroMQ* 2.1.4 is employed as a lightweight message service in front of Storm. (3) DS-Harmonizer is implemented as a *topology* as Figure 6, in which the data cleaning and complementing is realized as a respective *bolt*. Two types of sensory data are brought into the topology by a respective *spout*: ALPR_Spout is for ALRP data and TF_Spout is for traffic flow data. The data transmission between spout and bolt is *shuffle grouping*, which dispatches the data uniformly to each task of Cleaning_Blot; the data transmission between bolts is *field grouping* by the station identifier, which ensures the records of the same station would be dispatched to the same task of Complementing_Bolt. Moreover, the *worker* number is 1,

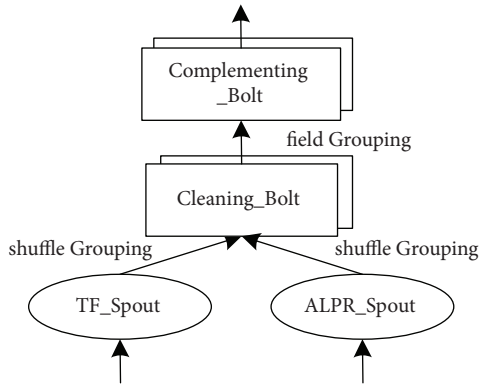


Figure 6: A Storm topology of a DS-Harmonizer instance.

the parallelism (*executor* number) of any bolt is 2, and the parallelism of any spout is 1, all of which can be tuned in Storm. The service instance is the worker of that topology running in an edge server, and the message communication among instances is just like hierarchy of Figure 2.

4.2. Service Performance. First, the performance of DS-Harmonizer is the focus, and its interrelation among the factors of time (i.e., interval length), space (i.e., sensors volume), and edge condition (i.e., service hierarchy) is evaluated. The first two can be configured by the velocity and concurrencies in the data generator and the last one is tuned by service instances' architecture.

Experiment 1. DS-Harmonizer is applied to harmonize the traffic flow data (abbr. TF) in two ways with (*fusion* way) or without (*solo* way) auxiliary ALPR data. All the service instances are in the same bottom level, and the factor concerned here is the number of inductive loops (i.e., hierarchical width). In both ways, the interval length of TF is, respectively, set as 5, 15, and 30 minutes, by configuring TF rate. In the fusion way, the rate of ALPR is kept as 1 record/second per camera. In each test around, the concurrency of TF and ALPR data is from 1000 to 5000 (i.e., 5~25 sensors in each of the 200 stations); the average latency of a record from a certain loop to the data center is counted after the services having run smoothly. The result is shown in Figure 7(a).

Experiment 2. Like two ways above, the factor concerned here is the edge level (i.e., hierarchical depth), and the levels of service instances are deployed from 1 to 5. The concurrency of either TF or ALPR data is kept as 1000 (i.e., 5 sensors in each of the 200 stations). In both ways, the interval length of TF is, respectively, set as 5, 15, and 30 minutes, by configuring TF rate. In the fusion way, the rate of ALPR is kept as 1 record/second per camera. In each test around, a new level is deployed between the top level and the data center by importing 5 service instances (i.e., each instance connects a lower one); the average latency of a record from a certain loop to the data center is counted after the services having run smoothly. The result is shown in Figure 7(b).

We found these consequences during the service execution. (1) DS-Harmonizer has well horizontal scalability in concurrency, and the latency is related to the interval length. As Figure 7(a), the latency stably holds second-level in both ways under different interval lengths. In the solo way when only TF data is harmonized, the latency reflects the processing capacity regardless of the concurrency of TF data. As a result, three interval lengths make the latency no different in the solo way. While in the fusion way collaborated with the ALPR data, the longer the interval length is, the higher the latency would be in any hierarchical width. The velocity of ALPR data is kept as a constant, longer interval length implies larger ones to handle during the data cleaning step, which undoubtedly requires much time. (2) The latency is related to the hierarchical depth of edges. As Figure 7(b), in both ways under different interval length, the latency increases progressively when more levels are introduced. More levels bring longer path of data transmission, which inevitably delays the data from the sensor to the data center. In the solo way, the latency of three interval lengths is still identical and grows slightly in almost the same extent. While in the fusion way, the latency of three intervals rises obviously, and the longer the interval length brings higher latency in any hierarchical depth. The reason also comes from the ALPR data volume discussed before. (3) The latency is also related to the data fusion. In either Figure 7(a) or 7(b), the fusion way under the same interval consumes longer time than that of solo way. It is an intuitive that more resources are required to handle the data in another type. The fusion way seems heavier than the solo one, but it is worthy due to its higher accuracy as discussed later.

4.3. Harmonization Effect. Next, we introduce artificial dirt into the records and evaluate the harmonization effect during the online data cleaning.

Experiment 3. From the real data of a certain day, some records are selected randomly. The proportion of the selected ones in that day is 5%, 10%, 15%, 20%, or 25%, respectively. For such selected records, the artificial dirt in temporal attributes would be introduced as follows: in a half of the selected records, a decimal digit of a timestamp (*time_start* or *time_end*) is altered randomly; in the other half, a decimal number of both timestamps would be altered randomly. Analogously, the artificial dirt in spatial attribute is introduced as follows: in the selected records, the station identifier (*exit_station*) would be substituted with an inexistent one. Then, the stream is relayed through the data generator on the data in that day, where the concurrency of TF data is kept as 1000 (i.e., 5 loops in each of the 200 stations) and the interval length of TF is set, respectively, as 5 minutes by configuring TF rate. DS-Harmonizer is applied as the solo way, and 5 service instances are in the same bottom level. In each test round, after the services having run smoothly, the proportion of the correct records in the output of online cleaning is counted after the comparison with the raw data before the artificial dirt is introduced. The result is showed as in Figure 8.

The feasibility of online data cleaning is proved when the dirty records are introduced. The proportion of correct

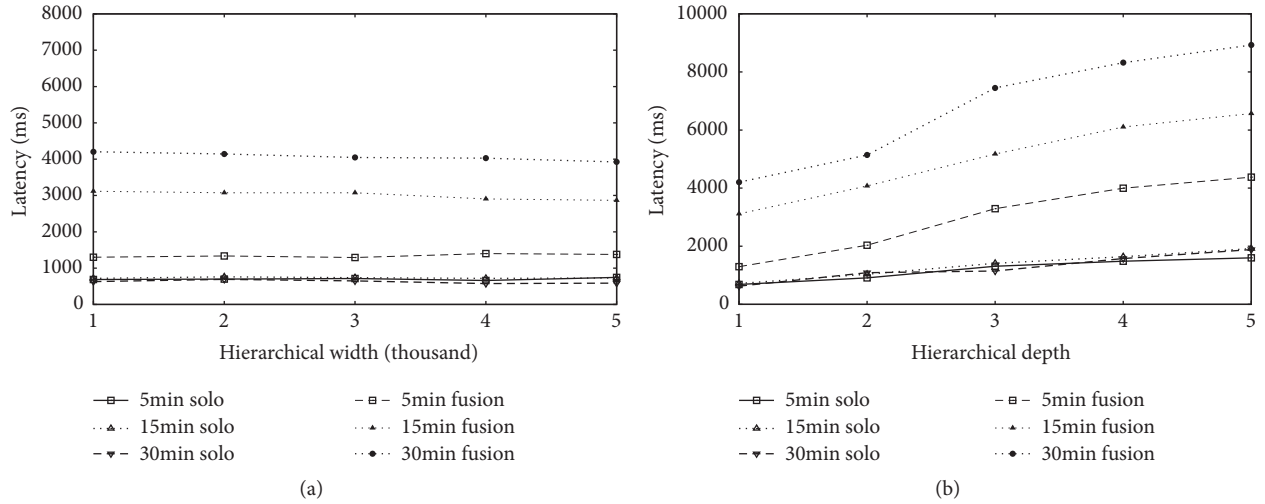


Figure 7: The latency of DS-Harmonizer.

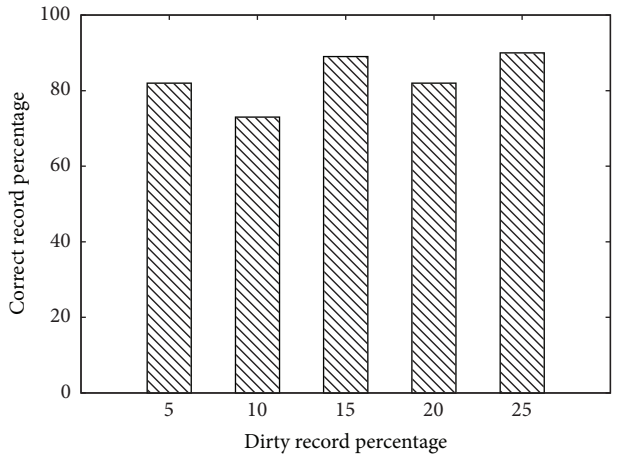


Figure 8: Data cleaning against artificial dirty records.

records in output is more than 70% and remains high even when dirty record scales. In intuition, the result ought to decline when more such altered records have to be handled; while in our online method, it is not related to the volume of the dirty records because any record would be examined once. Although some of dirty records are not distinguished here due to the limited business constraints (more can be introduced then), the online data cleaning still shows the advantage about accuracy.

Then, we analyze the effect of the online data complementing on the data of two different days, because the traffic flow of highway shows distinct trends on workday or holiday. The short-term traffic flow prediction is adopted in DS-Harmonizer to complement missing records, so the predictive error could be the indicator for the evaluation. In this section, three common metrics are used to evaluate errors for prediction. The first is the absolute percentage error (abbr. APE) defined as (1); the second is the mean absolute percentage error (abbr. MAPE) by the definition of (2); the third is the median absolute percentage error (abbr. MDAPE)

Table 3: The evaluation for the prediction on a workday.

Item	ARIMA	ARIMA+
MAPE (%)	6.87	5.79
MDAPE (%)	4.57	3.64
Latency(ms)	3725	1940

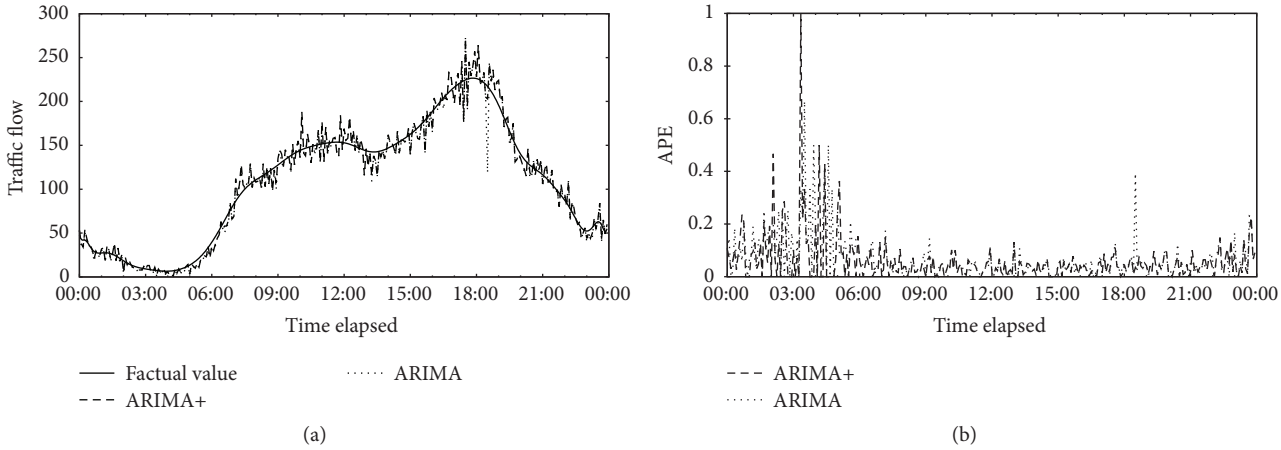
which is the median of APE. Here, at the beginning time of an interval t, x_t is the factual value of traffic flow; x_t' is the predictive value calculated at the beginning time of the previous interval t-1.

$$\mathrm{APE} = \frac{\left|x_t - x_t'\right|}{x_t} * 100\% \tag{1}$$

$$\mathrm{MAPE} = \frac{1}{N}\sum_{1}^{N}\frac{\left|x_t - x_t'\right|}{x_t} * 100\% \tag{2}$$

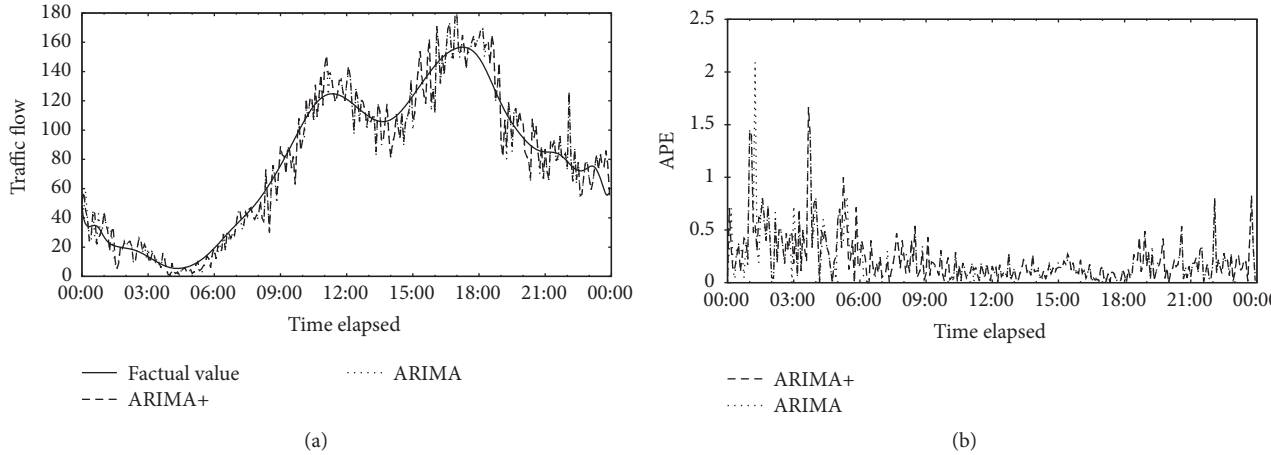
Experiment 4. The data in a workday, Apr 20th 2017, is imported to replay as data stream. In the data generator, a record of traffic flow data is generated every 5 minutes (i.e., interval length is 5 minutes) from one virtual inductive loop. A dedicated instance of DS-Harmonizer is deployed as the solo way in a single bottom level to receive the simulated data. For comparison, both traditional ARIMA (ARIMA) and our enhanced model (ARIMA+) are implemented in DS-Harmonizer. After the service having run smoothly, the output of online complementing and the latencies during each data complementing are noted. After the end of the replay, calculate three error metrics of both models with the real data in that day. The average latency of a record from the loop to the data center is also counted. For a heavy traffic station *ZhengzhouNan*, the result is illustrated as in Figure 9 and Table 3.

Experiment 5. The data on a holiday, Feb 11th 2017, is imported to replay as data stream. The day is the Lantern Festival on the Saturday just after the Chinese New Year.



(a) (b)

Figure 9: Prediction at a certain station on a workday.



(a) (b)

Figure 10: Prediction at a certain station on a holiday.

Other settings are the same as that of the last experiment. After the end of the replay, calculate three error metrics of both models with the real data in that day. The average latency of a record from the loop to the data center is also counted. For the same station *ZhengzhouNan*, the result is illustrated as in Figure 10 and Table 4.

Based on the results of two experiments above, we found that DS-Harmonizer holds the high accuracy with low latency as the discussion below. (1) Both models can approximate the factual trends on either workday or holiday. As in Figure 9(a) and Figure 10(a), two evident spikes appear on either day when the traffic is busy, but on holiday first spike is postponed about 30 minutes and the second one advances 30 minutes. As in Figure 9(b) with Table 3 and Figure 10(b) with Table 4, any of the three metrics in both models is acceptable. (2) Both models own the different precision in workday with that of holiday. Comparing MAPE in Tables 3 and 4, both models fit better on workday, because the stationary of traffic flow on holiday is not outstanding relatively due to more unexpected factors. (3) Our enhanced ARIMA+ model presents the better effects. On the one hand, ARIMA+ shows lower error than the traditional ARIMA, especially in MDAPE. Our model is tailed with the business constrains, which avoids the overfitting in certain extents. On the other hand, our model has advantage in the latency on the data stream. In Tables 3 and 4, ARIMA+ cuts down 35~45% of the average executed time. Compared with the traditional one, our model dramatically reduces the time due to the algorithmic parameters tuning, in which three key parameters d, q, and p have been restricted by their upper bound according to the business constrains.

In brief, DS-Harmonizer proved its high performance and low latency with acceptable accuracy in extensive conditions.

5. Conclusions

In edge computing environment, a data harmonization service *DS-Harmonizer* is proposed to handle imperfect and missing records among the spatiotemporal data stream. By the online data cleaning and data complementing of

Table 4: The evaluation for the prediction on a holiday.

Item	ARIMA	ARIMA+
MAPE (%)	22.52	22.28
MDAPE (%)	6.63	4.00
Latency (ms)	2337	1529

hierarchical services, the records' validity and continuity can be guaranteed in an efficient way. The service shows minute-level latency with horizontal scalability, and it can achieve better precision guarantee during either step in extensive conditions.

Data Availability

The TF (traffic flow) data and ALPR (Automatic License Plate Recognition) data used to support the findings of this study have not been made available because the data were supplied by local management Henan Transport Department under license with certain confidentiality level and so cannot be made freely available. Requests for access to these data should be made to the corresponding author for an application of joint research.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

The authors are grateful to the graduate student Xuefei Wang who helped in improving the quality of this paper. This work was supported by the Youth Program of National Natural Science Foundation of China (nos. 61702014, 61602437), Beijing Natural Science Foundation (no. 4162021), Youth Innovation Foundation of North China University of Technology (no. 1473-1743028), and the Ministry of Transportation, Institute of Highway Science Key Projects (no. 2015-9024).

References

[1] S. Wang, J. Xu, N. Zhang, and Y. Liu, "A Survey on Service Migration in Mobile Edge Computing," *IEEE Access*, vol. 6, pp. 23511–23528, 2018.

[2] E. Renart, D. Balouek-Thomert, X. Hu, J. Gong, and M. Parashar, "Online Decision-Making Using Edge Resources for Content-Driven Stream Processing," in *Proceedings of the 2017 IEEE 13th International Conference on e-Science (e-Science)*, pp. 384–392, October 2017.

[3] M. Zhong, P. Lingras, and S. Sharma, "Estimation of missing traffic counts using factor, genetic, neural, and regression techniques," *Transportation Research Part C: Emerging Technologies*, vol. 12, no. 2, pp. 139–166, 2004.

[4] B. M. Williams, P. K. Durvasula, and D. E. Brown, "Urban freeway traffic flow prediction: application of seasonal autoregressive integrated moving average and exponential smoothing models," *Transportation Research Record*, no. 1644, pp. 132–141, 1998.

[5] D. Sun, G. Zhang, W. Zheng, and K. Li, "Key Technologies for Big Data Stream Computing," in *Big Data: Algorithms, Analytics, and Applications*, CRC Press, Taylor & Francis Group, USA, 2014.

[6] DSDR, "Data Harmonization" https://www.icpsr.umich.edu/icpsrweb/content/DSDR/harmonization.html.

[7] N. Tang, "Big Data Cleaning," in *Proceedings of the Web Technologies and Applications: 16th Asia-Pacific Web Conference, APWeb 2014*, Proceedings., L. Chen, Y. Jia, T. Sellis, and., and G. Liu, Eds., pp. 13–24, Springer International Publishing, Changsha, China, 2014.

[8] M. Chen, S. Mao, and Y. Liu, "Big data: a survey," *Mobile Networks and Applications*, vol. 19, no. 2, pp. 171–209, 2014.

[9] W. Fan, F. Geerts, N. Tang, and W. Yu, "Inferring data currency and consistency for conflict resolution," in *Proceedings of the 2013 29th IEEE International Conference on Data Engineering (ICDE 2013)*, pp. 470–481, Brisbane, Australia, April 2013.

[10] J. Wang and N. Tang, "Towards dependable data repairing with fixing rules," in *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data, SIGMOD 2014*, pp. 457–468, Snowbird, Utah, USA, June 2014.

[11] W. Ding, S. Zhang, and Z. Zhao, "A collaborative calculation on real-time stream in smart cities," *Simulation Modelling Practice and Theory*, vol. 73, pp. 72–82, 2017.

[12] L. Wang, L. D. Xu, Z. Bi, and Y. Xu, "Data cleaning for RFID and WSN integration," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 1, pp. 408–418, 2014.

[13] G. Beskales, I. F. Ilyas, and L. Golab, "Sampling the repairs of functional dependency violations under hard constraints," in *Proceedings of the VLDB Endowment*, vol. 3, pp. 197–207, 2010.

[14] X. Chu, I. F. Ilyas, and P. Papotti, "Holistic data cleaning: Putting violations into context," in *Proceedings of the 29th International Conference on Data Engineering, ICDE 2013*, pp. 458–469, Australia, April 2013.

[15] W. Fan, J. Li, S. Ma, N. Tang, and W. Yu, "Towards certain fixes with editing rules and master data," *The VLDB Journal*, vol. 21, no. 2, pp. 213–238, 2012.

[16] M. Dallachiesat, A. Ebaid, A. Eldawy et al., "NADEEF: A commodity data cleaning system," in *Proceedings of the 2013 ACM SIGMOD Conference on Management of Data, SIGMOD 2013*, pp. 541–552, New York, NY, USA, June 2013.

[17] L. de S. Ribeiro, R. R. Goldschmidt, and M. C. Cavalcanti, *Complementing Data in the ETL Process*, Springer Berlin Heidelberg, Berlin, Germany, 2011.

[18] M. Kim, P. Jinsoo, O. Jaeyoung, C. Hakjin, and K. Yoonkee, "Study on network architecture for traffic information collection systems based on RFID technology," in *Proceedings of the 3rd IEEE Asia-Pacific Services Computing Conference, APSCC 2008*, pp. 63–68, Yilan, Taiwan, December 2008.

[19] T. Wang, *Research of the Short-term Traffic Flow Prediction Based on Spark Platform (in Chinese)*, South China University of Technology, Guangzhou, China, 2016.

[20] S. Wang, Y. Zhao, L. Huang, J. Xu, and C.-H. Hsu, "QoS prediction for service recommendations in mobile edge computing," *Journal of Parallel and Distributed Computing*, 2017.

[21] J. Guo, W. Huang, and B. M. Williams, "Real time traffic flow outlier detection using short-term traffic conditional variance prediction," *Transportation Research Part C: Emerging Technologies*, vol. 50, pp. 160–172, 2015.

[22] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, "Traffic flow prediction with big data: a deep learning approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 2, pp. 865–873, 2015.

[23] Wikipedia, "Augmented Dickey–Fuller test," https://en.wikipedia.org/wiki/Augmented_Dickey%E2%80%93Fuller_test.

[24] Wikipedia, "Akaike information criterion," https://en.wikipedia.org/wiki/Akaike_information_criterion.

[25] W. Ding, Y. Han, J. Wang, and Z. Zhao, "Feature-based high-availability mechanism for quantile tasks in real-time data stream processing," *Software: Practice and Experience*, vol. 44, no. 7, pp. 855–871, 2014.

[26] W. Ding, Z. Zhao, and Y. Han, "A Framework to Improve the Availability of Stream Computing," in *Proceedings of the 2016 23rd IEEE International Conference on Web Services (ICWS 2016)*, pp. 594–601, IEEE, San Francisco, CA, USA, June 2016.

Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 1287969, 13 pages
https://doi.org/10.1155/2018/1287969

Research Article

Message Relaying and Collaboration Motivating for Mobile Crowdsensing Service: An Edge-Assisted Approach

Shu Yang, Jinglin Li, Quan Yuan, Zhihan Liu, and Fangchun Yang

State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing, China

Correspondence should be addressed to Jinglin Li; jlli@bupt.edu.cn

Received 11 April 2018; Accepted 24 June 2018; Published 29 July 2018

Academic Editor: Kok-Seng Wong

Group sensing is a kind of crowdsensing service where HD map producers motivate private cars in a local region to collect data from real world. Group sensing needs vehicles to communicate physically and drivers to collaborate strategically in a mobile or edge-assisted environment. First, we consider collaboration module that motivates drivers to be participants; centralized and distributed motivating methods are discussed. Secondly, we consider communication module; two VANET-based methods are proposed to achieve message relaying in edge infrastructure. To accomplish participants' selection, three combinations of two modules are proposed and simulated based on a flexible framework. The results show that centralized selection could motivate collaboration at a low price but brings heavy communication overhead. Clustered selection requires more incentives and less communication overhead than centralized selection. Distributed selection is usually the first class choice because of its fine performances on both communicating and motivating.

1. Introduction

The rise of big data fuels the research of machine learning and data mining algorithms. Besides algorithms, the method of collecting data is also of great importance [1]. Being intellectualized and networked, smart vehicles are able to sense and communicate in urban area as edge infrastructures [2–4]. By means of crowdsensing, their intrinsic mobility can be leveraged to dynamically collect urban data in different time and areas [5].

A promising service of crowdsensing is collecting environmental data for building HD (High Definition) map [6]. The HD map producers, such as Here, TomTom, and Baidu, need lidar/camera/IMU data to build a live map for autonomous driving [7, 8]. The huge volume of information as well as fast updating requirement to build a "live" map challenges map producers because their own devices undoubtedly could not meet these requirements. Some researchers therefore proposed crowdsensing, in which private cars are incentivized to accomplish a sensing task and upload data to map producer. The incentives could be either real or virtual money.

This paper focuses on how to call up a group of smart vehicles to complete a sensing task in an edge environment, where vehicles and RSUs work together to provide support for reliable mobile services [9]. Map producer launches a task by putting it on a "seed vehicle" and will propagate task information to others in the neighborhood. Then, nearby vehicles communicate with each other by VANET (Vehicular Ad hoc NETworks), forming a group and sensing one segment of road together. We do not concentrate on specific data collecting method (*e.g.,* how to elicit depth from camera data). Instead, we stand in a perspective of crowdsensing mechanism and we are interested in methods that enable vehicles to communicate and collaborate.

As Figure 1 illustrated, we design a basic system composed of message relaying module and collaboration motivating module to enable vehicles to communicate physically and collaborate strategically.

One feature of our system is its assumption where vehicles only communicate by VANET rather than cellular networks. Admittedly, allocating sensing task by cellular networks enables map producers to directly know and control their "sensors" from servers on the cloud. This scheme needs drivers to report their fine-grained trajectories so that map producers could allocate or recommend an appropriate task. However, a majority of drivers are conservative rather than

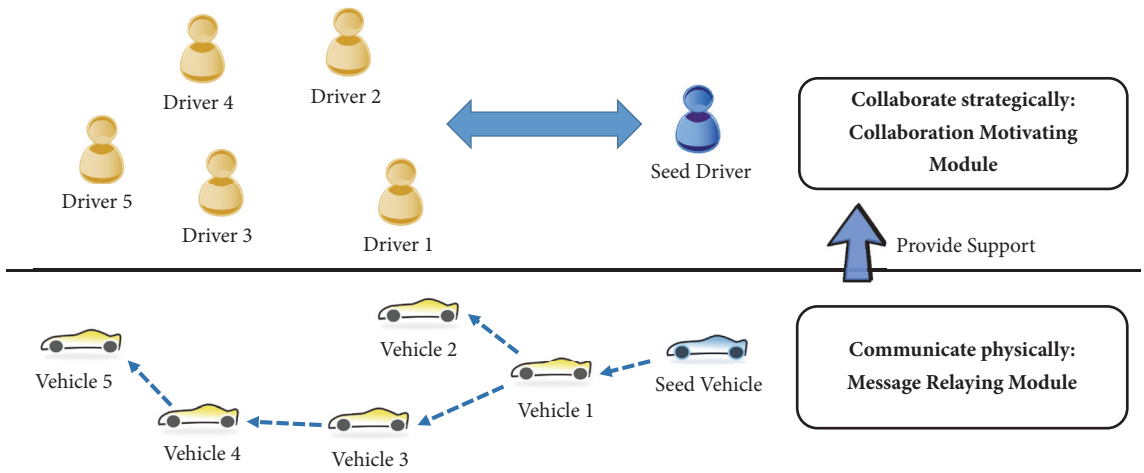


Figure 1: A schematic of group sensing problem. The problem is tackled by two modules.

collaborative if it comes to their private trajectories [10]. The cellular network based scheme therefore may not be fully accepted by most drivers. By contrast, our VANET based local participants selection could alleviate this problem by propagating task information from a seed vehicle and allows drivers to choose whether to join the task with no need of reporting fine-grained trajectories.

Another feature of our system is its game theory model. Vehicles are actually controlled by drivers whose behaviors are self-interest [11, 12]. The collaboration among drivers is modeled as a threshold-based game. We will see how vehicles' sensing costs, the numbers of vehicles, and the task incentives influence the collaboration's equilibrium.

Based on the basic system model, we mainly introduce and discuss the two following questions about group sensing:

(i) *How does task information propagated via VANET?*

Message relaying module is designed to consist of two processes, spread process and back process. In spread process, a message that contains basic task information is firstly given out by seed vehicle and subsequently relayed by other vehicles until it arrives at a region border specified by map producers. We call this spread process because it aims to spread task information to all vehicles within the region. Then it comes to back process; the message is modified and relayed back to seed vehicle. During back process, vehicles may add new information to the message.

(ii) *How could two modules work together?*

Collaboration motivating module helps drivers form a sensing group. Collaboration is relying on message relaying module underpinned to achieve information acquiring and propagating. Existing researches usually discuss relaying message and motivating collaboration separately while an integrated framework is discussed in this paper. The aim of module combination is to endow each networked vehicle ability to make appropriate decision. This combination should meet two requirements: first, the vehicle is automatic enough at control or communication level so that one meta-action could be executed fast enough to support high-level collaboration; second, to maintain the high-level collaboration, vehicle should work smoothly with human driver. Upon these requirements, we have actually added human factor to the collaborating process among crowdsensing.

This paper tries to answer these questions based upon the following assumptions and simplifications:

(i) We assume that each vehicle is equipped with Dedicated Short Range Communication (DSRC) device so that it could send or receive messages within a certain distance. We also assume that a vehicle could well coordinate with driver by advanced HMI technology, which means that two modules can seamlessly cooperate.

(ii) Each driver is assumed to be rational and self-interest. He makes decision to maximize his profits. For each driver, the cost of participating is assumed to conform normal distribution.

(iii) To concentrate on a perspective of map producers, we slightly oversimplify the physical layer by assuming perfect channel without collision and loss in simulation. This is detailed in Section 5.

The rest of this paper is organized as follows. The discussion of related work is presented in Section 2. Section 3 illustrates a framework of two modules. Section 4 describes how two modules work interactively to achieve local participants' selection. Section 5 presents simulation results and analysis. The paper ends with discussions in Section 6 and conclusions in Section 7.

2. Related Work

Leveraging vehicles or sensor networks for environmental sensing has been a hot topic in the recent years. Zhu [13] proposed PUS (Pervasive Urban Sensing) framework and

used probe cars to sense traffic density. The authors designed a compressive sensing algorithm to tackle sparsity of data. Chen [14] considered the distributed estimation problem over relay-assisted wireless sensor networks. Kalman filtering and consensus estimation are leveraged to improve the estimation efficiency and accuracy in large monitoring system. Yuan [15] observed that the distribution of probe vehicles is uneven over space and time. He therefore proposed an adaptive and compressive data gathering scheme based on matrix completion theory. The scheme could largely improve sensing efficiency and quality. Likewise, Du [16] employed matrix completion to estimate unsampled urban data. Besides data processing, there are some works focusing on optimal participants' selection of crowdsensing. Song [17] aims to select the most appropriate participants with different budget constraints. Considering different incentive requirements, associated sensing capabilities, and uncontrollable mobility, a multitask-oriented QoI (Quality of Information) optimization problem is discussed and converted to a nonlinear knapsack problem. Zhang [18, 19] proposed an event-driven QoI-aware participatory sensing framework with energy and budget constraints, where the main method is boundary detection. He [20] devised an efficient local ratio based algorithm and designed a motivating mechanism that decides the fair prices of sensing tasks. In most papers, multitask crowdsensing is modeled as a global optimization problem with need for participants' fine-grained trajectories. By contrast, our paper tries to call up plenty of participants for a single task in a specified region without reporting participants' trajectories.

The application of VANET has been wildly discussed. Papers in research community simply fall into two categories: safety-oriented and non-safety-oriented. The safety-oriented applications intend to avoid collision by broadcasting beacons. Taha [21] proposed a novel protocol for reliable broadcasting of life safety messages in VANET. In case of any dramatic change of speed or moving direction, the vehicle is regarded abnormal and hence it transmits an emergency warning message over the control channel of DSRC. Kumar [22] proposed CarSpeak, a communication system for autonomous driving modified from VANET. By changing the MAC protocol in a content-centric approach, CarSpeak scales the amount of data with the available bandwidth and improves safety of autonomous driving. The non-safety-oriented applications try to relay traffic or other non-real-time information to improve traffic efficiency or driving entertainment. To support high throughput in vehicular networks, Zhou [23] used a geolocation database assisted approach to jointly utilize DSRC and TVWS spectrum. It not only satisfied the dynamic vehicular access requirements but also has adaptive data piping performance. Yuan [24] used VANET for detecting traffic congestion on urban expressways. The scheme develops a spatiotemporal effectiveness model based on the potential energy theory to control the dissemination area and survival time of the congestion information. Similarly, Luo [25, 26] used a SDN-inspired approach to improve network capacity. Wang [27], Zhou [28], Cheng [29], and Yuan [30] use "hard" or "soft" method to expand the communication capacity of Internet of vehicles, respectively.

To improve trustworthiness of VANET information, Chen [31] presented a trust-based message propagation and evaluation framework in VANET, where peers share information regarding road condition or safety and others provide opinions about whether the information can be trusted. Chang [32] proposed a Sybil attack detection mechanism, where footprint of vehicle is employed. Vehicles can generate location-hidden trajectory for location-privacy-preserved identification. PTRS [33] is a robust trust-based relay scheme with an objective to distinguish trust levels of the vehicles.

Our work is also related to MAS (Multiagent System). Supported by today's advancing cloud computing technology, the vehicle could be represented by its virtual agent [34] in Cyber-Physical System. One goal of MAS is to find algorithms to allocate tasks and to design mechanisms to motivate collaboration among rational agents. Based on Stackelberg game, Duan [35] analyzed and compared different incentives mechanisms for a master to motivate the collaboration of smartphone users on both data acquisition and distributed computing applications. More specifically, he proposed a reward-based collaboration mechanism, where the master announces a total reward to be shared among collaborators, and the collaboration is successful if there are enough users wanting to collaborate. The mechanism is novel and practical in many cases including participants' selection of group sensing. Walid [36] proposed a game theoretical approach to tackle the problem of distributed formation of the uplink tree structure among the relay stations in a WiMAX network. From a MAS perspective, the authors modeled the problem as a network formation game, where each relay station aims to maximize its utility from cooperation. Using the proposed dynamics, the relay stations can self-organize into the tree structure. Talal [37] proposed an algorithm enabling multiagent cooperative calculation. Each agent is assigned some part of the calculation such that the agents' shares are exhaustive and disjoint. The algorithm is decentralized and requires no communication between the agents. Jiang [38] considered task allocation and load balance in social networked MAS, where agents always need to negotiate with other agents about their resources. In the presented task allocation model, while a task comes to the system, it is first assigned to a principal agent that has high contextual enrichment factor for the required resources. Then the principal agent will negotiate with its contextual agents to execute the assigned task. The authors also considered load balancing to avoid overconvergence of tasks at certain agents that are rich in contextual resources. Motivated by above papers, this paper has discussed smart vehicle collaboration from a MAS perspective.

3. Framework

Group sensing aims to collect data in a specified region, thus focusing on local participants' selection. The participants' selection of group sensing involves two modules. Collaboration motivating module aims to motivate plenty of collaborators to form a sensing group. The math of motivating

mechanism will be illustrated. Another module, namely, message relaying module, provides VANET-based media to execute motivating mechanism.

3.1. Collaboration Motivating Module. HD map data need to be batch-processed to reduce noise and correct error. One single vehicle's data are less valued than a batch of data from a sensing group. Therefore, a sensing group usually consists of many vehicles that share a similar traverse in both space and time. Map producers prefer to harvest a batch of data by calling up a group of vehicles.

Based on above facts, the essence of group sensing is threshold based game [35]. Map producers are only willing to give incentives when a certain number of vehicles (or drivers), denoted as n_0, participate in the sensing task. Consider that a set of N vehicles $\mathbb{N} = \{1, 2, \ldots, N\}$ have been notified of the task and each vehicle knows N and n_0. $vehicle_i$ ($i \in \mathbb{N}$) has a collaboration cost C_i, if it collaborates (*i.e.,* joins the sensing group). We assume that all collaboration costs conform normal distribution, known as $C \sim N(\mu, \sigma)$.

Since map producers need a sensing group, the sensing task will not be approved unless at least n_0 of N vehicles become participants. Moreover, each driver decides whether to participate by predicting others' decisions. These facts make a threshold-based game, which consists of two phases. In phase 1, seed vehicle gives out task information (R, n_0). R are total incentives given by map producers. In phase 2, after (R, n_0) is propagated and known by N vehicles, each driver decides whether to participate. If the number of participants, denoted by n, is bigger than n_0, then the task is approved; otherwise the task is abandoned by map producer. If the task is approved, incentives R are distributed equally to participants because each participant works in the similar road segment and time. As a result, participating $vehicle_i$ ($i \in \mathbb{N}$) undertakes cost C_i and gets incentives of R/n, with its overall profits being

$$\left(\frac{R}{n} - C_i\right) Pos\{n \geq n_0\} \tag{1}$$

where $Pos\{X\}$ is the indicator function (equals 1 when event X is true and equals 0 when event X is false). One could make negative profits even though he joins the collaboration, if R/n is smaller than C_i. If collaboration is not approved, profit is 0.

It is actually the driver who decides whether the vehicle collaborates or not. Driver is self-interest and makes own decision with limited information. All drivers' rational behaviors make a threshold-based game. The result of this game is subtle; for example, when n ($n \gg n_0$) drivers decide to collaborate, they may even gain a negative profit because R/n is so small that it even could not meet sensing cost. On the other hand, if only n ($n \ll n_0$) drivers collaborate, it is hard to persuade map producers to approve such task because the map producers could not harvest enough data. Therefore, how to motivate an appropriate number of drivers to collaborate is a critical problem in group sensing.

3.2. Message Relaying Module. Message relaying module executes two physical processes: spread process and back process. Map producers have owned a few seed vehicles. Reporting their fine-grained trajectories, seed vehicles are well connected with servers and controlled by map producer. Server monitors seed vehicles' states and launch a task package to one specific seed vehicle. As Table 1 shows, the package contains four kinds of information: (1) The first kind is sensing segment. Participants only need to collect data within a specified region. The region is the sensing segment of task. (2) The second kind is preparing segment. Before entering sensing segment, task information is propagated and some participants are generated from normal vehicles. Such process, including message relaying and collaboration motivating, takes place on preparing segment. As shown in Figure 2, preparing segment should be upstream of sensing segment because vehicles would have already traversed a length of road segment when preparing work is done. Since our topic is communicating and collaborating mechanism rather than sensing techniques, this paper concentrates on preparing segment. (3) The third kind is incentive. Map producer is willing to pay incentives R if he approves the task. (4) The fourth kind is threshold of participants. The task is successfully approved only when number of participants n meets the threshold n_0.

Table 1: Content of task/collaboration message.

Task message	Collaboration message
Current relaying vehicle's location (x, y)	
Seed vehicle's location (x_0, y_0)	
Potential participants list $\mathbb{L}$	
Reward R	Participants list $\mathbb{L}'$
Threshold of participants' number n_0	/
Sensing segment: a rectangle by (x_1^s, y_1^s), (x_2^s, y_2^s)	/
Preparing segment: a rectangle by (x_1^p, y_1^p), (x_2^p, y_2^p)	/

Figure 2 also illustrates spread process and back process. Spread process starts with a TM (task message) broadcast by seed vehicle. A TM is relayed one-by-one until it reaches the border of preparing segment; thus all vehicles in preparing segment are notified of the task. The border guard, a vehicle located at the border, modifies TM to CM (collaboration message). He also starts back process by reversing relaying direction and sending CM back to seed vehicle. During back process, the vehicle that receives CM decides whether to join the group sensing and adds its decisions to CM. Seed vehicle receives the final CM and knows how many participants it has motivated. The server then could decide if the task is approved.

Note that, in our threshold-based game, all drivers make decisions by predicting others' decisions. To give participants list $\mathbb{P}$ to seed vehicle, decisions are recorded in $\mathbb{P}$, which is relayed by vehicles. This means that some vehicles could "peep" decisions of former vehicles, which could lead to more complex models (*e.g.,* social learning [39]) and fails our threshold-based game model. To prevent this cheating, seed

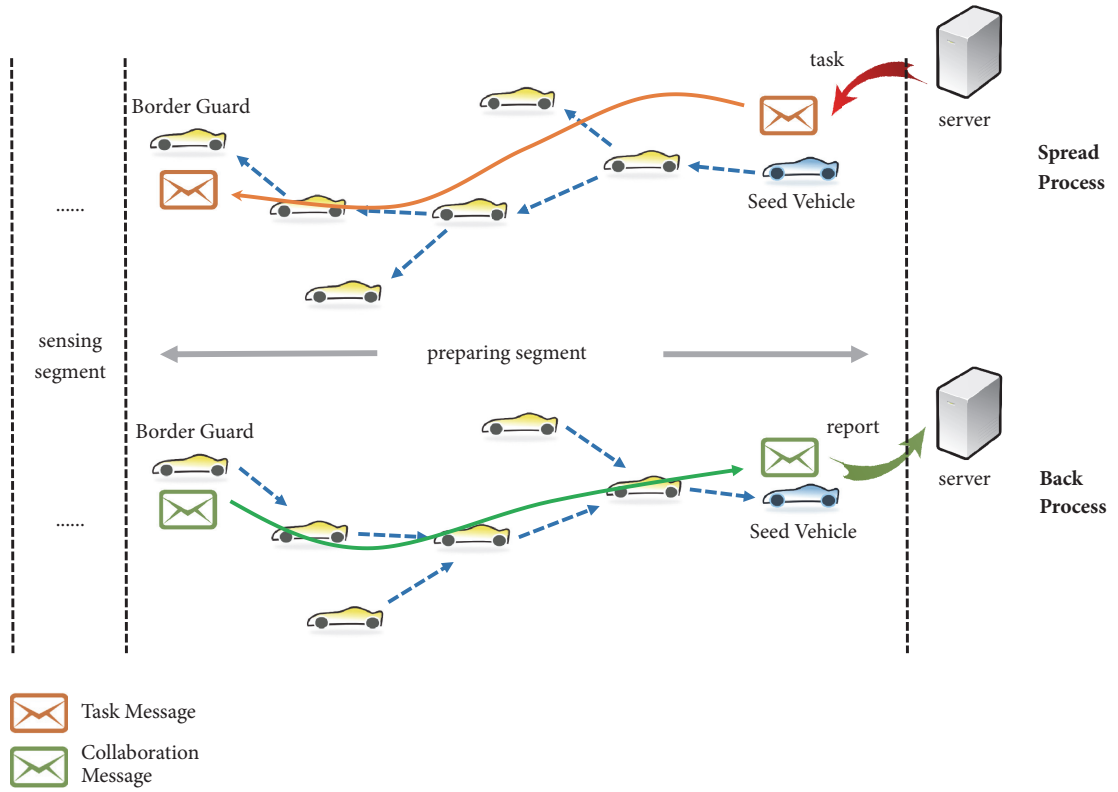


Figure 2: Illustration of spread process and back process.

and normal vehicles can use public-key cryptography [40] to tackle this problem. Normal vehicles encrypt their decisions with public key, while only seed vehicle can decipher the decisions with private key.

3.3. Combining Modules with Flexibility. Participants' selection needs two modules, namely, message relaying module and collaboration motivating module, to work together. The former module is in charge of propagating information among vehicles, and the latter module is responsible for coordinating drivers' behaviors.

In VANET, messages could be propagated by contention-based relaying. *That is,* only the periodical winner of contention could be relaying node. Another relaying method is built upon cluster structure, which helps vehicles reduce transmission data volume and improve communication efficiency. Supported by message relaying module, group formation still needs a mechanism to motivate and select appropriate participants. The motivating method could be realized in either a centralized or distributed way. Overall, our participants' selection framework is composed of two modules, and there are two alternative methods for each module, respectively.

Decoupling participants' selection into two modules endows the system flexibility, enabling map producers to choose optimal combination from several alternatives according to service requirements. We will discuss three combinations: centralized selection, distributed selection, and clustered selection. See Figure 3 for understanding how the combination of different modules can make three distinct participants' selection schemes.

4. Participants' Selection of Group Sensing

This section presents participants' selection schemes derived from one framework. Before diving into three schemes, some preliminaries on relaying methods should be introduced.

We only consider message relaying in "simple road condition." Simple road condition is referred to (1) the road being straight or having small curvature so that relaying method would not fail by extreme road shapes, such as zigzag or circle, and (2) neglecting intersection in preparing segment. In this way, we could simplify simulation by leaving out the situations where lots of vehicles join and leave. Furthermore, we could simplify simulation by decomposing complex road condition (zigzag, circle, etc.) into several simple road conditions. This divide-and-conquer strategy enables our proposed method to be adjustable in most transportation situations. Two examples of simple road condition are in Figure 4.

Contention-based relaying is motivated by geocasting [41], where message is relayed from original location P_0 to P_{n+1} by a node chain. As described in Figure 5(a), seed vehicle is located at $P_0(x_0, y_0)$, previous relaying node's location is $P_n(x_n, y_n)$, and current receiver's location is $P_{n+1}(x_{n+1}, y_{n+1})$. Then we have $L_{0,n} = \|P_0P_n\|$ and $L_{n,n+1} = \|P_nP_n + 1\|$. The angle $\theta = \angle P_0P_nP_{n+1}$ is calculated by the following equation:

$$\cos\theta = \frac{L_{0,n}^2 + L_{n,n+1}^2 - L_{0,n+1}^2}{2L_{0,n}L_{n,n+1}} \quad (2)$$

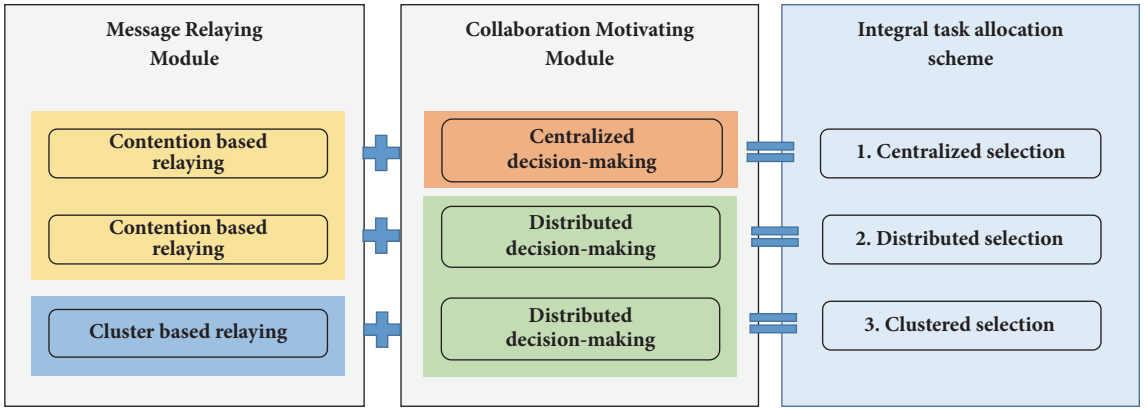


Figure 3: Three participants' selection schemes. Each scheme is a combination of two modules. Each module includes two methods.

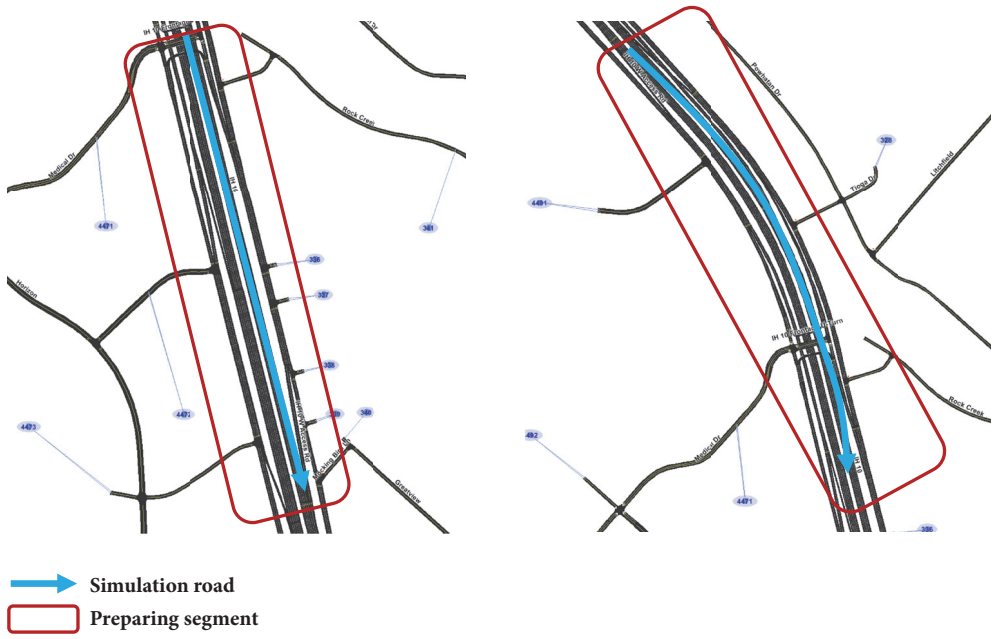


Figure 4: Simple road condition. Left: straight road. Right: road with small curvature. Both examples do not consider intersections.

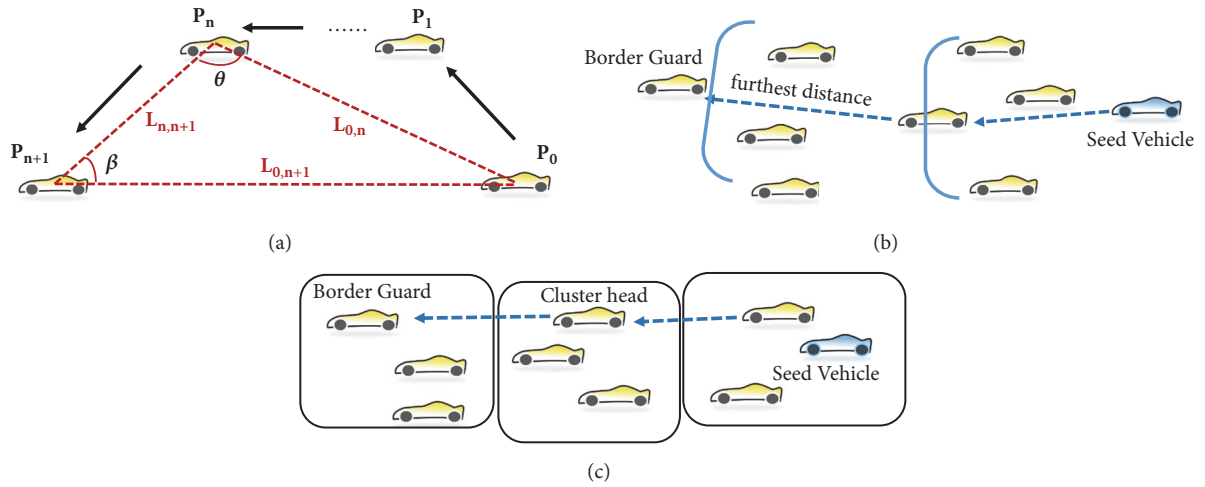


Figure 5: Message relaying methods. (a) Node chain. (b) Contention-based relay. (c) Cluster-based relay.

(1) Seed vehicle broadcasts TM.
(2) After receiving TM, $vehicle_i$ calculates $\cos\theta$. If $\cos\theta > 0$, goto (6).
(3) $vehicle_i$ calculates L', then contends to be relaying node. If fails to be, goto (6).
(4) $vehicle_i$ claims himself the new relaying node.
(5) If $vehicle_i$ is not on the border, it relays TM to downstream.
(6) end.

Process 1: Contention-based relaying.

It indicates that message is relayed away from original node P_0 if $\cos\theta < 0$. The next relaying node is generated by downstream nodes' contention. The projection of $L_{n,n+1}$ on $L_{0,n}$ is $L' = L_{n,n+1} \times \| \cos\beta \|$. Each downstream node calculates its back off time $t_{notifying}$ by [42]

$$t_{notifying} = T_{max} \cdot \left(1 - \frac{L'}{L_{max}}\right) \tag{3}$$

where T_{max} and L_{max} are predefined values and $\cos\beta$ could be calculated in a similar way to $\cos\theta$. According to (3), the most downstream node has the minimum $t_{notifying}$, thus having the highest priority to be next relaying node. Figure 5(b) illustrates how messages are delivered by winners of contention. Usually, a message could reach the destination by several hops.

Another prevailing relaying method is cluster-based relaying. A virtual network infrastructure should be created through the clustering of nodes in order to provide stability and scalability. See Figure 5(c) for an illustration of cluster-based relaying. Each cluster has a cluster head, which is responsible for intra- and intercluster coordination in the network management functions. Nodes inside a cluster communicate by direct links.

4.1. Distributed Selection. Distributed selection is a combination of contention-based relaying and distributed decision-making. As Process 1 illustrated, TM is propagated by contention-based relaying to avoid broadcast storm and ensures that all drivers in assigned local region receive TM.

Then TM is modified to CM by border guard; all drivers decide whether to be participants in a distributed manner; the process of distributed decision-making is a game, where all players have the following equilibrium:

(i) No collaboration: if $R < n_0\mu$, the reward is not enough to satisfy driver's expectation. No driver is willing to join collaboration

(ii) Mix strategy equilibrium): if $n_0\mu \le R \le N\mu$, each driver joins collaboration by probability p^*, where p^* is the unique solution of

$$E_m\left(\left(\frac{R}{m+1} - \mu\right) Pos\left(m + 1 \ge n_0\right)\right) = 0 \tag{4}$$

m follows binomial distribution $B(N-1, p)$. Equation (4) indicates that p^* is an equilibrium where each driver has an expected payoff of 0.

Furthermore, we also consider the scenario where each player knows his own cost c_i. Then there exists a stronger constraint than (5); a driver will only collaborate when $C_i \le \gamma^*(R)$. $\gamma^*(R)$ is unique solution of the equation

$$E_m\left(\left(\frac{R}{m+1} - \gamma\right) Pos\left(m + 1 \ge n_0\right)\right) = 0 \tag{5}$$

m follows binomial distribution $B(N-1, F(\gamma))$. Note that $F(\gamma) = P(C_i \le \gamma)$. We can learn that all members have the same information and estimation about others. Therefore, all members have the same $\gamma^*(R)$. However, different members will still make different decisions according to their private cost.

Finally, if a driver decides to collaborate, he sends a decision message to seed vehicle by contention-based relaying again.

4.2. Centralized Selection. Centralized selection is composed of contention-based relaying and centralized decision-making. It means that all potential participants send their messages into seed vehicle that collects all messages and selects participants in a centralized manner.

First, task message is propagated by process 1. After that, willing driver simply sends a message that includes his private cost and other information to seed vehicle.

Seed vehicle makes decisions for all drivers, since it knows all drivers cost set $\mathbb{C} = \{C_{i_1}, C_{i_2}, \ldots, C_{i_N}\}$, which could be contained in $\mathbb{P}$. Let us just assume that there is no same cost value among cost set. Then costs could be sorted in ascending order in $\mathbb{C}' = \{C_1, C_2, \ldots, C_N\}$. Seed vehicle will choose 1^{st} to n_0^{th} smallest cost in $\mathbb{C}'$ to be participants. The collaboration has optimal result under centralized decision-making:

(i) Optimal collaboration: each driver with $C_i \le C_{n_0}$ will be chosen and afterwards receives C_{n_0} incentives. The remaining $N - n_0$ drivers do not collaborate.

Each selected participant will receive C_{n_0} incentives, since we distribute incentives equally. By centralized selection, no driver needs to send his decision, since seed vehicle has already known the collaboration results.

4.3. Clustered Selection. Clustered selection scheme has a different relaying method from above two schemes. As Figure 3 illustrates, the main difference between clustered selection and the other two lies in the question of how messages are relayed. Clustered selection maintains a hierarchical structure, including cluster head and cluster members. Cluster head takes responsibility for intercluster communication, thus reducing message collision and improving network efficiency.

In this case, decision-making is still distributed. The integral scheme is described in process 2. The formation of cluster has been wildly researched [43] and thus is not discussed in this paper.

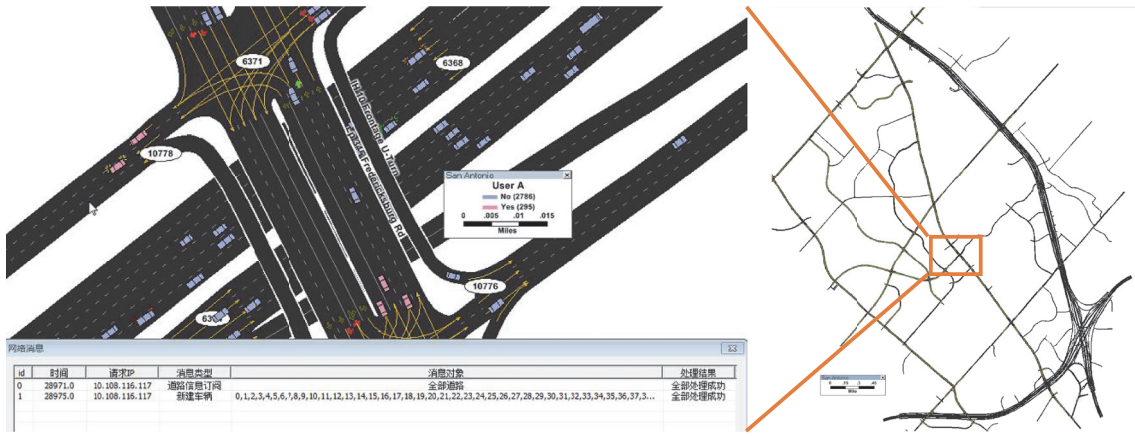


Figure 6: TransModeler provides a virtual transportation scenario.

(1) Seed vehicle broadcasts TM.
(2) After receiving TM, cluster head $vehicle_i$ informs cluster members the TM. It also calculates $\cos\theta$. If $\cos\theta > 0$, goto (4).
(3) If cluster head $vehicle_i$ is not on the border, it relays TM to a cluster head of downstream.
(4) end.

Process 2: Cluster-based relaying.

Finally, all members are informed of task information. Being similar to Section 4.1, distributed decision-making is leveraged to select appropriate participants. The results of decisions are sent back to seed vehicle by cluster-based relaying again.

5. Simulation and Analysis

We simulate the framework with TransModeler, a transportation simulating environment. Figure 6 is an illustration of our simulator. Based on its API, we achieve logical communication framework over vehicles to support network layer simulation. Additionally, we assume that each relaying consumes the same time slot and assume perfect channel without collision and loss in simulation. With such framework, we could simulate network layer of VANET communication in a virtual transportation scenario. Since the framework that integrates message relaying and collaboration motivating has been little discussed previously, no previous work could be compared with it. However, we still make comparison among our self-proposed methods and try to make this paper self-contained.

5.1. Effect Analysis of Motivating. We first analyze two motivating methods separately. Parameters are explained in Table 2 and the simulation results are demonstrated in Figure 7.

5.1.1. Result of Distributed Decision-Making. Figure 7(a) shows $R - \rho^*$ diagram depicting how driver's collaboration probability reacts with reward. When $R < 20$, reward is smaller than $n_0\mu$; no driver collaborates. When $20 \le R < 35$, reward is enough to motivate part of drivers. And it shows that the larger reward is, the higher probability is that drivers will collaborate. When $R \ge 35$, *that is,* $R \ge N\mu$, all drivers will collaborate; thus $p^* = 1$. Similarly, Figure 7(b) shows that larger μ leads to lower collaboration probability. It means that, given the same reward R, fewer drivers will collaborate if the mean of costs goes higher. These findings in two figures are consistent with human intuition.

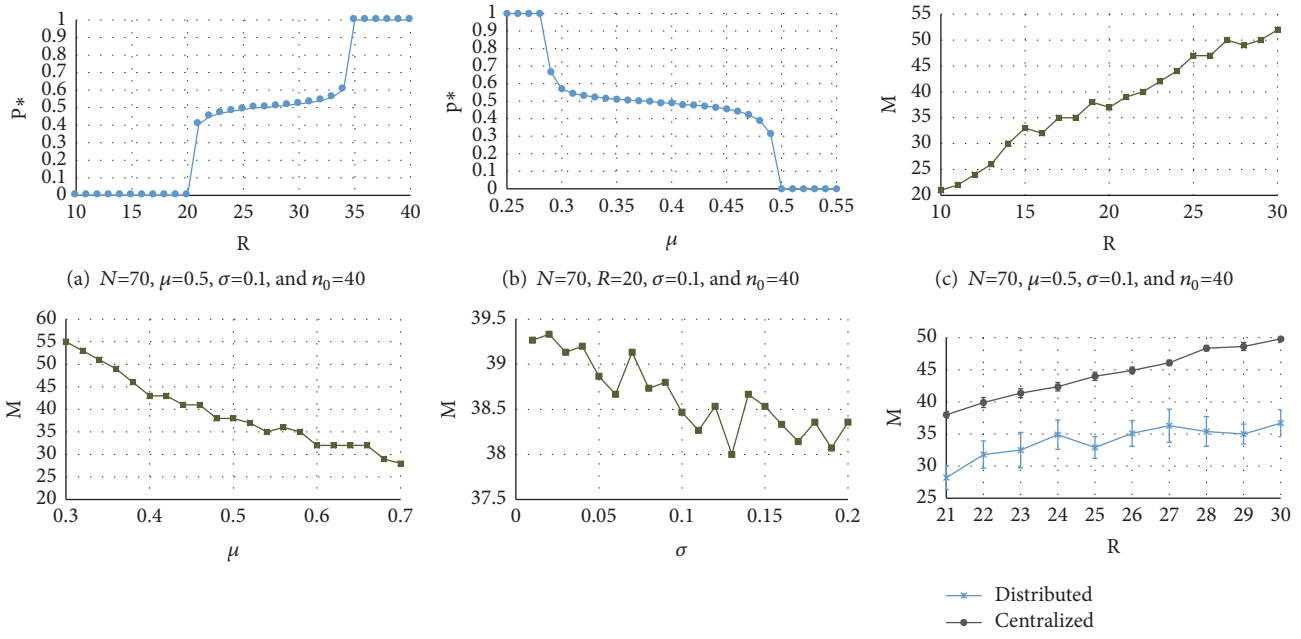
Table 2: Parameters in motivating simulation.

R	Reward or incentive
N	The number of potential participants
μ	Mean of all potential participants' cost
σ	Std. of potential participants' cost
n_0	Threshold of participants' number
M	The number of final participants

5.1.2. Result of Centralized Decision-Making. Figure 7(c) describes R's influence on M. As R increases, M also increases. More reward could motivate more participants. In Figure 7(d), we could see that higher μ leads to smaller M. Figure 7(e) shows that larger σ (which means that the costs values are more dispersed) leads to smaller M. In the case in Figure 7(e), $n_0 = 40 > N/2$, so it needs to motivate more than $N/2$ drivers. With the increase of σ, ${n_0}^{th}(n_0 > N/2)$ smallest cost $C_{n_0}(C_{n_0} - \mu > 0)$ becomes bigger, and the value of R/C_{n_0} gets smaller; thus participants' number $\lfloor R/C_{n_0} \rfloor$ decreases.

5.1.3. Comparison of Two Motivating Methods. We set $N = 70$, $n_0 = 40$, $R = 20$, $\mu = 0.5$, and $\sigma = 0.1$ and use M as indicator of motivating effect. The simulation is repeated for 20 times. Figure 7(f) shows that centralized decision-making achieves larger M than distributed decision-making.

It should be mentioned that a task is approved by map producer only when $M \ge n_0 = 40$. However, as Figure 7(f) shows, no method could achieve $M \ge n_0$. For centralized decision-making, map producer should prepare $R' = 22 \sim 23$



(a) N=70, μ=0.5, σ=0.1, and n_0=40 (b) N=70, R=20, σ=0.1, and n_0=40 (c) N=70, μ=0.5, σ=0.1, and n_0=40

(d) N=70, R=20, σ=0.1, and n_0=40 (e) N=70, R=20, μ=0.5, and n_0=40 (f) N=70, R=20,μ=0.5, σ=0.1, and n_0=40

Figure 7: Parameters and their influences on motivating effect.

(while the planned reward is $R = 20$) to successfully motivate $n_0 = 40$ collaborators. Distributed decision-making could only motivate 3/4 participants of centralized method. To conclude, map producers should always give "extra reward" besides planned reward to reach expected M.

In this section, only two basic motivating methods are discussed. We did not discuss three participants' selection schemes, because motivating effect is only depending on the collaboration motivating module and is irrelevant to message relaying module. However, in Section 5.2, we will find that the communication overhead is influenced by both modules; we therefore need to compare three integral schemes.

5.2. Communication Overhead of Message Relaying. The communication simulation scenario is 1000-meter road (in Figure 4). DSRC range is set as 150 m. Final results are averaged by 20 simulations. Performance analysis is based on three indicators. (1) First indicator is *Slots Duration*. We assume that each relay consumes one time slot. During whole process, the counted number of slots is slots duration. (2) Second indicator is *Total Transmission bytes*. The total bytes are gross of all messages bytes. (3) Third indicator is *Total Relaying Times*. It shows how many times of relay for all vehicles have taken place during the process.

5.2.1. Communication Overhead on Straight Road. Figure 8 shows communication performance of three schemes on straight road (in Figure 4 (left)). Figure 8(a) shows that, for three schemes, total transmission bytes go up with vehicle density. The reason is intuitive; more vehicles cause more transmitted content. Centralized selection has the largest bytes because all vehicles send messages to seed vehicle.

In Figure 8(b), slots duration of each scheme is a constant regardless of vehicle density. Clustered selection consumes more slots than centralized and distributed selection because it spares slots on cluster maintenance.

In Figure 8(c), centralized selection has the largest total relaying times. Because each potential participant generates a message and sends it to seed vehicle, centralized selection has larger total relaying times than the other two.

5.2.2. Communication Overhead on Curve Road. Robustness of scheme is reflected on communication performance in Figure 9. A robust scheme is able to work well on both straight (in Figure 4 (left)) and curve roads (in Figure 4 (right)). In all three schemes, the indicator in straight road is lower than that in curve road. The performance difference between two road conditions is at most 20%. This figure shows that, for each local participants' selection scheme, communication overhead on curve road is a little higher than straight road, demonstrating that these schemes could still work fine on roads with small curvature.

6. Discussion

Given the same reward, centralized decision-making could motivate more participants than distributed decision-making. The reason is that distributed decision-making is executed by each potential participant who lacks a global view of all information. Derived from distributed decision-making, clustered selection scheme and distributed selection have the same performance on motivating. Centralized selection is better than the other two if only motivating effect is considered.

Besides motivating effect, communication overhead also deserves attention. If communication overhead is considered alone, distributed selection should be the first choice, since it performs well in three indicators. Centralized selection

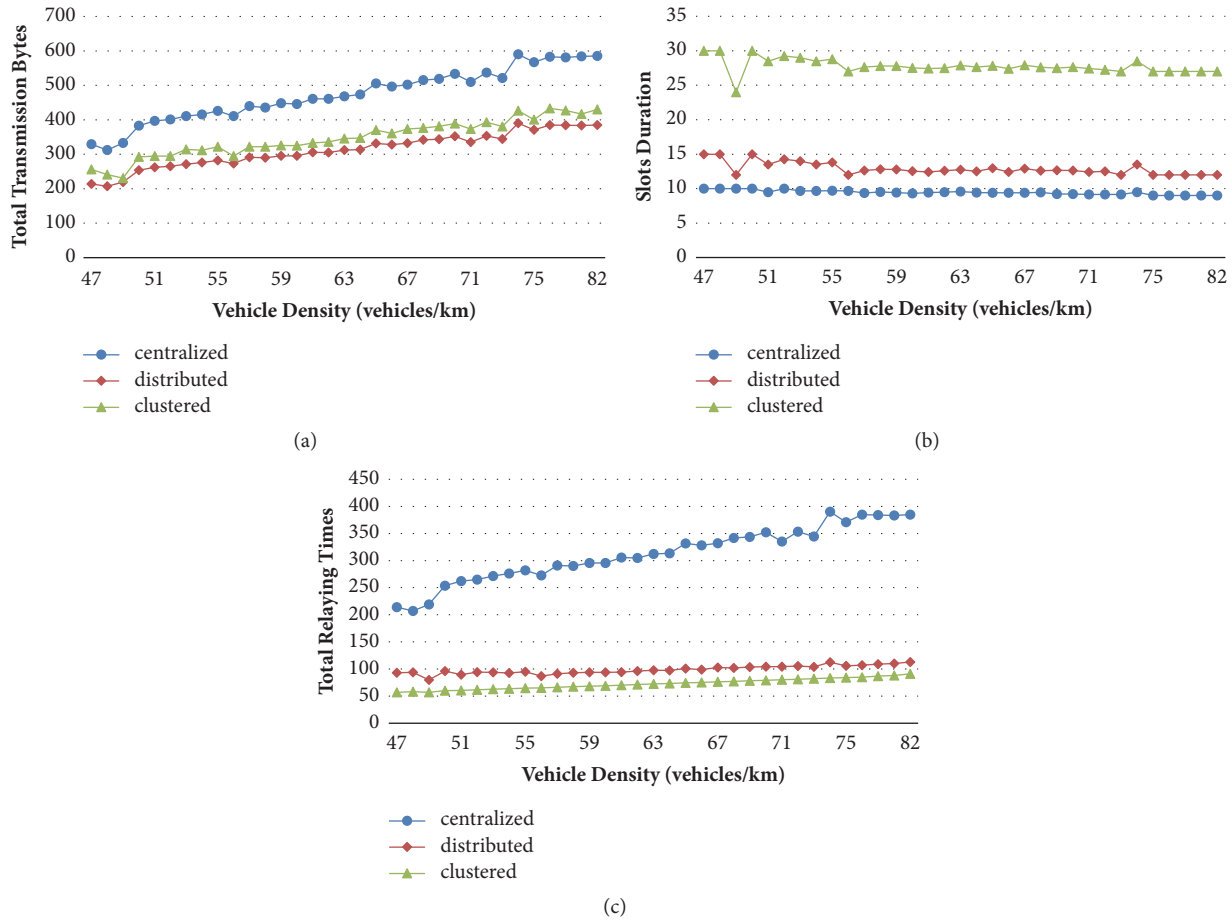


Figure 8: Communication overhead of three participants' selection. (a) Centralized selection has the largest bytes overhead because each vehicle sends its information to seed vehicle. (b) Clustered selection has the longest slots duration because of its stable structure. (c) Centralized selection consumes the largest total relaying times for the same reason as (a).

Table 3: Comprehensive comparison of three schemes.

\	Centralized selection	Distributed selection	Clustered selection
moti. perf.	Good	Normal	Normal
comm. perf.	Limited	Good	Normal

and clustered selection are second-class citizens and are only suitable for particular scenarios. For example, if a map producer hopes the participants' selection to be finished as soon as possible, then he would like to use centralized selection. Despite its shortcomings in total transmission bytes and total relaying times, centralized selection has the shortest slots duration.

In Table 3, we give a comprehensive comparison of motivating performance and communication performance. Centralized selection has a good performance in motivating but has a heavy communication overhead. Clustered selection requires more incentives and less communication overhead than centralized method. Distributed selection seems to be the first-class choice because it works fine in both motivating and communicating. It should be noted that all three selection schemes could be realized from one single framework with few modifications of two modules. The flexibility of design framework enables map producers to select participants adaptively according to scenario requirements.

7. Conclusions

This paper has discussed local participants' selection of crowdsensing. We have proposed a flexible framework that could support three kinds of selection schemes with respect to different service requirements. The framework is consisted of two modules. For collaboration motivating module, we introduced two motivating methods: centralized decision-making and distributed decision-making. For message relaying module, we introduced two methods: contention-based relaying and cluster-based relaying. Three combinations of two modules are discussed and simulated. The result shows that distributed selection is usually the first-class choice because of its fine performances on communicating and motivating. Our research analyzed group sensing problem from both communication and collaboration perspectives,

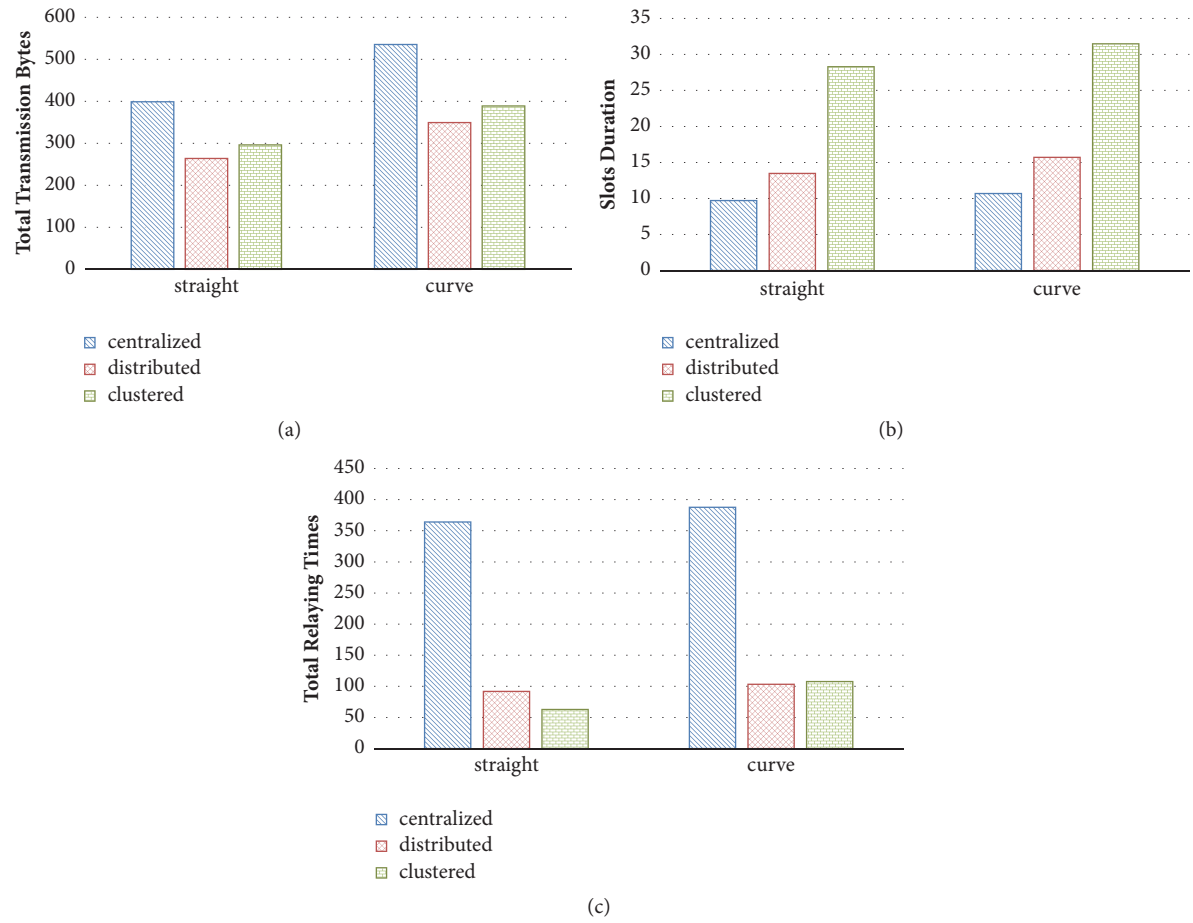


Figure 9: Communication overhead of straight and curve road: (a) comparison of total transmission bytes between straight and curve. (b) Comparison of slots duration between straight and curve. (c) Comparison of total relaying times between straight and curve. Three schemes still work fine in road with small curvature.

providing a primary but integrated reference for data service providers or crowdsensing operators, especially for map producers.

In the future, we will integrate ns-2 [44] into our framework to enable accurate time-simulation. We will also try to refine spread process and back process into "one-shot process."

Data Availability

No additional data are available.

Disclosure

Part of this work has been published by 2017 IEEE International Conference on Services Computing [45].

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work is supported by National Science and Technology Major Project of the Ministry of Science and Technology of China (Grant no. 2016ZX03001025-003) and the Natural Science Foundation of Beijing (Grant no. 4181002).

References

[1] R. K. Ganti, F. Ye, and H. Lei, "Mobile crowdsensing: current state and future challenges," *IEEE Communications Magazine*, vol. 49, no. 11, pp. 32–39, 2011.

[2] X. Hou, Y. Li, M. Chen, D. Wu, D. Jin, and S. Chen, "Vehicular fog computing: a viewpoint of vehicles as the infrastructures," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 6, pp. 3860–3873, 2016.

[3] N. Cheng, N. Lu, N. Zhang, T. Yang, X. S. Shen, and J. W. Mark, "Vehicle-assisted device-to-device data delivery for smart grid," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 4, pp. 2325–2340, 2016.

[4] T. Lei, S. Wang, J. Li, and F. Yang, "A cooperative route choice approach via virtual vehicle in IoV," *Vehicular Communications*, 2017.

[5] S. S. Kanhere, "Participatory sensing: crowdsourcing data from mobile smartphones in urban spaces," in *Proceedings of the 12th IEEE International Conference on Mobile Data Management (MDM '11)*, vol. 2, pp. 3–6, June 2011.

[6] Q. Yuan, H. Zhou, J. Li, Z. Liu, F. Yang, and X. S. Shen, "Toward Efficient Content Delivery for Automated Driving Services: An Edge Computing Solution," *IEEE Network*, vol. 32, no. 1, pp. 80–86, 2018.

[7] J. Levinson, J. Askeland, J. Becker et al., "Towards fully autonomous driving: systems and algorithms," in *Proceedings of the IEEE Intelligent Vehicles Symposium (IV '11)*, pp. 163–168, June 2011.

[8] M. Campbell, M. Egerstedt, J. P. How, and R. M. Murray, "Autonomous driving in urban environments: Approaches, lessons and challenges," *Philosophical Transactions of the Royal Society A: Mathematical, Physical & Engineering Sciences*, vol. 368, no. 1928, pp. 4649–4672, 2010.

[9] A. Zhou, S. Wang, Z. Zheng, C.-H. Hsu, M. R. Lyu, and F. Yang, "On cloud service reliability enhancement with optimal resource usage," *IEEE Transactions on Cloud Computing*, vol. 4, no. 4, pp. 452–466, 2016.

[10] B. Zhang, C. H. Liu, J. Lu et al., "Privacy-preserving QoI-aware participant coordination for mobile crowdsourcing," *Computer Networks*, vol. 101, pp. 29–41, 2016.

[11] R. Cavallo, D. C. Parkes, and S. Singh, *Optimal coordinated planning amongst self-interested agents with private state*, 2012.

[12] E. Ephrati and J. Rosenschein, "Distributed consensus mechanisms for self-interested heterogeneous agents," in *Proceedings of the [1993] International Conference on Intelligent and Cooperative Information Systems*, pp. 71–79, Rotterdam, Netherlands.

[13] Y. Zhu, X. Liu, and Y. Wang, "Pervasive urban sensing with large-scale mobile probe vehicles," *International Journal of Distributed Sensor Networks*, vol. 2013, 2013.

[14] C. Chen, J. Yan, N. Lu, Y. Wang, X. Yang, and X. Guan, "Ubiquitous monitoring for industrial cyber-physical systems over relay—assisted wireless sensor networks," *IEEE Transactions on Emerging Topics in Computing*, vol. 3, no. 3, pp. 352–362, 2015.

[15] Q. Yuan, Z. Liu, J. Li, S. Yang, and F. Yang, "An adaptive and compressive data gathering scheme in vehicular sensor networks," in *Proceedings of the In Parallel and Distributed Systems (ICPADS), 2015 IEEE 21st International Conference*, pp. 207–215, Melbourne, VIC, December 2015.

[16] R. Du, C. Chen, B. Yang, N. Lu, X. Guan, and X. Shen, "Effective urban traffic monitoring by vehicular sensor networks," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 1, pp. 273–286, 2015.

[17] Z. Song, C. H. Liu, J. Wu, J. Ma, and W. Wang, "QoI-aware multitask-oriented dynamic participant selection with budget constraints," *IEEE Transactions on Vehicular Technology*, vol. 63, no. 9, pp. 4618–4632, 2014.

[18] B. Zhang, Z. Song, C. H. Liu, J. Ma, and W. Wang, "An Event-Driven QoI-Aware Participatory Sensing Framework with Energy and Budget Constraints," *ACM Transactions on Intelligent Systems and Technology*, vol. 6, no. 3, pp. 1–19, 2015.

[19] C. H. Liu, B. Zhang, X. Su, J. Ma, W. Wang, and K. K. Leung, "Energy-aware participant selection for smartphone-enabled mobile crowd sensing," *IEEE Systems Journal*, vol. PP, no. 99, 2015.

[20] S. He, D.-H. Shin, J. Zhang, and J. Chen, "Toward optimal allocation of location dependent tasks in crowdsensing," in *Proceedings of the 33rd IEEE Conference on Computer Communications (INFOCOM '14)*, pp. 745–753, IEEE, Toronto, Canada, May 2014.

[21] M. M. I. Taha and Y. M. Y. Hasan, "VANET-DSRC protocol for reliable broadcasting of life safety messages," in *Proceedings of the ISSPIT 2007 - 2007 IEEE International Symposium on Signal Processing and Information Technology*, pp. 104–109, Egypt, December 2007.

[22] S. Kumar, L. Shi, N. Ahmed, S. Gil, D. Katabi, and D. Rus, "CarSpeak: A content-centric network for autonomous driving," in *Proceedings of the ACM SIGCOMM 2012 Conference Applications, Technologies, Architectures, and Protocols for Computer Communication, SIGCOMM 2012*, pp. 259–270, fin, August 2012.

[23] H. Zhou, N. Cheng, Q. Yu, X. S. Shen, D. Shan, and F. Bai, "Toward multi-radio vehicular data piping for dynamic DSRC/TVWS spectrum sharing," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 10, pp. 2575–2588, 2016.

[24] Q. Yuan, Z. Liu, J. Li, J. Zhang, and F. Yang, "A traffic congestion detection and information dissemination scheme for urban expressways using vehicular networks," *Transportation Research Part C: Emerging Technologies*, vol. 47, no. 2, pp. 114–127, 2014.

[25] G. Luo, J. Li, L. Zhang, Q. Yuan, Z. Liu, and F. Yang, "sdnMAC: A Software-Defined Network Inspired MAC Protocol for Cooperative Safety in VANETs," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 6, pp. 2011–2024, 2018.

[26] G. Luo, S. Jia, Z. Liu, K. Zhu, and L. Zhang, "SdnMAC: A software defined networking based MAC protocol in VANETs," in *Proceedings of the 24th IEEE/ACM International Symposium on Quality of Service, IWQoS 2016*, chn, June 2016.

[27] S. Wang, C. Fan, C.-H. Hsu, Q. Sun, and F. Yang, "A vertical handoff method via self-selection decision tree for internet of vehicles," *IEEE Systems Journal*, 2014.

[28] H. B. Zhou, B. Liu, T. H. Luan et al., "ChainCluster: engineering a cooperative content distribution framework for highway vehicular communications," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 6, pp. 2644–2657, 2014.

[29] N. Cheng, N. Lu, N. Zhang, X. S. Shen, and J. W. Mark, "Opportunistic WiFi offloading in vehicular environment: A queueing analysis," in *Proceedings of the 2014 IEEE Global Communications Conference, GLOBECOM 2014*, pp. 211–216, Austin, Tex, USA, December 2014.

[30] Q. Yuan, J. Li, Z. Liu, and F. Yang, "Space and Time Constrained Data Offloading in Vehicular Networks," in *Proceedings of the 2016 IEEE 18th International Conference on High Performance Computing and Communications; IEEE 14th International Conference on Smart City; IEEE 2nd International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, pp. 398–405, Sydney, Australia, December 2016.

[31] C. Chen, J. Zhang, R. Cohen, and P.-H. Ho, "A trust modeling framework for message propagation and evaluation in VANETs," in *Proceedings of the 2nd International Conference on Information Technology Convergence and Services (ITCS '10)*, IEEE, August 2010.

[32] S. Chang, Y. Qi, H. Zhu, J. Zhao, and X. Shen, "Footprint: detecting Sybil attacks in urban vehicular networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 6, pp. 1103–1114, 2012.

[33] H. Hu, R. Lu, C. Huang, and Z. Zhang, "PTRS: A privacy-preserving trust-based relay selection scheme in VANETs," *Peer-to-Peer Networking and Applications*, vol. 10, no. 5, pp. 1204–1218, 2017.

[34] F. Yang, S. Wang, J. Li, Z. Liu, and Q. Sun, "An overview of internet of vehicles," *China Communications*, vol. 11, no. 10, pp. 1–15, 2014.

[35] L. Duan, T. Kubo, K. Sugiyama, J. Huang, T. Hasegawa, and J. Walrand, "Motivating smartphone collaboration in data acquisition and distributed computing," *IEEE Transactions on Mobile Computing*, vol. 13, no. 10, pp. 2320–2333, 2014.

[36] W. Saad, Z. Han, M. Debbah, A. Hjorungnes, and T. Basar, "A Game-Based Self-Organizing Uplink Tree for VoIP Services in IEEE 802.16j Networks," in *Proceedings of the ICC 2009 - 2009 IEEE International Conference on Communications*, pp. 1–5, Dresden, Germany, June 2009.

[37] T. Rahwan and N. R. Jennings, "Distributing Coalitional Value Calculations Among Cooperating Agents," in *Proceedings of the 25th National Conference on AI (AAAI)*, pp. 152–157, Pittsburgh, Pa, USA, 2005.

[38] Y. Jiang and J. Jiang, "Contextual resource negotiation-based task allocation and load balancing in complex software systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 20, no. 5, pp. 641–653, 2009.

[39] V. Krishnamurthy and H. Vincent Poor, "Social learning and Bayesian games in multiagent signal processing: How do local and global decision makers interact?" *IEEE Signal Processing Magazine*, vol. 30, no. 3, pp. 43–57, 2013.

[40] M. N. Mejri, J. Ben-Othman, and M. Hamdi, "Survey on VANET security challenges and possible cryptographic solutions," *Vehicular Communications*, vol. 1, no. 2, pp. 53–66, 2014.

[41] F. Li and Y. Wang, "Routing in vehicular ad hoc networks: a survey," *IEEE Vehicular Technology Magazine*, vol. 2, no. 2, pp. 12–22, 2007.

[42] H. Füßler, J. Widmer, M. Käsemann, M. Mauve, and H. Hartenstein, "Contention-based forwarding for mobile ad hoc networks," *Ad Hoc Networks*, vol. 1, no. 4, pp. 351–369, 2003.

[43] S. Vodopivec, J. Bešter, and A. Kos, "A survey on clustering algorithms for vehicular ad-hoc networks," in *Proceedings of the 2012 35th International Conference on Telecommunications and Signal Processing, TSP 2012*, pp. 52–56, cze, July 2012.

[44] Q. Chen, F. Schmidt-Eisenlohr, D. Jiang, M. Torrent-Moreno, L. Delgrossi, and H. Hartenstein, "Overhaul of IEEE 802.11 modeling and simulation in NS-2," in *Proceedings of the 10th ACM Symposium on Modeling, Analysis, and Simulation of Wireless and Mobile Systems (MSWiM '07)*, pp. 159–168, October 2007.

[45] S. Yang, J. Li, Z. Liu, and Q. Yuan, "Relaying Message and Motivating Collaboration for VANET Data Service," in *Proceedings of the 14th IEEE International Conference on Services Computing, SCC 2017*, pp. 52–59, usa, June 2017.

Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 3698198, 10 pages
https://doi.org/10.1155/2018/3698198

Research Article

Personalized POI Recommendation Based on Subway Network Features and Users' Historical Behaviors

Danfeng Yan, Xuan Zhao, and Zhengkai Guo

State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China

Correspondence should be addressed to Zhengkai Guo; zhengk_guo@bupt.edu.cn

Received 16 April 2018; Accepted 14 June 2018; Published 9 July 2018

Academic Editor: Kok-Seng Wong

Current recommender systems often take fusion factors into consideration to realize personalize point-of-interest (POI) recommendation. Historical behavior records and location factors are two kinds of significant features in most of recommendation scenarios. However, existing approaches usually use the Euclidean distance directly without considering the traffic factors. Moreover, the timing characteristics of users' historical behaviors are not fully utilized. In this paper, we took the restaurant recommendation as an example and proposed a personalized POI recommender system integrating the user profile, restaurant characteristics, users' historical behavior features, and subway network features. Specifically, the subway network features such as the number of passing stations, waiting time, and transfer times are extracted and a recurrent neural network model is employed to model user behaviors. Experiments were conducted on a real-world dataset and results show that the proposed method significantly outperforms the baselines on two metrics.

1. Introduction

Recommender system has always been a hot topic and a lot of approaches have been proposed up to now. Nowadays, there are varieties of recommender systems in people's daily life, providing news, music, video, goods, or other kinds of items to the users. For example, Netflix is a famous online video rental provider, which recommends videos to its users based on historical rating data, and online retailers like Amazon and Alibaba provide products recommendation services on their websites and guide the customers' consumption actions. In recent years, with the development of mobile Internet, people get to use mobile APPs to find points of interest such as restaurants, malls, and view spots. In this situation, a good POI recommender system could show the most suitable candidate targets to users, thus helping to save users' time for picking and attain higher satisfaction.

Current recommender systems often use fusion factors, like users' profile, historical behaviors, and attributes of items, to calculate the relevance between users and items. Different from recommender systems for items like news or music, the location and other geographical factors are more important in personalized POI recommendation. This is because the distance between the points of interest and the user mainly determines the travel time, and people usually have more activities like eating, shopping, or watching movies in the region nearby. Some previous work has focused on applying location features in recommender systems: Natarajan and Moh [1] added a unique new feature of location preference to their news recommender system. Liu et al. [2] proposed a framework to recommend potential customers to vendors in location-based social network. Lee et al. [3] proposed a recommender system integrating location context, personal context, environment context, and user preference. However, these approaches only consider the geographical distance provided by location services such as GPS and use the location features from a single dimension. In fact, location-based recommendation problem is more complicated and more factors should be taken into consideration. In the real world, people cannot reach their destination in a straight line in most cases; thus the distance calculation should be based on road network information. In addition, people usually have to take the transport when they travel at a distance, and public transports such as bus or subway all have their own

lines and running schedules. In this context, factors such as station location, waiting time, and transfer times have varying degrees of impact on users' selection and should be taken into account in the POI recommender system. Compared with the bus system, the subway network is simpler and easier to analyze. At the same time, the subway is not susceptible to traffic jams, and the estimation of the time cost is more accurate. Therefore, our work focused on extracting the subway network features and using them to enhance the effect of POI recommendation.

Most recommender systems learn the users' preferences from their historical behaviors. For example, movie recommendations often predict users' preferences for new movies based on their rating scores on watched ones; the news recommender systems usually predict the click-through rate of news by analyzing users' actions like clicking, commenting, and sharing. In this context, how to represent and make use of the user preferences features has become a key issue. The user-based collaborative filtering algorithm represents user preferences in the form of vectors and uses them to calculate the similarity between users. The dimension of the vector is usually equal to the number of recommended items. The main disadvantage of this method is that the vector is very sparse in most cases and the similarity measure may not work. Matrix Factorization algorithms such as Probabilistic Matrix Factorization (PMF)[4], Nonnegative Matrix Factorization (NMF)[5] and Singular Value Decomposition (SVD) overcome this problem by using latent factors to represent user preferences. These methods consider the different aspects of user preferences equally but do not reflect the changes in users' interests. In fact, users' interests often change over time. In order to model the variation trend of user preferences over time, we trained a recurrent neural network using users' historical behaviors data and use the state of the units in the hidden layer to represent the user preferences. In this way, we took the user preferences as input features and added them into the ranking model of our POI recommender system.

In summary, the main contributions of our work include the following:

(1) We proposed a POI recommender system based on merged features. We took restaurant recommendation as the study case and managed to merge user profile, users' historical behavior, restaurant information, and subway network features.

(2) We represented the variation trend of user preferences over time as a fixed-length vector by employing a recurrent neural network.

(3) We used the Wide & Deep [6] model to predict the scores given to the restaurants by users.

(4) We conduct experiments on a real-world dataset to evaluate the effectiveness of our method. Experimental results show that our method outperforms 4 baseline methods in terms of root mean squared error (RMSE) and Mean Absolute Error (MAE).

The rest of this paper is organized as follows. Section 2 reviews the related works. Section 3 presents the main approach of the proposed restaurant recommender system. Section 4 introduces the conducted experiments and analyzes the results. Finally, Section 5 concludes the paper.

2. Related Work

In this section, we mainly review the related works from three aspects, including recommendation approaches, methods on processing users' historical behaviors data, and ranking models commonly used in recommender systems.

2.1. Recommendation Approaches. There are varieties of recommendation strategies, including collaborative filtering algorithm, content based recommendation, association rule based recommendation, and knowledge based recommendation. Collaborative filtering algorithm is one of the most commonly used methods in recommender systems. The main idea of collaborative filtering algorithms is to predict the unknown rating score based on the known ones. Collaborative filtering algorithm can be further divided into user-based collaborative filtering and item-based collaborative filtering [7]. The user-based collaborative filtering algorithm supposes that you may also like the things liked by the people who have the same preference as you. Therefore, the first step of the algorithm is to find users' nearest neighbors according to the distance metrics such as Pearson correlation coefficient and cosine similarity. The user-based collaborative filtering algorithm has a performance bottleneck that the complexity of finding the nearest neighbors increases substantially as the number of users growing. In this context, the item-based method turns to compute the similarity between different items, which reduces the computation cost and can be done offline. However, both of these two algorithms have the disadvantage that the user-item matrix could be so sparse that the distance metric may not work very well. To overcome this issue, Matrix Factorization algorithms factorize the user-item matrix and reduce the dimension of the parameters. The content based recommendation works based on users' historical behaviors, which is widely employed in information retrieval. Compared with the collaborative filtering methods, the content based recommendation models the attributes of the item and the results are easier to explain. The association rule based method makes the recommendation by mining the relevance between different items and has been successfully applied in retailing. The knowledge based recommendation utilizes the functional knowledge and makes the inference. Since the domain knowledge is hard to obtain, the knowledge based recommendation has greater limitations compared with other strategies.

Specifically, Wang et al. [8] unified user-based and item-based collaborative filtering approaches by similarity fusion and made the model more robust to data sparsity. Yao et al. [9] used a probabilistic generative model to take both the rating data and the semantic content data into consideration in web service recommendation. Tewari et al. [10] proposed a book recommendation system based on combine features of content based filtering, collaborative filtering, and association rule mining. Zhu et al. [11] take into account the importance of location and the intragroup influence in POI

group recommendation and employ distance prefiltering and distance ranking adjustment to improve recommendation satisfaction.

Quality-of-service (QoS) of restaurants is an important factor to be considered when recommending restaurants to users. Wang and Zheng et al. [12] propose a reputation measurement approach and a malicious feedback ration prevention scheme of web services. Wang and Ma et al. [13] present an integrated QoS prediction approach, which unifies the modeling of multidimensional QoS data via a multilinear-algebra based concept of tensor, for web service recommendations. Wang and Zhao et al. [14] propose a service recommendation approach based on collaborative filtering and make QoS prediction using user mobility.

In this paper, we recommend the restaurants to the user by predicting the user's rating score on each restaurant. In this way, the recommendation task is considered as a regression problem. Multidimensional features, such as user profile, restaurant attributes, users' historical behaviors, and subway network features, are selected as the input of the model. Compared with other methods, this approach is more scalable and has higher accuracy. The proposed architecture will be illustrated in detail in Section 3.

2.2. Processing History Data. Users' historical behaviors data are very significant in recommender systems. This kind of data, such as the list of videos that have been watched, purchase records, and browser history, reflects the users' preference to a certain extent and is of great significance for predicting users' future actions [15–18]. A simple way to make use of this kind of data is to concatenate users' historical behaviors in series and represent these records with a vector. This method is widely used in recommender systems of video websites, online shopping, and so on. However, it has some disadvantages and may have a bad impact on the prediction model. To be specific, users usually have different number of historical behavior records, as a result of this the length of the vector is not fixed and cannot be input into the prediction model directly. Some techniques are employed to solve this problem, such as truncation and completion. As shown in Figure 1, for the user who has few historical behavior records, the completion processing fills the vacancy position of the vector by zero. Correspondingly, for the user who has more historical actions, the truncation processing abandons the earlier records and just keeps the newer data. By these means the historical behaviors feature vector of different users is fixed-length and represented in a unified form. The truncation and completion processing generate the fixed-length vector; nevertheless, these operations bring the problems such as information loss and introduction of noise at the same time. Moreover, the dimension of the vector may be so high that influences the efficiency of model training.

Another common method of processing users' historical behavior records is employing the embedding technique. For example, the video recommender system of YouTube averages the embeddings of the sparse video IDs and the embeddings of the historical search tokens and then inputs the watch vector and the search vector to the hidden layers of the neural network model [19]. Zheng et al. [20] proposed

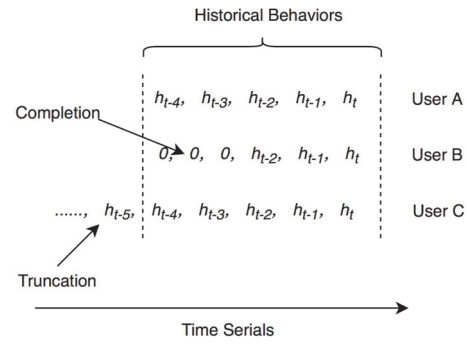


Figure 1: Example of historical behavior records processing method.

a model named DeepCoNN, which utilizes two parallel convolutional neural networks to model users' behaviors and the properties of items from the reviews and designs a shared layer to couple the two networks together. As illustrated in Figure 2, this method learns the latent factors of users' preferences and the properties of items in a similar way as the Matrix Factorization algorithms. Actually, the Matrix Factorization algorithms can also be considered as a sort of embedding technique, where the matrices after decomposition represent the embeddings of users and items. These methods compress the variable sized user behavior data into a fixed-length vector but do not make the most of the timing characteristics of the data. The user behavior sequence reflects the user behavior patterns and the variation trend of user preferences over time; thus, the sequence order should be taken into consideration when modeling. The recurrent neural network is suitable for modeling the time series data and has been successfully applied in processing history records in recommender systems. Specifically, Liu et al. [21] proposed an extension model of recurrent neural network which outputs the embeddings of the sequential heterogeneous attributes for item recommendation. Dai et al. [22] put forward a point of view that user features and item features may drift, evolve, and coevolve over time due to the interaction, and they employed a recurrent neural network to learn the representation of influences automatically and capture the coevolving nature of both user feature and item feature. In the news recommendation method introduced in [23], a recurrent neural network is used to generate the representation of the user with browsing histories as the input.

In the proposed recommender system, we use a single layer recurrent neural network to learn the behavior pattern and the preference of the user. In the training phase, users' historical behavior data are organized in sequence and fed into the model with a sliding window. After the model is well trained, we input the historical record of the user into the model and use the final states of the hidden layer as the vector representation of the user behavior feature.

2.3. Models. Varieties of machine learning models have been employed for ranking in recommender systems. Logistic Regression, which is one of the most famous and commonly

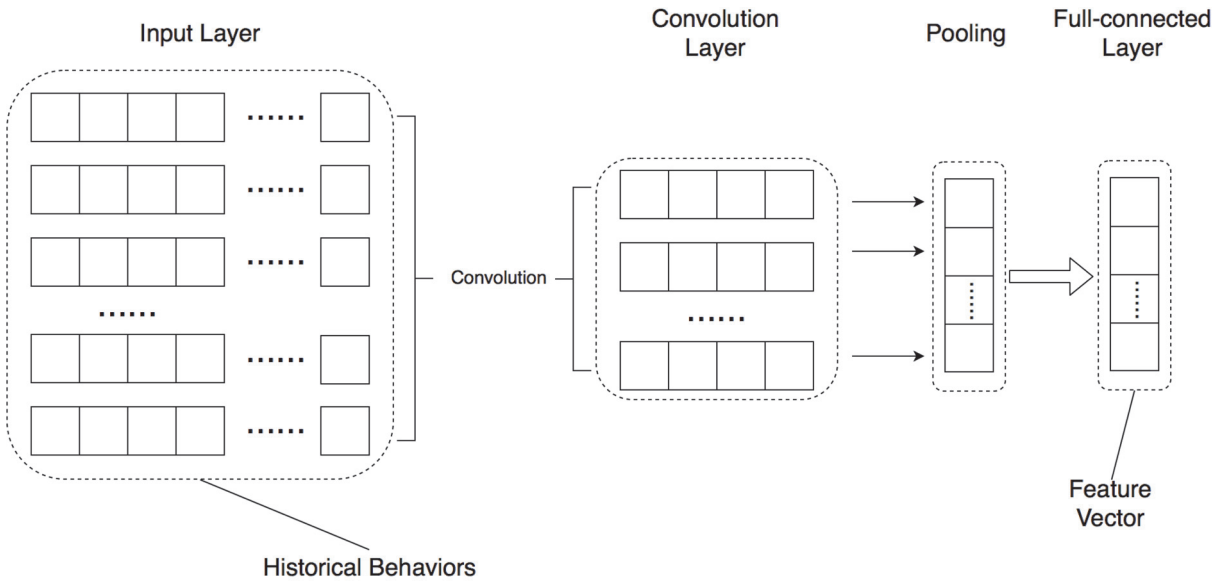


Figure 2: Encode historical behaviors with convolutional neural network.

used algorithm, has been proved to be effective in application scenarios such as online advertising, recommender systems, and web search engine. The main advantages of Logistic Regression algorithm are easy implementation, high computing efficiency, easy to parallelize, and suitable for processing high dimensional and sparse features. As a generalized linear model, Logistic Regression algorithm weights and sums the feature values and maps the result to the interval (0, 1) with the logistic function. Therefore, the algorithm requires much manual feature engineering to filter and validate the combined features, which is very complex and lacks efficiency. In this context, Rendle proposed an advanced model named Factorization Machines (FMs) [24]. Factorization Machines add the explicit combined features to the equation and use factorized parameters to estimate the interactions between variables. Field-aware Factorization Machines (FFMs) [25] are variant of FMs, and the model is based on the idea that features can be grouped into fields for most datasets. Every variable in FMs has only one parameter vector representing the latent effect with other variables, but in FFMs, each variable has several latent vectors to interact with variables belonging to different fields. Neural networks have a strong nonlinear approximation ability and have higher prediction accuracy in most cases. With the development of deep learning, there have been several approaches to employ deep neural network to process the ranking task in recommender systems. Among these approaches, the Wide & Deep model proposed in [6] joins the generalized linear models and the deep neural networks and trains the two components synchronously. In this model, the generalized linear model (wide component) is designed to learn the memorization of feature interactions, and the deep neural network (deep component) can learn the dense embeddings and generalize to feature combinations. Since the Wide & Deep model combines the benefits of memorization and generalization for recommender systems, we select this model to make the prediction in this paper.

3. System Design

In this section, we introduce the system design of the proposed restaurant recommender system from four main aspects. The first part presents the overall architecture of the recommender system and the major functions of all the components. Then, the extraction of subway network features is introduced in the second part. The third part illustrates the user behaviors modeling method. Finally, the ranking model of the recommender system is introduced in the fourth part of this section.

3.1. Architecture. The overall architecture of the proposed restaurant recommender system is illustrated in Figure 3.

As the figure shows, the recommender system consists of two main components, implementing the function of ranking model generation and restaurant recommendation, respectively. The ranking model generation component can further be divided into three parts: subway network features extraction, user behaviors modeling, and ranking model training. Concretely, the subway network feature extraction part firstly attaches the user and candidate restaurants to the nearest subway stations and then calculate the features such as distance from the station, number of stations passing through, and transfer times based on the subway network structure. The user behaviors modeling part learns user preferences and behavior patterns through the historical visiting records and generates users' behavior feature vectors. Finally, these features are organized as the input and employed to train the ranking model of the recommender system. As for the restaurant recommendation component, the system firstly concatenates users' basic information, candidate restaurants

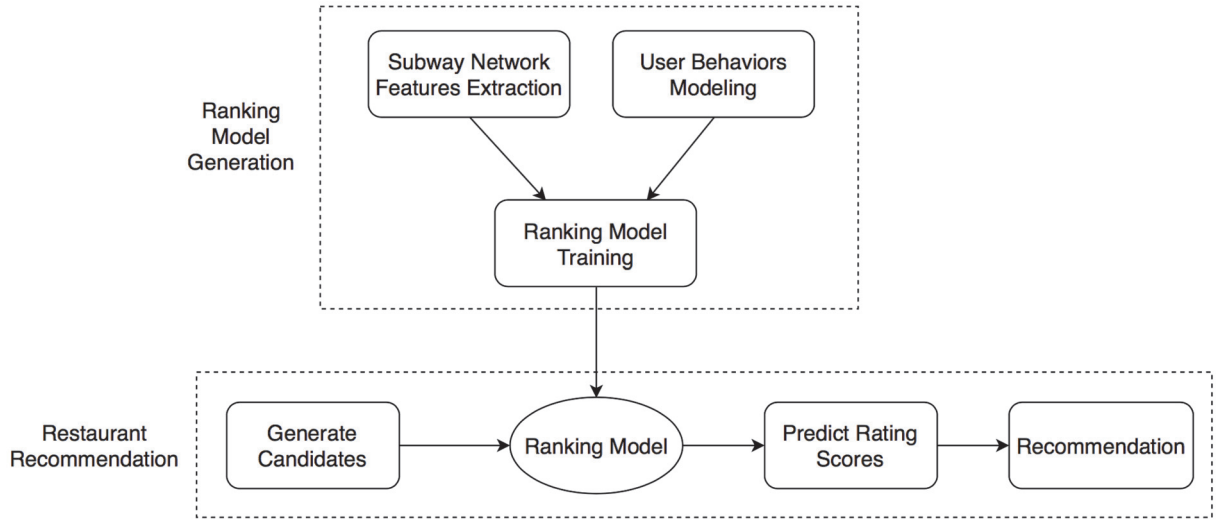


Figure 3: The overall architecture of the proposed restaurant recommender system.

characteristics, subway network features, and user behavior features together to obtain the input feature vectors. Then, the well trained ranking model is used to predict users' rating scores on each candidate restaurant. Finally, the system will make the recommendation based on the prediction results.

The subway network features extraction, user behaviors modeling method, and ranking model design are the focus of our work. These three aspects will be introduced in detail in the following subsections.

3.2. Subway Network Feature Extraction. Buses and subways are two of the most important ways of public transportation. In this paper, we select the subway network as the study object because it has some advantages comparing with buses. Firstly, the subway network is more popular with users when they travel by public transport. Secondly, the subway has a more accurate timetable and is not susceptible to traffic jams, which makes it more accurate to estimate the time cost. Thirdly, users prefer to take the subway when the destination is far away. For above-mentioned reasons, we focus on analyzing the subway network and extract a series of features. Specifically, we studied the subway network of Guangzhou City, which includes 13 lines and 205 stations up to now.

The main steps of subway network features extraction are as follows:

(1) The first step of subway network features extraction is finding the nearest stations to the user and the candidate restaurant. As a nearest neighbor problem, one possible solution is to calculate the Euclidean distance between the object and every subway stations. However, this method is so inefficient that is unable to meet the needs of real-time recommendation. In our work, we constructed a KD-Tree [26] based on the latitude and longitude of the subway stations. By this means, the city map is partitioned into areas and the lookup complexity is reduced from $O(n)$ to $O(\sqrt{n})$, where n represents the number of subway stations.

(2) When the first step is finished, the nearest stations to the user and candidate restaurant are set as the origin and the destination respectively and then calculate the number of stations passing through. This is a typical shortest path problem and we employ the Floyd algorithm [27] to calculate the shortest path between any two stations.

(3) The number of stations passing through reflects the distance between two subway stations. However, the distance between adjacent stations is often different, and it is inaccuracy and insufficient to use only this feature. Moreover, the travel time is the most intuitive feeling for users. Thus, we additionally calculate the travel time after getting the shortest path. The travel time is calculated according to the subway schedule, which can be obtained through Internet.

These subway network features improve the location-based recommendation approach by taking into consideration the transportation conditions. Moreover, measuring the distance based on the number of stations passing through and the travel time is more appropriate than calculating the Euclidean distance on the plane directly.

3.3. Modeling User Behaviors. As discussed in Section 2, we use a three layers RNN to learn the behavior pattern and the preference of users. The model structure and the main process of the method are shown in Figure 4.

The modeling process consists of two stages: model training and feature vector calculation. Firstly, we constructed the feature vector of every restaurant. There are four parts in the feature vector: restaurant ID, restaurant type, average consumption, and average rating scores. As categorical features, the ID and type of the restaurant are transformed to vector representation by embedding method. Moreover,

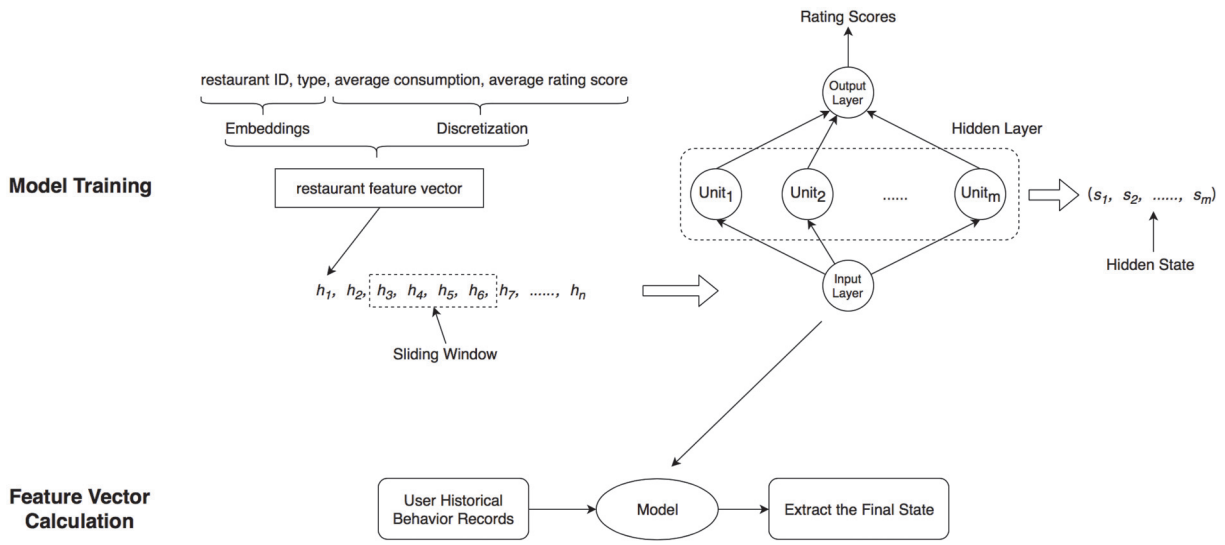


Figure 4: The RNN based user behaviors modeling method.

the other two continuous variables are transformed to the vector form by using equal-depth discretization. The neural network model consists of three layers, including input layer, output layer, and hidden layer. The hidden layer contains m units where m determines the dimension of users' preference feature vector. In the training phase, for each user, the feature vectors of the visited restaurants are arranged in order and fed into the model with a sliding window. User's rating score on the next restaurant behind the sliding window is set as the prediction target and the optimization target is square loss. According to the principle of the recurrent neural network, the hidden state is updated based on both the current input and the hidden state of the last time-step. In this context, we can conclude that the final hidden state contains the whole context information of the input users' behaviors. Furthermore, the final hidden state reflects users' preferences and behavior patterns, and it is reasonable to be used as the behavior feature vector of users.

After the recurrent neural network is well trained, it is saved and employed to generate the behavior feature vector of the user. Assuming that the sliding window length is T, we fed into RNN model the user's last T visited restaurants and extract the final state of hidden layer. We regard the final state of hidden layer as user's behavior feature vector. In this approach, the RNN model handles the varieties in user's behaviors over time.

There are mainly two key hyperparameters in this user behaviors modeling method: the units number in the hidden layer and the size of the sliding window. Both of these two hyperparameters have a great impact on the modeling effect. Specifically, the feature vector can hardly express users' preferences if there are few units in the hidden layer. By contrast, too many units generate high dimensional feature vector, which is usually very sparse and fails to compress users' historical behavior data. For the size of the sliding window or the number of time-steps, the model needs

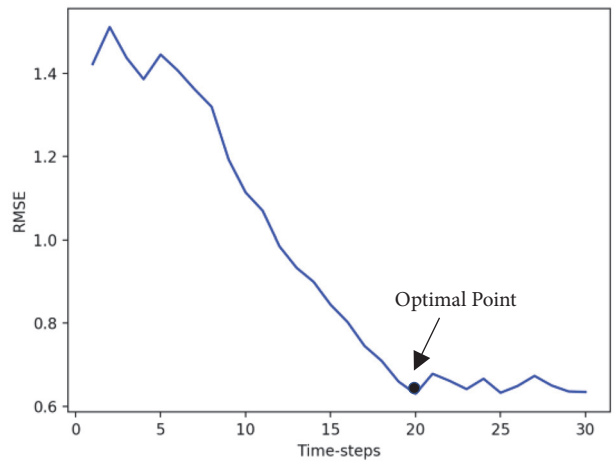


Figure 5: Prediction deviation with various time-steps.

enough records to learn user characteristics. Theoretically, more historical records contain more information and can improve the modeling effect to a certain extent. However, for ordinary RNN model, later inputs usually have more weight than previous ones. Therefore, the proposed model pays more attention to users' recent behaviors and the long-term records only have a little influence. Actually, we tested the modeling effect with various sliding window sizes and found that the prediction deviation is approximately asymptotically convergent. The experiment result is shown in Figure 5; it shows that the sliding window size value has an "optimal point". When the size value is in the left interval of the point, the modeling effect has a significant improvement with historical records growing. Correspondingly, the modeling effect is tending to be stable for more input records exceeding the point. As to the issue of how to determine the optimal point, we have not come up with effective methods besides

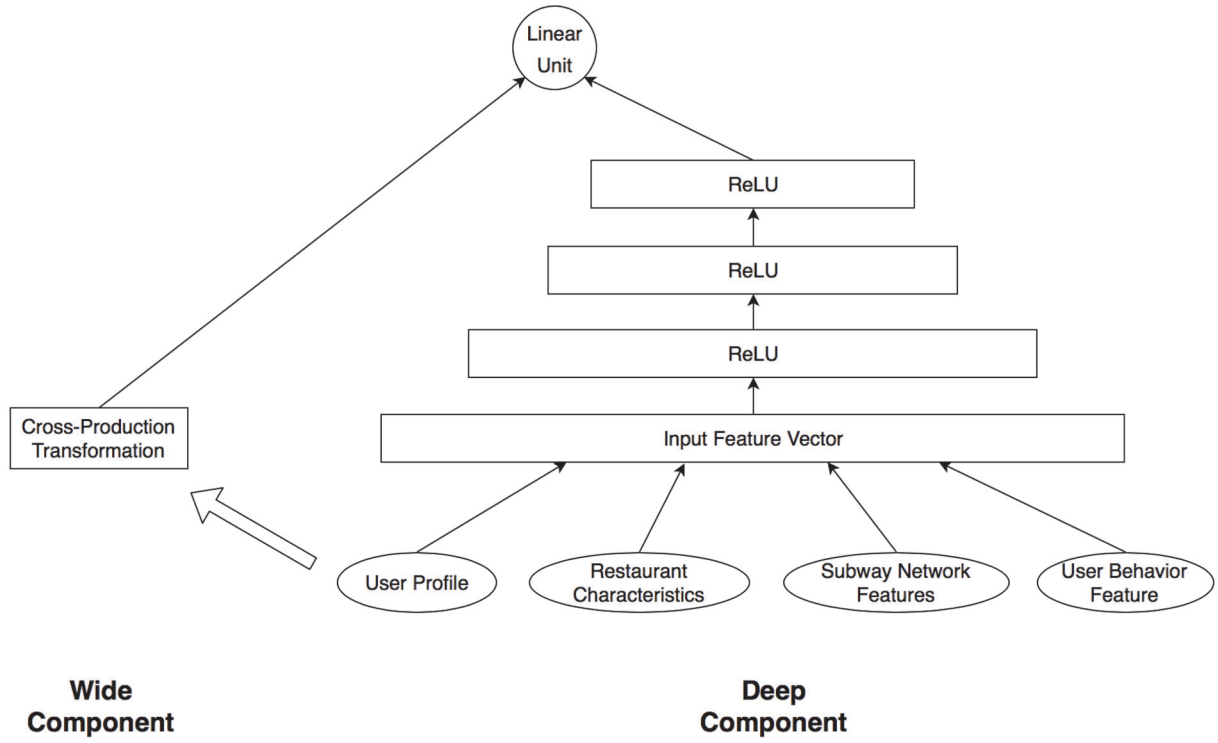


Figure 6: The Wide & Deep model structure designed for the ranking task of restaurant recommender system.

linear detection at present. We will focus on this question in our future work.

3.4. Ranking Model. The ranking model is always the core component of recommender systems. There are several training strategies for the task of learning to rank, including Pointwise, Pairwise, and Listwise. In this paper, we take the Pointwise method and solve the ranking task as a regression problem. Four types of features, including users' basic information, candidate restaurants characteristics, subway network features, and user behavior features, are concatenated as the input of the model. Users' rating scores on restaurants are set as the learning target and square error is set as the loss function. As mentioned in Section 2.3, we employed the Wide & Deep model to predict the scores, and the model structure is shown in Figure 6.

The model consists of two parts: the wide component made up of a generalized linear model and the deep component of a deep neural network. According to [6], the wide component is designed to learn the memorization of feature interactions and the deep component achieves the generalization. In our model, the cross-production transformation of binary features and the user historical behavior features are fed into the wide part. For the deep part of the model, the categorical features such as the user age and the type of the restaurant are transformed into embeddings firstly. Then all the embeddings and dense features are concatenated together and fed into the network which contains three ReLU layers. Finally, there is a linear unit output of the regression prediction result. The prediction result is shown as follows:

$$\text{prediction} = W_{wide}^{T} X_{wide} + W_{deep}^{T} a^{(l)} + \text{b} \tag{1}$$

where X_{wide} is the features fed into the wide part, $a^{(l)}$ is the activation of the final ReLU layer, W_{wide} and W_{deep} are the vectors of wide component weights and weights applied on $a^{(l)}$, respectively, and b is the bias term.

The Wide & Deep model is trained jointly. The gradients of the loss function is backpropagated from the output to both the wide and deep component of the model at the same time. Specifically, the wide part of the model is optimized using Adam method.

4. Experiments

In this section, we conduct extensive experiments with a real-word dataset. First, we present the dataset and the evaluation metric used in our experiments. Then, we describe the baseline algorithms selected for comparisons. Then, we introduce the parameter settings. Finally, we analyze the results.

4.1. Dataset and Evaluation Metric. In our experiments, we use a dataset collected from a real APP named Green Travel. It contains a large number of users, restaurants, and ratings for restaurant. We choose 1M ratings from 1525 users to 2657 restaurants. Each rating is an integer between 1 and 5.

Table 1: MAE and RMSE values of baselines and our model.

Model	MAE	RMSE
PMF	0.6142	0.8176
NMF	0.5863	0.7207
SVD	0.8218	1.0344
SVD++	0.8221	0.9352
W&D	0.5532	0.6738
W&D-his	0.5205	0.6419
W&D-sub	0.5398	0.6638
W&D-his-sub	**0.5148**	**0.6317**

In this paper, we use the most popular metrics, Root Mean Square Error (RMSE), and Mean Average Precision (MAE):

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n} (r_i - \hat{r}_i)^2}{n}} \quad (2)$$

$$MAE = \frac{\sum_{i=1}^{n} |r_i - \hat{r}_i|}{n} \quad (3)$$

where r_i denotes the ith observed rating, $\hat{r}_i$ denotes the ith predicted rating, and n is the number of ratings in test set.

4.2. Baselines. In order to evaluate the performance of our model, we compare it with the following state-of-the-art models.

(i) **NMF**: **N**onnegative **M**atrix **F**actorization. It only uses the rating matrix as the input.

(ii) **PMF**: **P**robabilistic **M**atrix **F**actorization. It models latent factors of users and items by Gaussian distributions.

(iii) **SVD**: **S**ingular **V**alue **D**ecomposition. It uses rating matrix as input and estimates two low-rank matrices. It also uses bias to reduce the error.

(iv) **SVD++**. It extends **S**ingular **V**alue **D**ecomposition by considering implicit feedback information for latent factor modeling [28].

4.3. Parameter Settings. We divided the dataset into three subsets: 80% for training set, 10% for validation set, and 10% for test set. All the hyperparameters of the baselines and our model are tuned with the validation set.

We used grid search to find the best value for the number of latent factors and regularization parameters. For NMF, PMF, SVD, and SVD++, we set the number of latent factors k = 64, 64, 48, 32, and regularization parameters r = 0.1, 0.1, 0.15, 0.2.

For our model, we use 1-layer RNN to model user's historical behaviors. Specifically, we used CuDNNGRU in order to benefit from its performance. The dimension of the user's latent factor is set to 128. We tried different number of user's historical behaviors and find 20 is the best. The number of hidden layers of the deep component in the model is 3.

All the neural matrix parameters in hidden layers and RNN layers are initialized from a uniform distribution between [-0.1, 0.1]. The network was trained via Adam optimizer with batch size 64 and learning rate 0.001.

Our models are implemented in Tensorflow, a well-known Python library for deep learning. Our models are trained and tested on an NVIDIA GTX1080 TI GPU.

4.4. Results and Discussion. We tried different feature combinations in our model:

(i) **W&D**: Wide & Deep model only use ratings, user profile, and restaurant information.

(ii) **W&D-his**: Wide & Deep model use ratings, user profile, restaurant information, and user's historical behaviors.

(iii) **W&D-sub**: Wide & Deep model use ratings, user profile, restaurant information, and subway network feature.

(iv) **W&D-his-sub**: Wide & Deep model use ratings, user profile, restaurant information, user's historical behaviors, and subway network feature.

Table 1 shows the best MAE and RMSE of PMF, NMF, SVD, SVD++, and our model with different features. We can observe from Table 1 that our models achieve better performance than all the baselines. It significantly outperforms the four other baselines in MAE and RMSE, which demonstrates the effectiveness of our models.

Moreover, W&D-his and W&D-sub outperform W&D, which shows that user historical behaviors and subway network features are important for restaurant recommendation. Furthermore, we can see that W&D-his obtains lower MAE and RMSE than W&D-sub, from Table 1, which validates user's historical behaviors, and is more effective than subway network feature. Finally, we add user's historical behaviors and subway network feature into our model, which obtains the best performance.

5. Conclusions

In this paper, we take the restaurant recommendation as an example and propose a POI recommender system based on fusion features. Subway network features, including the number of passing stations, waiting time, and transfer times, are extracted to measure the distance between the user and the restaurant in a more reasonable way than calculating the

Euclidean distance directly. We employ a recurrent neural network with one hidden layer to learn the embeddings of users' historical behaviors. In this way, the time characteristics of behaviors data are captured and user preferences can be represented with a fixed-length vector. Experiment results show that both the subway network features and user historical behavior features are helpful for restaurant recommendation and can improve the prediction accuracy of the ranking model.

Data Availability

The authors use a dataset collected from a real APP named Green Travel. It contains a large number of users, restaurants, and ratings for restaurant.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This paper is supported by "National 863 Project (no. 2015AA050204)", "State Grid Science and Technology Project (no. 520626170011)", and "Natural Science Foundation of China (no. 61532006 and no. 61320106006)".

References

[1] S. Natarajan and M. Moh, "Recommending news based on hybrid user profile, popularity, trends, and location," in *Proceedings of the 2016 International Conference on Collaboration Technologies and Systems, CTS 2016*, pp. 204–211, usa, November 2016.

[2] Y. Liu, P. Zhao, V. S. Sheng et al., "RPCV: Recommend Potential Customers to Vendors in Location-Based Social Network," in *Web-Age Information Management*, vol. 9098 of *Lecture Notes in Computer Science*, pp. 272–284, Springer International Publishing, Cham, 2015.

[3] B. Lee, H. Kim, J. Jung, and G. Jo, "Location-Based Service with Context Data for a Restaurant Recommendation," in *Database and Expert Systems Applications*, vol. 4080 of *Lecture Notes in Computer Science*, pp. 430–438, Springer, Berlin, Heidelberg, 2006.

[4] A. Mnih and R. Salakhutdinov R, "Probabilistic matrix factorization," *Advances in Neural Information Processing Systems*, pp. 1257–1264, 2008.

[5] D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," *Advances in neural information processing systems*, pp. 556–562, 2001.

[6] H. Cheng T, L. Koc, and J. Harmsen, *Wide & Deep Learning for Recommender Systems*, 2016.

[7] G. Linden, B. Smith, and J. York, "Amazon.com recommendations: item-to-item collaborative filtering," *IEEE Internet Computing*, vol. 7, no. 1, pp. 76–80, 2003.

[8] J. Wang, A. P. De Vries, and M. J. T. Reinders, "Unifying user-based and item-based collaborative filtering approaches by similarity fusion," in *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 501–508, ACM, August 2006.

[9] L. Yao, Q. Z. Sheng, A. H. H. Ngu, J. Yu, and A. Segev, "Unified collaborative and content-based web service recommendation," *IEEE Transactions on Services Computing*, vol. 8, no. 3, pp. 453–466, 2015.

[10] A. S. Tewari, A. Kumar, and A. G. Barman, "Book recommendation system based on combine features of content based filtering, collaborative filtering and association rule mining," in *Proceedings of the 2014 4th IEEE International Advance Computing Conference, IACC 2014*, pp. 500–503, ind, February 2014.

[11] Q. Zhu, S. Wang, B. Cheng, Q. Sun, F. Yang, and R. N. Chang, "Context-Aware Group Recommendation for Point-of-Interests," *IEEE Access*, vol. 6, pp. 12129–12144, 2018.

[12] S. Wang, Z. Zheng, Z. Wu, M. R. Lyu, and F. Yang, "Reputation measurement and malicious feedback rating prevention in web service recommendation systems," *IEEE Transactions on Services Computing*, vol. 8, no. 5, pp. 755–767, 2015.

[13] S. Wang, Y. Ma, B. Cheng, F. Yang, and R. Chang, "Multi-dimensional QoS prediction for service recommendations," *IEEE Transaction on Services Computing*, 2016.

[14] S. Wang, Y. Zhao, L. Huang, J. Xu, and C.-H. Hsu, "QoS prediction for service recommendations in mobile edge computing," *Journal of Parallel and Distributed Computing*, 2017.

[15] J. Davidson, B. Liebald, J. Liu, P. Nandy, and T. Van Vleet, "The YouTube video recommendation system," in *Proceedings of the 4th ACM Recommender Systems Conference (RecSys '10)*, pp. 293–296, Barcelona, Spain, September 2010.

[16] X. Zhao, H. Luan, J. Cai et al., "Personalized video recommendation based on viewing history with the study on YouTube," in *Proceedings of the International Conference on Internet Multimedia Computing and Service*, pp. 161–165, ACM, 2012.

[17] S. Seko, M. Motegi, T. Yagi, and S. Muto, "Video content recommendation for group based on viewing history and viewer preference," in *Proceedings of the 2011 IEEE International Conference on Consumer Electronics, ICCE 2011*, pp. 351-352, usa, January 2011.

[18] A. V. Bodapati, "Recommendation systems with purchase data," *Journal of Marketing Research*, vol. 45, no. 1, pp. 77–93, 2008.

[19] P. Covington, J. Adams, and E. Sargin, "Deep neural networks for youtube recommendations," in *Proceedings of the 10th ACM Conference on Recommender Systems, RecSys 2016*, pp. 191–198, September 2016.

[20] L. Zheng, V. Noroozi, and P. S. Yu, "Joint Deep Modeling of Users and Items Using Reviews for Recommendation," in *Proceedings of the the Tenth ACM International Conference*, pp. 425–434, Cambridge, United Kingdom, Feburary 2017.

[21] K. Liu, X. Shi, and P. Natarajan, "Sequential Heterogeneous Attribute Embedding for Item Recommendation," in *Proceedings of the 2017 IEEE International Conference on Data Mining Workshops (ICDMW)*, pp. 773–780, New Orleans, LA, November 2017.

[22] H. Dai, Y. Wang, R. Trivedi, and L. Song, "Recurrent Coevolutionary Latent Feature Processes for Continuous-Time Recommendation," in *Proceedings of the the 1st Workshop*, pp. 29–34, Boston, MA, USA, September 2016.

[23] S. Okura, Y. Tagami, S. Ono, and A. Tajima, "Embedding-based news recommendation for millions of users," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2017*, pp. 1933–1942, August 2017.

[24] S. Rendle, "Factorization machines," in *Proceedings of the 10th IEEE International Conference on Data Mining, ICDM 2010*, pp. 995–1000, Australia, December 2010.

[25] Y. Juan, Y. Zhuang, W.-S. Chin, and C.-J. Lin, "Field-aware factorization machines for CTR prediction," in *Proceedings of the 10th ACM Conference on Recommender Systems, RecSys 2016*, pp. 43–50, usa, September 2016.

[26] J. L. Bentley, "Multidimensional binary search trees used for associative searching," *Communications of the ACM*, vol. 18, no. 9, pp. 509–517, 1975.

[27] R. W. Floyd, "Algorithm 97: shortest path," *Communications of the ACM*, vol. 5, no. 6, p. 345, 1962.

[28] Y. Koren, "Factorization meets the neighborhood: a multi-faceted collaborative filtering model," in *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '08)*, pp. 426–434, New York, NY, USA, August 2008.

Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 4263831, 11 pages
https://doi.org/10.1155/2018/4263831

Research Article

Dynamic Pricing for Resource Consumption in Cloud Service

Bin Cao,[1] **Kai Wang**,[1] **Jinting Xu**,[1] **Chenyu Hou**,[1] **Jing Fan**,[1] **and Hangning Que**[2]

[1]*College of Computer Science & Technology, Zhejiang University of Technology, Hangzhou, China*
[2]*NetEase, Inc., Hangzhou, China*

Correspondence should be addressed to Jing Fan; fanjing@zjut.edu.cn

Received 24 February 2018; Accepted 19 April 2018; Published 24 May 2018

Academic Editor: Shangguang Wang

This paper studies dynamic pricing for cloud service where different resources are consumed by different users. The traditional cloud resource pricing models can be divided into two categories: on-demand service and reserved service. The former only takes the using time into account and is unfair for the users with long using time and little concurrency. The latter charges the same price to all the users and does not consider the resource consumption of users. Therefore, in this paper, we propose a flexible dynamic pricing model for cloud resources, which not only takes into account the occupying time and resource consumption of different users but also considers the maximal concurrency of resource consumption. As a result, on the one hand, this dynamic pricing model can help users save the cost of cloud resources. On the other hand, the profits of service providers are guaranteed. The key of the pricing model is how to efficiently calculate the maximal concurrency of resource consumption since the cost of providers is dynamically varied based on the maximal concurrency. To support this function in real time, we propose a data structure based on the classical B+ tree and the implementation for its corresponding basic operations like insertion, deletion, split, and query. Finally, the experiment results show that we can complete the dynamic pricing query on 10 million cloud resource usage records within 0.2 seconds on average.

1. Introduction

In recent years, the development of cloud computing is extremely rapid [1]. With the cloud computing, we can deploy different kinds of computing resources in the cloud, rather than a specified server [2]. In other words, there is no need to use hardware equipment, which is very convenient for the companies or organizations so they can focus on their core businesses rather than expending resources on computer infrastructure and maintenance [3–7]. Therefore, more and more users are looking for the cloud computing, which makes the cloud resource providers emerge. Due to the reason that the service deployed in the cloud may migrate between different resources, and in the meantime mobile users such as smart devices cause the dynamic change in resource consumption, hence, a dynamic pricing scheme is needed for both service providers and consumers.

The traditional cloud resource pricing models can generally be divided into two categories. The first one is on-demand service; i.e., the users should pay for the fee, which is based on the using time and the actual resource consumption. This pricing model is suitable for the short-term users. The second one is reserved service; i.e., each user pays a fixed fee for a month or year and could use the cloud resource during this period of time without limitation.

However, both of the two pricing models have disadvantages. For the first one, users are required to pay a huge fee if they use the cloud resource for a long time. But in this case, the concurrency for these users maybe little; e.g., only one computer is online, and compared with users who have multiple computers online simultaneously, it is unfair to charge users with little concurrency. As for the second one, it is unfair for the users who use the cloud service for little time monthly. They must pay for the same fee with the users who use the cloud resource for a long time. Therefore, it is meaningful to design a more reasonable pricing model for the cloud resources service.

In this paper, for cloud resource, we propose a flexible dynamic pricing model which takes into account occupying time, resource consumption, and maximal concurrency. In

our pricing model, the fee of cloud service for the user is mainly composed of three parts: the monthly rental, the fee of his maximal concurrency, and the fee of his using time and resource consumption. As we all know, it is inevitable that many users use the cloud resources at the same time, which may cause a great concurrency of resource consumption on the cloud servers. The greater number of concurrency is, the higher load is on the cloud servers, which results in the higher cost for the cloud resource providers. In other words, the cost of the cloud resource providers is mainly determined by the maximal concurrency of all the users. In fact, many service providers also price their service based on the maximal concurrency of resource consumption that the user needs, which also motivates our work.

However, the dynamic of service migration and user mobility causes a large number of records for the resource consumption, which makes it challenging to find the maximal concurrency of resource consumption in real time when we calculate the fee of the cloud servers for users in our dynamic pricing model.

Suppose that there are 100,000 users using the cloud resources with their usage records. In order to calculate the cost, the cloud resource provider needs to query their usage records and work out the maximal concurrency. However, it is difficult to find the maximal concurrency of resource consumption since it may occur at an arbitrary instant time. It would be extremely inefficient if we calculate the concurrency for each timestamp when data amount is large. Therefore, we should design a new efficient algorithm to find the maximal concurrency. As we know, this problem is similar to the aggregation query [8] since both of them calculate the overall information at a given time. However, different from the aggregation which has a time interval as a given condition, our problem needs to calculate the maximal concurrency without a time interval. Therefore, there is no existing algorithm that can solve this problem directly as far as we know. In order to solve the problem, we propose a data structure called B++-tree and corresponding operations including insertion, deletion, and split. Additionally, query processing algorithm based on B++-tree is also presented. Basically, we solve the efficiency issue by storing all users records into B++-tree and calculating the maximal concurrency of resource consumption by traversing all leaf nodes of B++-tree.

The contribution in this paper can be summarized as follows:

(1) We propose a more reasonable pricing model of cloud resource, which takes into account occupying time, resource consumption, and maximal concurrency.

(2) We propose a data structure called B++-tree and the operational algorithm based on B++-tree to calculate the maximal concurrency which is the fundamental of the cost for the cloud resource providers.

(3) We performed extensive experiments to test our algorithm. The experiment results show that we can complete the maximal concurrency query on 10 million data within 0.2 seconds.

The rest of this paper is organized as follows. Section 2 discusses the related work. Section 3 introduces the pricing models we proposed. Section 4 introduces the B++-tree and the implementation details for its corresponding operations. Section 5 presents the experiment. Section 6 is our conclusion.

2. Related Work

Our problem can be divided into two major subproblems: (1) how to design a reasonable pricing model and (2) how to find the maximal concurrency of resource consumption. Therefore, we investigate related work from these two parts, respectively.

For the first part, cloud computing is different from the classic distributed system. The pricing model of the cloud service should take the pricing fairness, evolving system dynamics, and cost of failures into account [9]. The existing pricing schemes in the cloud market can be summarized into three types: trading on-demand service, reserved service, and spot service, respectively [10]. Trading on-demand service means that the cost of a user is based on the time he used for cloud resources. But users pay a fixed fare for cloud resources in the reserved service. These two static pricing schemes are the main pricing models in the current cloud market [11, 12]. Different from them, the spot service is a dynamic pricing scheme where users' payments depend on the relation of their demand and the available cloud resource. Based on the main idea of the spot model, many kinds of dynamic pricing model have been proposed in recent literature, like auction mechanisms [13–16]. Zheng et al. [17] developed a predator-prey model which can simulate the interactions between demand and resource and compute the fare of cloud service. Zhang et al. [18] proposed a joint pricing and scheduling strategy and proved the worst-case competitive ratios of the pricing functions. However, none of above works consider the dynamic pricing for cloud service that we do since the algorithm proposed in this paper can efficiently report the total cost for the providers no matter how the service migrates or user behaves.

For the second part, as far as we know, there are few algorithms that can find the maximal concurrency of resource consumption to solve our problem directly. But we find a closed problem called temporal aggregations, in which there are some temporal aggregations operators such as count, sum, and average. These operators are similar to the method of calculating the maximal concurrency. There are also many algorithms to solve temporal aggregations problem. Kline and Snodgrass [19] proposed aggregation tree, which is a data structure, to support incremental computation for temporal aggregation. However, the aggregation tree is unbalanced whose time complexity is $O(n^2)$ for constructing a tree, where n is the number of intervals. Moon et al. [20] presented a balanced tree algorithm whose time complexity is $O(n \log n)$, where n is the number of intervals. Besides, Moon et al. [21] also proposed a bucket algorithm and parallelized it on a shared-nothing architecture. Yang and Widom [22] presented a data structure called SB-Tree which combines B-tree [23, 24] and segment tree [25]. SB-Tree can feedback a query in $O(n \log n)$ and an update in $O(n \log n)$, where n is the number of intervals. However, there is a difference between our problem and temporal aggregation. The time interval is

Table 1: User records.

User	Time interval	Concurrency
A	[5–10]	2
B	[10–20]	4
C	[0, 15]	6
D	[5, 15]	1

Table 2: Intervals after splitting from Figure 1.

Users	Time interval	Concurrency
C	[0–5]	6
A, C, D	[5–10]	9
B, C, D	[10, 15]	11
B	[15, 20]	4

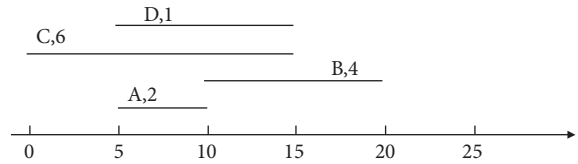


Figure 1: Graphical representation of user records.

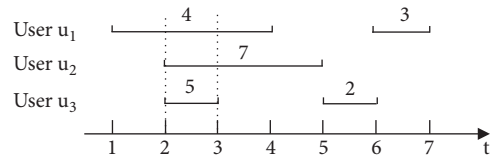


Figure 2: The consumption of the cloud server.

a condition given in the temporal aggregation. But in our problem, we do not know when the concurrency of resource consumption is maximal. In other words, we do not have a certain time interval in our problem which needs to be calculated by our algorithm. Therefore, the algorithms for temporal aggregation cannot solve our problem directly.

3. Pricing Model

As mentioned in Section 1, the maximal concurrency of resource consumption plays an important role in pricing the cloud service. In other words, the price is mainly decided based on the maximal concurrency on cloud servers at the same time.

To further illustrate the meaning of the maximal concurrency, we give an example as follows. Table 1 and Figure 1 show the using time and usage records of four users, A, B, C, and D. Each user record consists of a time interval and its concurrency. The time interval represents the time range of users using the cloud resources. We can easily see that user A uses the cloud resources in time interval $[5, 10]$ and his concurrency is 2.

According to Table 1 and Figure 1, we can split the original interval again, which is shown in Table 2. In Table 2, obviously we can see that 11 is all the users' maximal concurrency of resource consumption. So we find out the cost of the cloud resource provider. After that, we can calculate the user's price based on his usage data and the profit for the cloud resource provider.

Next, based on the maximal concurrency of resource consumption, we can design the following price model.

3.1. Pricing Model

Modeling the Concurrency. Suppose that there is a rate q, which means the cost of each concurrency for the cloud service provider. In addition, in a certain month, the total number of users is n and the maximum concurrency of resource consumption is m. Therefore, the cost of the cloud service provider is $C = q \times m$. As for users, we propose a new pricing model. Firstly, each user needs to pay for B as the monthly rental. In addition, the provider will charge a fee for each user based on his maximal concurrency. Therefore, a user u_i should pay $\text{price}(u_i) = \max_i \times q + B$ for his resource consumption. Finally, the profit of the cloud service provider is

$$\text{profit} = \sum_{i=1}^{n} (\max_i \times q + B) - m \times q. \tag{1}$$

Example. As shown in Figure 2, assume that the user's monthly fare $b = 1$, the rate $q = 2$. From time 1 to 7 is a month. The maximum concurrency of each user u_1, u_2, and u_3 is 4, 7, and 5, respectively. The cloud service fare of users u_1, u_2, and u_3 is $\text{price}(u_1) = 4 \times 2 + 1 = 9$, $\text{price}(u_2) = 7 \times 2 + 1 = 15$, and $\text{price}(u_3) = 5 \times 2 + 1 = 11$. The maximum concurrency of resource consumption is the dotted line part in the figure, which is $m = 4 + 7 + 5 = 16$. Therefore, the profit of the cloud service provider is $\text{profit} = 9 + 15 + 11 - 16 \times 2 = 3$.

However, this pricing model has some drawbacks. For example, the user u_3 uses the cloud server for a little time. But he needs to pay for more fare than user u_1 because of his higher concurrency. Therefore, the merely using concurrency model is not reasonable in some special cases, and we further improve this model by taking resource consumption into account.

Combining Resource Consumption. Assume that a user u_i has x records of cloud resource consumption on the current month and each record starts at t_i and ends at t_i'. The concurrency of the record is m_{t_i}. Therefore, the cloud consumption of u_i is $U_i = \sum_{i=1}^{x} (t'_i - t_i) m_{t_i}$. In addition, the maximal concurrency of user u_i is $\max_i$. The final cloud service fare of u_i can be calculated by following equation:

$$\text{price}(i) = \alpha U_i p + (1 - \alpha) \max_i q + B, \tag{2}$$

where $0 < \alpha < 1$ is a factor that adjusts the using time and the maximum concurrency of users. p is the resource consumption rate which means the cost of each resource consumption. q is the maximum concurrency rate. b is the monthly rental of cloud service. The values of α, p, q, and b can be adjusted according to actual needs.

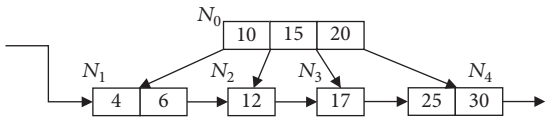


Figure 3: An example of B+tree.

Example. Still using Figure 2 as an example, assume that the user's monthly rental $b = 1$, the maximum concurrency rate $q = 2$, the usage rate $p = 1$, and α is 0.5. Then the consumption of users u_1, u_2, and u_3 is $U_1 = 4 \times 3 + 3 \times 1 = 15$, $U_2 = 7 \times 3 = 21$, $U_3 = 5 \times 1 + 2 \times 1 = 7$. The maximum concurrency of each user u_1, u_2, and u_3 is 4, 7, and 5, respectively. Therefore, the cloud service pare of users u_1, u_2, and u_3 is price(u_1) $= 0.5 \times 15 \times 1 + 0.5 \times 4 \times 2 + 1 = 12.5$, price($u2$) $= 0.5 \times 21 \times 1 + 0.5 \times 7 \times 2 + 1 = 18.5$, and price($u3$) $= 0.5 \times 7 \times 1 + 0.5 \times 5 \times 2 + 1 = 9.5$. The profit of cloud service providers is profit $= 12.5 + 18.5 + 9.5 - 2 \times 16 = 8.5$.

This pricing model compensates for the defect of the former price model which only takes the maximum concurrency of resource consumption into account. This pricing model adds the factor of users' resource consumption, which makes the model more reasonable. Besides, we can adjust the coefficient α to adapt to different cloud resources and make the model more flexible.

4. Algorithm Description

In Section 3, we propose a flexible dynamic pricing model for cloud resources. As the maximal concurrency of resource consumption plays an important role in the pricing model, how to efficiently calculate the maximal concurrency of resource consumption is a difficulty in our problem. Because when we calculate the price for resource consumption in cloud service, we should calculate the maximal concurrency of resource consumption first. Therefore, we propose a new data structure and the operational algorithms based on it to solve this problem.

In this part, we will introduce the new data structure called B++-tree which extends from B+ tree. We first introduce the structure of the B+ tree and then introduce the structure of the B++-tree. Finally, we introduce the insertion, deletion, and split operations of B++-tree, which can calculate the maximal concurrency of resource consumption.

The nodes of B+ tree can store a lot of index entries, which helps reduce the height of the tree. Besides, the leaf nodes of B+ tree are connected by pointers. It is very suitable for the query. The following is a brief introduction for the B+ tree.

Each B+ tree has a parameter called capacity c, which determines the maximal capacity for each node. For each interior node, it contains k ($k < c$) elements and points to $k + 1$ child nodes. For each leaf node, it contains l ($l < c$) elements and have a pointer to its right sibling node as shown in Figure 3. In this way, it is efficient to traverse all leaf nodes by utilizing their pointers rather than traversing from the root.

But a simple B+ tree cannot solve our problem, because we need to store the concurrency of resource consumption into the tree. So we propose a new data structure called B++-tree, which extends from B+ tree.

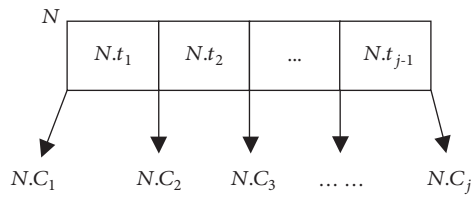


Figure 4: An interior node of B++-tree.

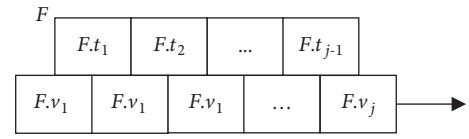


Figure 5: A leaf node of B++-tree.

4.1. Structure of B++-Tree. Different from B+ tree, we add an addition attribute for each leaf node of B++-tree, which is used to store the concurrency of each interval. Note that the interior nodes keep the same structure as the B+ tree and do not have the additional attribute. In this way, the concurrency of all intervals is stored in leaf nodes, and we can only traverse leaf nodes to obtain the final result, which improve the performance of our algorithm efficiently.

In the B++-tree, each node can hold up to c timestamps. If the number of timestamps stored in the node exceeds c, we should call *split* (Section 4.4) process to split the overflowed node into two new nodes. Actually, two adjacent timestamps in a node represent an interval. The interval is a criterion which can help us determine the child node that we need to traverse when we need to insert or delete a new record. The detailed structures of interior nodes and leaf nodes are as follows.

Interior Node. The structure of an interior node is shown in Figure 4. Each interior node contains j ($j \leq c$) intervals, $N \cdot I_1, \ldots, N \cdot I_i, \ldots, N \cdot I_j$. $N \cdot C_1$ to $N \cdot C_j$ are the corresponding pointer to the child nodes. $N \cdot I_i$ is associated with $N \cdot C_i$.

$N \cdot I_i = [N \cdot t_{i-1}, N \cdot t_i]$ is the ith interval. There are two special conditions:

(1) The start time of $N \cdot I_1$ is $-\infty$, if node N is the root node or it is the first child. Otherwise, the start time of $N \cdot I_1$ is $N' \cdot t_{k-1}$, where N' is the parent node of N and $N' \cdot C_k$ points to N.

(2) The end time of $N \cdot I_j$ is $+\infty$, if this node is the root node or it is the last child. Otherwise, the end time of $N \cdot I_j$ is $N' \cdot t_k$, where N' is the parent node of N and $N' \cdot C_k$ points to N.

Leaf Node. Figure 5 shows the structure of a leaf node. Compared with interior nodes, leaf nodes have additional attributes which store the concurrencies of intervals. The definition of leaf nodes' intervals is the same as interior nodes. Moreover, each leaf node has a pointer which points to its next

Input: Interval I.
Output: B++-tree after inserting I.
(1) Start from the root node N
(2) **for** each interval $N \cdot I$ in N. **do**
(3) **if** interval $N \cdot I$ intersects with I
(4) **if** N is the leaf node
(5) Set the common interval of $N \cdot I$ and I as the new interval, add v of $N \cdot I$ and v of I to the new interval
(6) **else**
(7) search into the child node N' of N, insert I to N', back to step (2).
(8) **end if**
(9) **end if**
(10) **end for**

Algorithm 1: Insert operation.

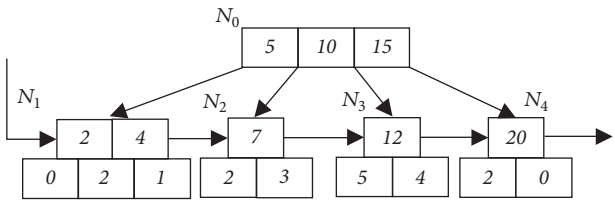


Figure 6: An example of B++-tree.

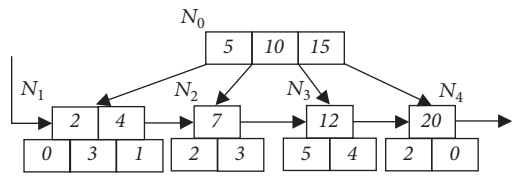


Figure 7: The B++-tree after inserting $\langle [2, 4], 1 \rangle$.

sibling leaf node. Specially, the last leaf node has no pointer. In addition, there is a header pointer pointing to the first leaf node.

For example, Figure 6 plots the example of B++-tree. The first interval of N_0 is $[-\infty, 5]$ and the last interval of N_0 is $[15, +\infty]$, because it is a root node. The first interval of N_1 is $[-\infty, 2]$, because it is the first child of its parent, and the concurrency of the interval is 0. The first interval of N_2 is $[5, 7]$, and the concurrency of the interval is 2.

4.2. Insertion

Main Idea. In this section, we introduce the insert operation. We define the procedure insert $(\langle I, v \rangle, N)$ as an insert operation, where I indicates the user's usage interval, such as $[5, 10]$, v indicates the user's concurrency, and N indicates the node that to insert. Insertion will be firstly processed from the root node N_{root}. We firstly traverse the root node and find the intervals which intersect with I. If $N_{\text{root}} \cdot I_i$ intersects with I, we search in the $N_{\text{root}} \cdot C_i$ and traverse it in the same way until the leaf nodes. When we traverse to a leaf node N_l, we find the intervals which intersect with I and add the v to its corresponding concurrency.

Description. Algorithm 1 shows the pseudo code of insertion. The input is the interval I, and the output is the B++-tree after insert I. Suppose that now we want to insert the record $\langle I, v \rangle$ into the tree. Start from the root node N (line (1)), for each interval $N \cdot I_i$ of N:

(1) If $N \cdot I_i$ intersects with the time interval I and N is a leaf node, and if interval I contains $N \cdot I_i$, then add v to $N \cdot I_i$ directly (line (5)). If interval I does not contain $N \cdot I_i$ but only intersects with it, then we take their intersecting interval as a new interval and add v to the new interval. The original interval keeps unchanged (line (2)~(5)).

(2) If $N \cdot I_i$ intersects with the time interval I but N is an interior node, then call insert $(\langle I, v \rangle, N \cdot C_i)$, i.e., regarding $N \cdot C_i$ as N, executed the algorithm start from the second step (line (6)~(7)).

Example. For example, we want to insert $\langle [2, 4], 1 \rangle$ to the tree in Figure 6. As for N_0, only interval $(-\infty, 5]$ intersects with interval $[2, 4]$, so we insert $\langle [2, 4], 1 \rangle$ to its child node. Then we find the second interval of $N_1[2, 4]$, which is contained by the interval that we want to insert. So we add v to $N \cdot I_i$ directly, as shown in Figure 7.

As a slightly more complicated example, suppose that we want to insert $\langle [5, 8], 4 \rangle$ to the tree in Figure 6. Firstly we start from the root node N_0. We can see that only interval $[5, 10]$ intersects with the interval $[5, 8]$, so we should still call insert $(\langle [5, 8], 4 \rangle, N_2)$. Now N_2 is the leaf node and the interval $[5, 7]$ and $[7, 10]$ both intersect with the interval $[5, 8]$. As for $[5, 7]$, it is contained by $[5, 8]$, so add v directly. As for $[7, 10]$, $[5, 8]$, they have an intersecting interval $[7, 8]$. So we create a new interval $[7, 8]$ and its concurrency is $3+4 = 7$. Other intervals keep unchanged. Then we complete the insertion of $\langle [5, 8], 4 \rangle$, as shown in Figure 8.

Input: Node N which is overflowed;
Output: Node N_1 and N_2 which are split from N.
(1) create new node N_1 and N_2
(2) N_1 retains the first half intervals of the original node N; N_2 retains the rest of intervals
(3) **if** N is the interior node
(4) N_1 retains the first half pointers, N_2 retains the rest of pointers
(5) **if** N is the leaf node
(6) N_1 retains the first $\lceil n/2 \rceil$ values, N_2 retains the rest values
(7) **if** N is the root node
(8) create a new root node N', make it be the parent node of N_1 and N_2. Make N' be the new root node
(9) **else if** N is not the root node
(10) suppose N' is the parent node of N, make N' be the parent of N_1 and N_2
(11) **end if**
(12) **If** N' is overflowed
(13) **Split** N'
(14) **end if**
(15) **end**
(16) **end**

Algorithm 2: Split operation.

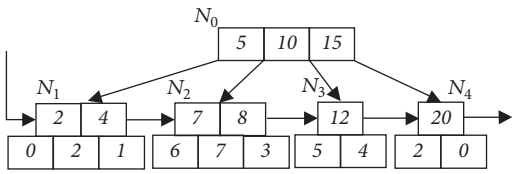


Figure 8: The B++-tree after inserting $\langle [5, 8], 4 \rangle$.

4.3. Deletion. The deletion operation is similar to the insertion operation. We can regard the deletion operation as an opposing operation to the insertion. Specifically, if we want to delete the record $\langle I, v \rangle$, we can insert a record $\langle I, -v \rangle$. Therefore, deletion operation can be easily understood without extra explanation.

4.4. Split

Main Idea. As records are inserted, the number of leaf nodes' intervals gradually increases. As mentioned above, each node can hold up to c intervals. Therefore, when a node N becomes overflowed, i.e., its number of intervals exceeds c, we should split N into two nodes N_1 and N_2. The first half intervals of N are assigned to N_1 and the remaining intervals of N are assigned to N_2. Suppose N' is the parent node of N. Because of the split of N, the numbers of interval in N' will be added 1. If N' also become overflowed, split N'.

Description. Algorithm 2 shows the pseudo code of split. The input is the node N which is overflowed and the output is nodes N_1 and N_2 which are split from N. Suppose that node N is overflowed and currently stores n intervals. We define the procedure as split (N). Specific operations are as follows:

(1) Node N splits into N_1 and N_2. Node N_1 retains the first half intervals of the original node N; i.e., N_1 retains the first $\lceil n/2 - 1 \rceil$ time points. If N is an interior node, N_1 also retains the pointer from $N \cdot C_1$ to $N \cdot C_{\lceil n/2 \rceil}$. If N is a leaf node, N_1 also retains the first $\lceil n/2 \rceil$ values from $N \cdot v_1$ to $N \cdot v_{\lceil n/2 \rceil}$. Node N_2 retains the rest of intervals, which means N_2 retains the time points from $N \cdot t_{\lceil n/2 \rceil + 1}$ to $N \cdot t_{n-1}$. If N is the interior node, N_2 also retains the pointer from $N \cdot C_{\lceil n/2 \rceil + 1}$ to $N \cdot C_n$. If N is the leaf node, N_2 also retains the value from $N \cdot v_{\lceil n/2 \rceil + 1}$ to $N \cdot v_n$ (lines (1)~(6)).

(2) If node N is the root node, then create a new root node N', which is the parent node of N_1 and N_2. Make $N' \cdot t_1 = N \cdot t_{\lceil n/2 \rceil}$, $N' \cdot C_1 = N_1$, and $N' \cdot C_2 = N_2$ (line (7)~(8)).

(3) If node N is not the root node, suppose that N' is the parent node of N, and $N' \cdot C_j = N$. Keep the first $j-1$ intervals of N' unchanged. Then starting from the $j + 1$th interval to the end, move them to the right one. Make $N' \cdot t_j = N \cdot t_{\lceil n/2 \rceil}$, $N' \cdot C_j = N_1$, and $N' \cdot C_{j+1} = N_2$. If node N' is overflowed, call split (N') (lines (9)~(13)).

Example. For example, Figures 9-10 show us how the nodes split. Suppose that the capacity of the tree is 4. This means if the number of node's intervals exceeds 4, the node needs to be split. The node N_3 in Figure 9 has 5 intervals, $[30, 35]$, $[35, 40]$, $[40, 45]$, $[45, 50]$, and $[50, +\infty]$, whose intervals are bigger than 4, so the node N_3 should be split.

According to the split rules, node N_3 split into N_4 and N_5. N_4 keeps the first 3 intervals: $[30, 35]$, $[35, 40]$, and $[40, 45]$. N_5 keeps the remaining intervals: $[45, 50]$, $[50, +\infty]$. Because

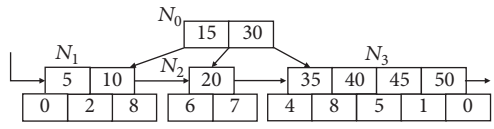


Figure 9: The B++-tree before splitting operation.

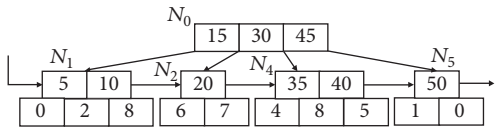


Figure 10: The B++-tree after splitting operation.

the node N_3 is leaf node, so N_4 keeps the first 3 values and N_5 keeps last 2 values. Move $N_3 \cdot t_{\lceil n/2 \rceil}$ to N_0. The result of the split is shown in Figure 10.

4.5. Query for the Maximal Concurrency. When the index building is complete, we can traverse all leaf nodes to acquire each interval and corresponding concurrency. Since all leaf nodes are connected by pointers, we can easily traverse these leaf nodes sequentially without traversing any interior node.

For example, in Figure 6, traversing the leaf node we can get the result: $\langle[2,4],2\rangle$, $\langle[4,5],1\rangle$, $\langle[5,7],6\rangle$, $\langle[7,8],7\rangle$, $\langle[8,10],3\rangle$, $\langle[10,12],5\rangle$, $\langle[12,15],4\rangle$, $\langle[15,20],2\rangle$. It is easy to see that the maximum number of concurrent users is 7.

When we get the maximal concurrency, we get the cost of cloud resource provider. Then we can use our pricing model to charge each user.

5. Experiment

In this section, we provide experimental evaluation of our algorithm. We simulate five datasets which contain 10,000, 100,000, 500,000, 1 million, and 2 million records, respectively. Each record in the dataset contains the user's name, time interval, and the concurrency. For example, "$u1$--> [2017-07-31 11:46:15, 2017-07-31 21:02:56], 4" is a record. "$u1$" is the user's name, and his time interval is from "2017-07-31 11:46:15" to "2017-07-31 21:02:56"; the concurrency is 4. We design a series of experiments in the construction of the B++-tree and the performance of operations, like query, insertion, and deletion. There are two explicit factors in our experiments, which are the data size and the capacity c. Thus, we conduct two sets of experiments by changing the data size and the capacity. For the one set of experiments, we change the data size while fixing the value of capacity. For the other set of experiments, we change the capacity while fixing the data size.

5.1. Construction of the B++-Tree. Firstly, we test the performance of the construction of B++-tree. In this experiment, we change the data size and capacity to compare with the time needed to build a B++-tree.

In Figure 11(a), we vary the data size from 10,000 to 2,000,000, while fixing the capacity to 50. We denote the time of constructing a B++-tree as CT, which means the construction time. As we can see, when the data size is small, for example 10,000 and 100,000, their CTs are very small and similar. But with the growing of data size, CT is also increasing. The greater the size of data is, the more CT consumed. Because with the increment of data size, the structure of B++-tree will be more complicated. Therefore, the construction operation consumes more time.

In Figure 11(b), we vary the capacity of B++-tree from 10 to 100, while fixing the data size to 1,000,000. It is easy to find that CT is the shortest when the capacity $c = 35$. If the capacity is too small, the tree will be very high. If the capacity c is too big, the node will store too many intervals. Neither of these conditions contributes to the construction of the B++-tree. Therefore, the capacity $c = 35$ is the most suitable for the construction of B++-tree, while the data size is 1,000,000.

5.2. Performance of Operations. In this subsection, we test the performance of operations, which includes the query performance, the insertion performance, and the deletion performance. Then we analyze the reason of the performance and give the conclusion of experiments.

Query. Firstly we test the performance of query. Figure 12 gives the result of the experiment. We change c and the data size separately to conduct comparative experiments. We denote the time of traversing the leaf nodes as TT, which means the traversal time.

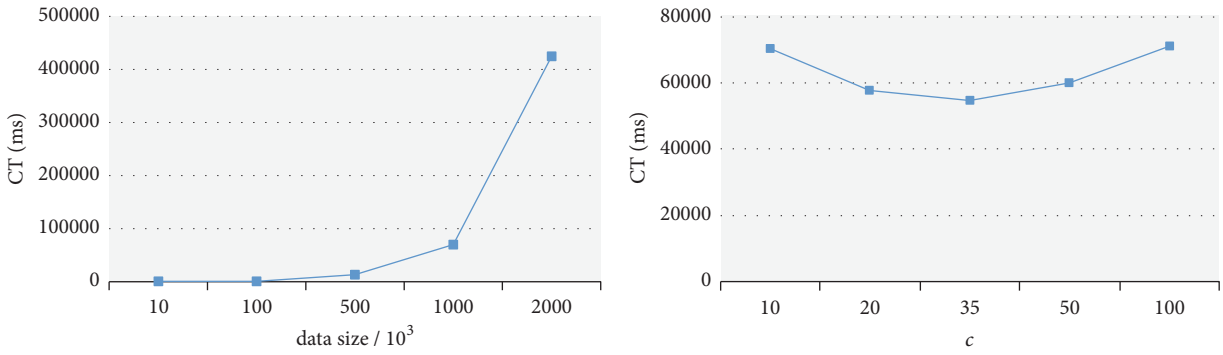
In Figure 12(a), we vary the data size from 10,000 to 2,000,000 while fixing the capacity to 50. From the figure, we can see that the greater the data size is, the bigger the TT is. Because the larger the data size is, the more leaf nodes the B++-tree has. Therefore, it takes more time to traverse the B++-tree.

In Figure 12(b), we vary the capacity c from 10 to 100 while fixing the data size to 1,000,000. The trend of TT is decreasing first but increasing again. Because when the capacity c is small, there will be too many leaf nodes in B++-tree, which cost much more time to traverse the B++-tree. When the capacity c is too big, the B++-tree will has too many leaf nodes. So when $c = 35$, the query performance is the best, while the data size is 1,000,000.

But the most important is that TT is very small all the time, which means query speed is very fast. Even the data size is 2 million; the query time is less than a second. This shows that the B++-tree we proposed is very suitable for the query operation.

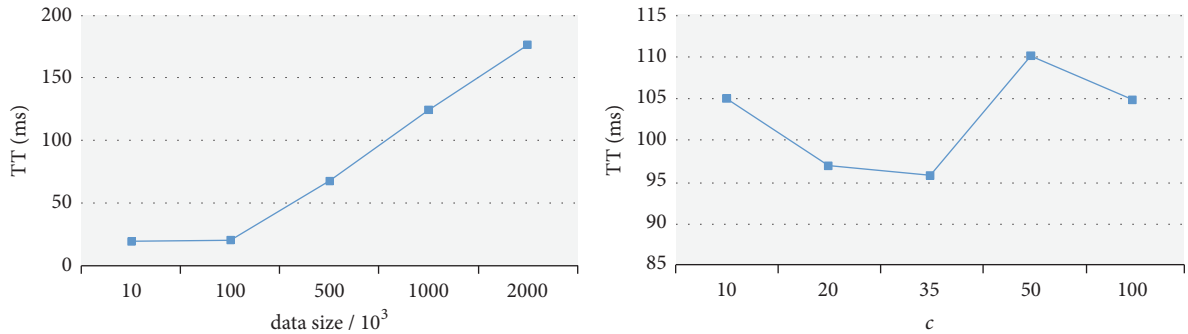
Insertion. Secondly we investigate the insertion performance of B++-tree. Data insertion is the most common operation in constructing a B++-tree. We still test the insertion performance by changing the data size and capacity. We compare the time of inserting a bunch of data and the time of inserting a piece of data. The time of inserting data is called IT, which means the insertion time.

In Figure 13, we compare the time of inserting a bunch of data. In Figure 13(a), we vary the data size from 10,000 to 2,000,000 while fixing the capacity to 50 and the inserting data size to 5000. From the figure, we can see that with the data size increasing, the IT increases as well. Because



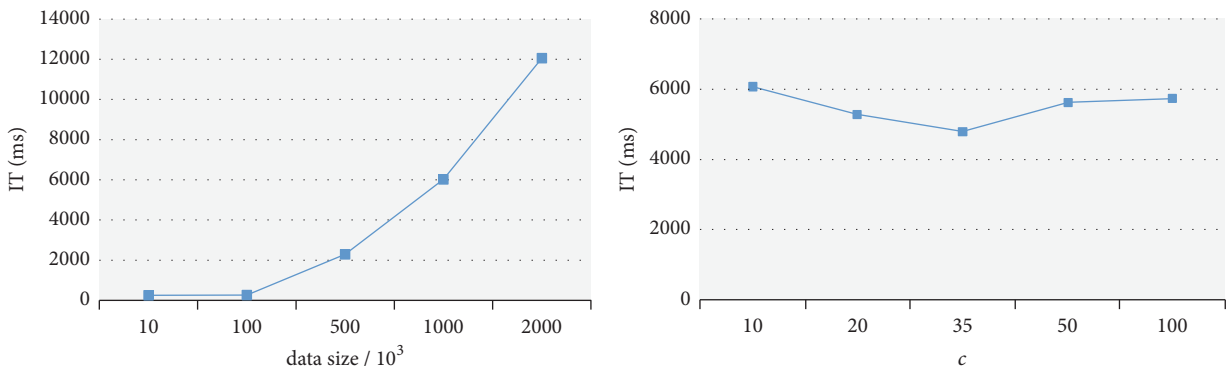
(a) Varying data size (b) Varying c

Figure 11: Experimental results of CT.



(a) Varying data size (b) Varying c

Figure 12: Experimental results of TT.



(a) Varying data size (b) Varying c

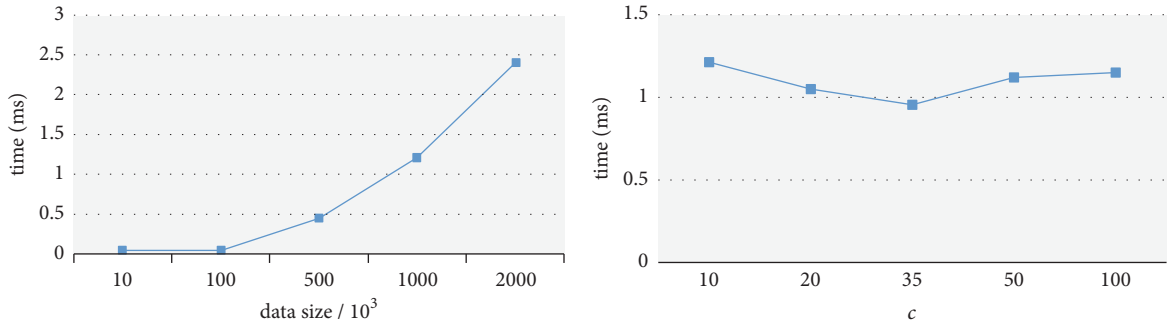
Figure 13: Experimental results of IT.

with the data size increasing, the B++-tree becomes more complicated. We need traverse more nodes to insert a record.

In Figure 13(b), we vary the capacity from 10 to 100 while fixing the data size to 1,000,000 and the inserting data size to 5000. Same as the previous experiment of query, IT is the shortest when $c = 35$, because the structure of the B++-tree is the best at this capacity. When the capacity $c = 35$, B++-tree will not have too many nodes or too many intervals in a leaf node.

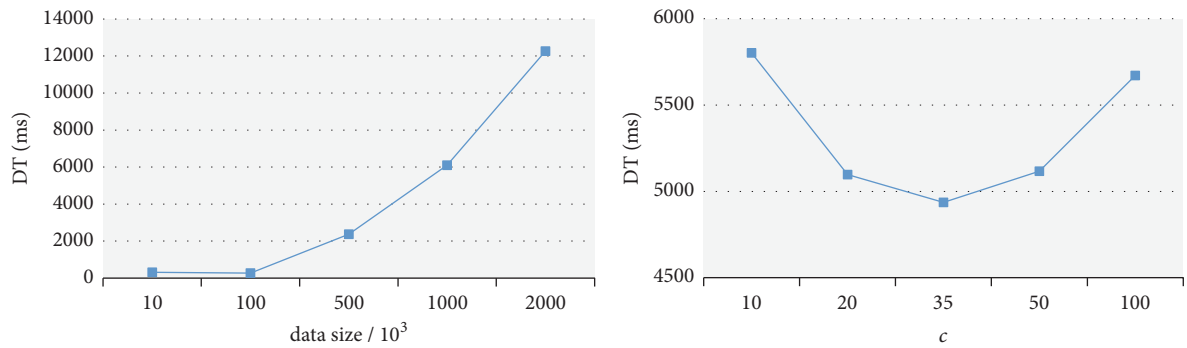
In Figure 14, we compare the time of inserting a piece of data. In Figure 14(a), we vary the data size from 10,000 to 2,000,000 while fixing the capacity to 50. We can see that the trend of Figure 14(a) is similar to Figure 13(a). It only takes 2 ms to insert a piece of data when the data size is 2,000,000.

In Figure 14(b), we vary the capacity from 10 to 100 while fixing the data size to 1,000,000. The time of inserting a piece of data is only about 1 ms. It is obvious that B++-tree is very efficient in data insertion.



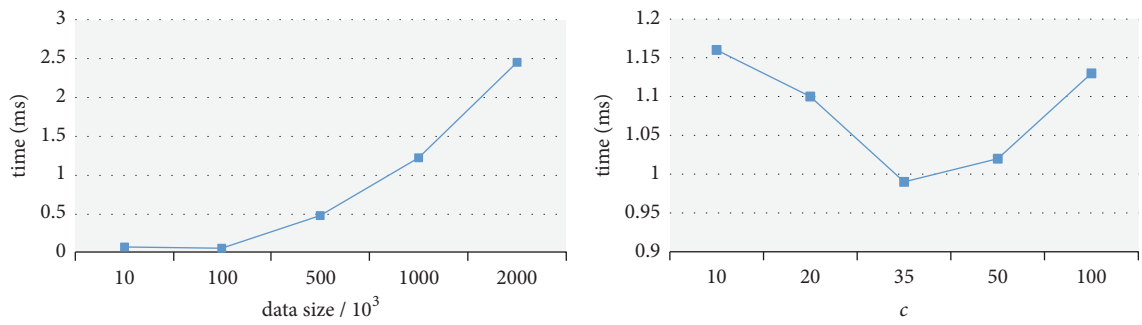
(a) Varying data size (b) Varying c

Figure 14: Experimental results of inserting a piece of data.



(a) Varying data size (b) Varying c

Figure 15: Experimental results of DT.



(a) Varying data size (b) Varying c

Figure 16: Experimental results of deleting a piece of data.

Deletion. Thirdly we investigate the deletion performance of B++-tree. Actually the principle of deletion is the same as the principle of insertion. Like the data insertion, we also test the deletion performance by changing the data size and capacity. We compare the time of deleting a bunch of data and the time of deleting a piece of data. The time of deleting data is called DT, which means the deletion time.

In Figure 15, we compare the time of deleting a bunch of data. In Figure 15(a), we vary the data size from 10,000 to 2,000,000 while fixing the capacity to 50 and the deleting data size to 5000. The result of Figure 15 is similar to Figure 13. With the increment of data size, the time of deletion increases as well, because of traversing more nodes.

In Figure 15(b), we vary the capacity from 10 to 100 while fixing the data size to 1,000,000 and the deleting data size to 5000. Same as Figure 13(b), DT is the shortest when $c = 35$. And the reason is the same as in Figure 13(b).

In Figure 16, we compare the time of deleting a piece of data. Figure 16 shows the same result as Figure 14. In Figure 16(a), we vary the data size from 10,000 to 2,000,000 while fixing the capacity to 50. The time of deleting a piece of data is very similar to the time of inserting a piece of data.

In Figure 16(b), we vary the capacity from 10 to 100 while fixing the data size to 1,000,000. The time of deleting a piece of data is only about 1 ms. The result shows that B++-tree is suitable for data deletion.

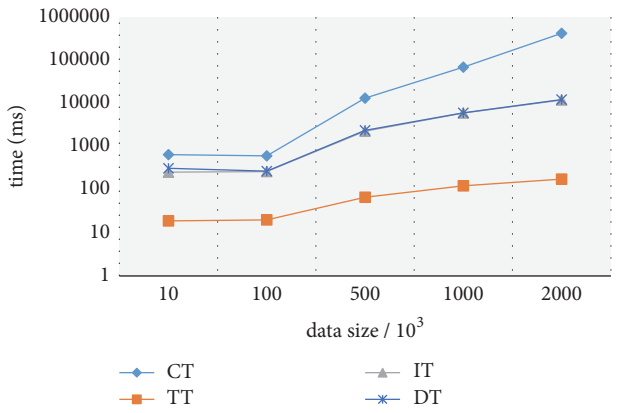


Figure 17: The trend of CT, TT, IT, and DT when varying the data sizes.

Finally we give the conclusion of experiment. We combine the results of the previous experiments and put them into Figure 17.

For ease of viewing, we set the ordinate of Figure 17 to a logarithmic scale of 10. From the figure we can see IT is almost the same as DT, because those two operations are the same in principle. With the increment of data size, TT, CT, IT, and DT increase at the same time. But the growth rate of the TT is not very big. Because according to the description, the query operation only needs to traverse leaf nodes, which is very fast and efficient. Besides, we can see from Figure 17 that the time of query is very short and less than a second.

As we can see from our experiments, the B++-tree we proposed in this paper is well suited for calculating the maximal concurrency for our pricing model.

6. Conclusion

In this paper, we propose a dynamic pricing model, which takes into account using time, resource consumption, and maximum concurrency to make the price of cloud resources more reasonable for both users and providers. In order to calculate the maximal concurrency of all the users, we propose a new data structure, named B++-tree, which extends from B+tree and has additional information in leaf nodes. Besides, we introduce the insertion, deletion, split, and query operation of B++-tree, which can calculate the maximal concurrency.

Finally, we performed extensive experiments to study the performance of the construction, query, insertion, and deletion operations with different data sizes and capacities of B++-tree. The result of the experiments shows that the B++-tree we proposed in this paper is well suited for calculating the maximal concurrency for our pricing model. We can complete the query operation on 10 million data in only 0.2 seconds. For the future work, we plan to find a much more reasonable pricing model which can take more factors into account.

Data Availability

The 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 research was partially supported by the National Natural Science Foundation of China (nos. 61602411 and 61572437) and Key Research and Development Project of Zhejiang Province (no. 2015C01034).

References

[1] J. Lee, "A view of cloud computing," *Communications of the Acm*, vol. 53, no. 4, pp. 50–58, 2013.

[2] K. Sowmya and R. P. Sundarraj, "Strategic bidding for cloud resources under dynamic pricing schemes," in *Proceedings of the International Symposium on Cloud and Services Computing (ISCOS '12)*, pp. 25–30, IEEE, Mangalore, India, December 2012.

[3] A. Zhou, S. Wang, Q. Sun, H. Zou, and F. Yang, "Dynamic virtual resource renting method for maximizing the profits of a cloud service provider in a dynamic pricing model," in *Proceedings of the 2013 International Conference on Parallel and Distributed Systems (ICPADS '13)*, pp. 118–125, Seoul, Korea (South), December 2013.

[4] S. Wang, T. Lei, L. Zhang, C.-H. Hsu, and F. Yang, "Offloading mobile data traffic for QoS-aware service provision in vehicular cyber-physical systems," *Future Generation Computer Systems*, vol. 61, pp. 118–127, 2016.

[5] S. Wang, A. Zhou, C.-H. Hsu, X. Xiao, and F. Yang, "Provision of data-intensive services through energy-and QoS-aware virtual machine placement in national cloud data centers," *IEEE Transactions on Emerging Topics in Computing*, vol. 4, no. 2, pp. 290–300, 2016.

[6] Y. Ma, S. Wang, P. C. Hung, C. H. Hsu, Q. Sun, and F. Yang, "A highly accurate prediction algorithm for unknown web service QoS value," *IEEE Transactions on Services Computing*, vol. 9, no. 4, pp. 511–523, 2016.

[7] A. Zhou, S. Wang, Z. Zheng, C.-H. Hsu, M. R. Lyu, and F. Yang, "On cloud service reliability enhancement with optimal resource usage," *IEEE Transactions on Cloud Computing*, vol. 4, no. 4, pp. 452–466, 2016.

[8] E. Mykletun and G. Tsudik, "Aggregation queries in the database-as-a-service model," in *Proceedings of the IFIP Conference on Data and Applications Security and Privacy*, vol. 4127, pp. 89–103, Springer Berlin Heidelberg, 2006.

[9] H. Wang, Q. Jing, and R. Chen, "Distributed systems meet economics: pricing in the cloud," in *Proceedings of the USENIX Conference on Hot Topics in Cloud Computing*, 2010.

[10] C. Kilcioglu and J. M. Rao, "Competition on price and quality in cloud computing," in *Proceedings of the International Conference on World Wide Web*, pp. 1123–1132, April 2016.

[11] M. Al-Roomi, S. Al-Ebrahim, S. Buqrais et al., "Cloud computing pricing models: a survey," *International Journal of Grid & Distributed Computing*, vol. 6, no. 5, pp. 93–106, 2013.

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

[13] S. Gu, Z. Li, C. Wu, and C. Huang, "An efficient auction mechanism for service chains in the NFV market," in *Proceedings of the IEEE INFOCOM 2016—IEEE Conference on Computer Communications*, pp. 1–9, San Francisco, Calif, USA, April 2016.

[14] W.-Y. Lin, G.-Y. Lin, and H.-Y. Wei, "Dynamic auction mechanism for cloud resource allocation," in *Proceedings of the 10th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing (CCGrid '10)*, pp. 591-592, May 2010.

[15] R. Zhou, Z. Li, C. Wu, and Z. Huang, "An Efficient Cloud Market Mechanism for Computing Jobs with Soft Deadlines," *IEEE/ACM Transactions on Networking*, vol. 25, no. 2, pp. 793–805, 2017.

[16] W. Shi, L. Zhang, C. Wu, Z. Li, and F. C. M. Lau, "An online auction framework for dynamic resource provisioning in cloud computing," in *Proceedings of the 2014 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS '14)*, pp. 71–83, June 2014.

[17] L. Zheng, W. Tarneberg, and K. Maria, "Using a Predator-Prey Model to Explain Variations of Cloud Spot Price," *Journal of Intelligent & Fuzzy Systems: Applications in Engineering and Technology*, vol. 28, no. 6, pp. 2679–2689, 2017.

[18] Z. Zhang, Z. Li, and C. Wu, "Optimal posted prices for online cloud resource allocation," *ACM SIGMETRICS Performance Evaluation Review*, vol. 44, no. 1, pp. 60-60, 2017.

[19] N. Kline and R. T. Snodgrass, "Computing temporal aggregates," in *Proceedings of the 1995 IEEE 11th International Conference on Data Engineering*, pp. 222–231, March 1995.

[20] B. Moon, I. F. V. Lopez, and V. Immanuel, "Scalable algorithms for large temporal aggregation," in *Proceedings of the 2000 IEEE 16th International Conference on Data Engineering (ICDE'00)*, pp. 145–154, March 2000.

[21] B. Moon, I. F. V. Lopez, and V. Immanuel, "Efficient algorithms for large-scale temporal aggregation," *IEEE Transactions on Knowledge and Data Engineering*, vol. 15, no. 3, pp. 744–759, 2003.

[22] J. Yang and J. Widom, "Incremental computation and maintenance of temporal aggregates," *The VLDB Journal*, vol. 12, no. 3, pp. 262–283, 2003.

[23] R. Bayer and E. McCreight, "Organization and maintenance of large ordered indices," in *Proceedings of the 1970 ACM SIGMOD Workshop on Data Description, Access and Control (SIGFIDET '70)*, pp. 107–141, November 1970.

[24] G. Graefe and H. Kuno, "Modern B-tree techniques," in *Proceedings of the 2011 IEEE 27th International Conference on Data Engineering (ICDE '11)*, pp. 1370–1373, April 2011.

[25] P. Yao, H. Zhang, Y. Xue et al., "Segment-tree based cost aggregation for stereo matching with enhanced segmentation advantage," in *Proceedings of the 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP '17)*, pp. 2027–2031, New Orleans, LA, USA, March 2017.

Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 4186243, 19 pages
https://doi.org/10.1155/2018/4186243

Research Article

Dynamic Outsourced Proofs of Retrievability Enabling Auditing Migration for Remote Storage Security

Lu Rao, Tengfei Tu, Hua Zhang, Qiaoyan Wen, and Jia Xiao

State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing, China

Correspondence should be addressed to Hua Zhang; zhanghua_288@bupt.edu.cn

Received 26 December 2017; Accepted 18 February 2018; Published 9 May 2018

Academic Editor: Kok-Seng Wong

Remote data auditing service is important for mobile clients to guarantee the intactness of their outsourced data stored at cloud side. To relieve mobile client from the nonnegligible burden incurred by performing the frequent data auditing, more and more literatures propose that the execution of such data auditing should be migrated from mobile client to third-party auditor (TPA). However, existing public auditing schemes always assume that TPA is reliable, which is the potential risk for outsourced data security. Although Outsourced Proofs of Retrievability (OPOR) have been proposed to further protect against the malicious TPA and collusion among any two entities, the original OPOR scheme applies only to the static data, which is the limitation that should be solved for enabling data dynamics. In this paper, we design a novel authenticated data structure called bv23Tree, which enables client to batch-verify the indices and values of any number of appointed leaves all at once for efficiency. By utilizing bv23Tree and a hierarchical storage structure, we present the first solution for Dynamic OPOR (DOPOR), which extends the OPOR model to support dynamic updates of the outsourced data. Extensive security and performance analyses show the reliability and effectiveness of our proposed scheme.

1. Introduction

In today's information era of data explosion, it is an inevitable trend for most people to have the ever-increasing big data storage demands. Storage outsourcing through the cloud has become a promising technology paradigm that populates the recent literatures [1–3] and has been regarded as a faster profit growth point [4] by various IT industry giants (e.g., Google Drive, Microsoft OneDrive, and Amazon EC2 and S3). Cloud storage not only allows mobile clients to access their outsourced data from anywhere at any time, but also provides mobile clients with many benefits such as the inexpensive storage cost and elastic configuration of the storage capacity, which attract more and more mobile clients (e.g., smartphones and laptops) to join the cloud for the convenient lifestyle.

However, at the side of the cloud storage server (CSS), there still exist all kinds of internal and external threats against the storage security of outsourced data, such as Byzantine failures, the monetary reasons, and the hacker attacks [5, 6]. So, it is well known that CSS would be considered as a malicious entity that might try to hide the accident when data loss occurs, or even deliberately delete client's data for saving storage cost. In this case, for the mobile client who has not actually possessed her data after storage outsourcing, an urgent requirement is how to guarantee the *correctness* and *retrievability* of the outsourced data. Here, correctness guarantee means that any appointed data returned from CSS should be the latest version of the authentic data, and retrievability guarantee means that the whole outsourced data can be correctly retrieved by the client without any data loss.

Based on the application of erasure code and the periodic auditing against CSS, the security model Proof of Retrievability (POR) [7, 8] is defined to offer client-side devices the above two guarantees in the context of malicious CSS. However, given the fact that most mobile clients only have a limited capacity so that these clients are unlikely to keep online all the time to perform the frequent auditing, various public auditing schemes are proposed [5, 6, 9–11] for auditing migration, which enables mobile client to free herself by moving the heavy auditing tasks to a third-party auditor (TPA). But existing public schemes rely on the hypothesis

that TPA is trusted to complete the migrated auditing tasks, meaning that these schemes do not provide any security guarantee to resist a malicious TPA that might break the auditing protocols, which is exactly the potential risk that has not been covered by current public auditing schemes [12].

The first Outsourced Proof of Retrievability (OPOR) solution, proposed by Armknecht et al. and called Fortress [12], is a stronger security model to protect against any malicious entity (e.g., malicious TPA) and against collusion among any two entities. Fortress enables auditing migration, but it is just a static scheme. Supporting dynamic updates is an essential requirement for numerous practical cloud storage applications [13]. Although all kinds of authenticated data structures [6, 9, 13–16] have been proposed to support data dynamics, there still exist research gaps in these structures. For one thing, most existing dynamic authenticated data structures [9, 14–16] are designed based on the Merkle Hash Tree (MHT). Unfortunately, the use of MHT is not efficient in some cases since MHT is an unbalanced tree. For example, the height of MHT will increase linearly when many new data blocks are continuously inserted in the same leaf position, so in this worst case the expected $O(\log n)$ performance cannot be ensured for authenticating the index and value of any appointed leaf node via MHT. For another, although some other authenticated structures such as rb23Tree [6] and Skip List [13] are proposed for dynamism, when there is a need to verify multiple leaf nodes of different data blocks (i.e., authenticate the indices and values of these leaf nodes), all above-mentioned dynamic methods merely adopt the straightforward way of verifying these different leaf nodes one by one with their respective proof paths. Since the verification upon a single proof path is of $O(\log n)$ bandwidth and computation costs, as the number of verified leaf nodes increases, it is clearly not an efficient way to separately verify that many leaf nodes one by one. Although a balanced authenticated data structure (e.g., the rb23Tree [6]) constructed upon the balanced tree can avoid the worst case of MHT as discussed above, however, to the best of our knowledge, there is no such balanced authenticated structure that can deal with the problem of how to efficiently batch-verify any number of appointed leaf nodes altogether, which is a limitation that should be further addressed in this paper.

Furthermore, to support data dynamics when applying erasure code, the scheme of [6] is based on the way of local coding, that is, encoding each raw data block individually to ensure that an update upon any raw block only affects a small amount of the encoded blocks. However, such local coding solution is vulnerable to the selective deletion attack from malicious CSS [14], because once a targeted raw block has been updated in the case of local coding, CSS can learn which congenetic encoded blocks correspond to this targeted block. Then, CSS can selectively delete these congenetic encoded blocks to actually cause the loss of the targeted block, and simultaneously CSS can pass the data auditing with significant probability, since the auditing relies on the sampling technology that can hardly cover these selectively deleted encoded blocks if the size of deleted data is tiny. Although both the private and the public Dynamic POR (DPOR) schemes [14] are proposed to resist the above selective deletion attack, the direct application of these two DPOR schemes into OPOR model will result in security or efficiency problems. On the one hand, within the private DPOR of [14], the auditing can only be executed using client's secret key. But TPA is prohibited from obtaining such secret in OPOR, or otherwise the malicious TPA might share client's secret key with malicious CSS [12] so that CSS can break the auditing protocols without actually holding the outsourced data. On the other hand, in order to support public auditing, the public DPOR of [14] cannot apply the blockless verification technique [8, 9, 13, 17] that combines multiple challenged blocks into a single aggregated block for efficiency, so it has to use the straightforward way of requiring TPA to retrieve all randomly challenged actual blocks during each POR audit. As shown in [9], this straightforward way could lead to a large communication overhead and thus is inefficient and should be avoided.

From the above, there will be various problems if the current dynamic schemes are directly ported to the OPOR model. In this paper, to solve these mentioned problems, we propose a concrete Dynamic Outsourced Proofs of Retrievability (DOPOR) scheme enabling auditing migration. DOPOR not only can defend against the malicious TPA and collusion, but also can enable efficient data dynamics under the setting of erasure code. Specifically, our contributions are summarized as follows.

(1) Different from traditional authenticated structures that only have the ability to verify different leaves one by one, we propose the novel authenticated data structure called bv23Tree, which is based on the balanced 2-3 tree to ensure the logarithmic complexity in any case of updates and simultaneously enables the verifier to batch-verify the indices and values of multiple appointed leaves all at once for efficiency.

(2) To defend against the selective deletion attack, we utilize a hierarchical storage structure with the same-sized levels for the unified management of outsourced encoded data and encoded update operations. According to this hierarchical structure and the bv23Tree, we resolve the open questions of [12] by transforming another secure public POR into the OPOR model and designing an appropriate dynamic scheme to efficiently integrate dynamic updates with OPOR.

(3) We analyze the security of our solution and conduct an extensive experimental study. The experimental results demonstrate the effectiveness of our scheme.

The rest of this paper is organized as follows: Section 2 states the background and introduces the architecture and system model of DOPOR. Section 3 shows the novel dynamic structure bv23Tree in detail, based on which we present detailed DOPOR solution in Section 4. Section 5 provides the security analysis, and Section 6 evaluates the experimental performance. Section 7 overviews the related work. Finally, this paper is concluded in Section 8.

2. Background and System Architecture

2.1. Problem Statement. We begin with the background of OPOR, as shown in [12]. There are three entities involved

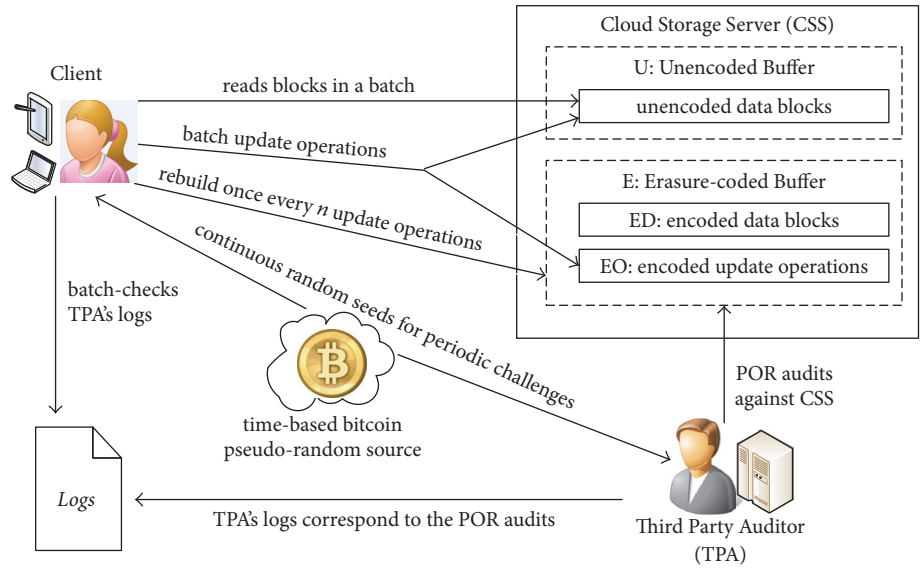


Figure 1: Dynamic OPOR (DOPOR) architecture.

in an outsourced auditing environment: mobile client (i.e., data owner), cloud storage server (CSS), and third-party auditor (TPA). Because of their limited local storage capacity, mobile clients are motivated to outsource their large data files to CSS and then can make use of various on-demand cloud storage services. However, because CSS might be misbehaving, it is very important to design the periodic remote data auditing mechanism against CSS, which enables the mobile client to have the assurance that her outsourced data is always available and can be completely retrieved from CSS if necessary. Further, to liberate the mobile client from struggling with the endless online data auditing, OPOR also introduces TPA that has the expertise and ability to perform the above frequent auditing tasks on behalf of the mobile client, and then the mobile client can be offline to rest most of the time.

Although OPOR model has the same three entities as existing public auditing model [9, 10, 15, 16], one of the main differences between the two is that *TPA might be also malicious within OPOR* [12]. In other words, TPA might violate the auditing protocols, for example, by claiming that he has honestly performed all past auditing, but actually he tells a lie. *Furthermore, any two entities might be in collusion within OPOR* [12]. For example, firstly, malicious CSS might collude with malicious TPA to deceive the honest client when outsourced data has been lost. Secondly, malicious client might collude with malicious CSS to frame the honest TPA, by asserting that TPA did not correctly perform the required auditing work to bring TPA into the compensation lawsuit. Since any entity might be malicious, in order to solve the problem of securely sampling the periodic challenges for frequent auditing, the time-based bitcoin pseudorandom source is introduced into the OPOR model. As demonstrated in [12], due to the fact that the bitcoin pseudorandom source cannot be manipulated by any entity, it is secure to be used for generating the continuous random seeds for periodic challenges.

OPOR inherits the retrievability guarantee of POR [8] by applying erasure code, meaning that the client can retrieve the whole outsourced data in case of minor data corruption. However, when both of the data dynamics and erasure code are considered, the problem of how to efficiently perform the updates is intractable. Within the first OPOR scheme Fortress [12], client's original data file (including n raw data blocks) is entirely encoded before outsourcing, so an update operation upon any single raw data block will affect the whole outsourced encoded data. In this case, the only way for client is to download and decode the whole outsourced encoded data and then encode and upload all the data again after performing the updates, which means unbearable bandwidth and computation costs. This is why Fortress is just a static scheme that cannot support efficient dynamic updates.

2.2. Dynamic OPOR (DOPOR) Architecture. The representative DOPOR architecture is presented in Figure 1. Based on the bitcoin source that controls the random sampling of periodic challenges, once TPA accepts the migrated auditing tasks from the client, TPA must generate the corresponding logs after he completes each specified POR audit against CSS. In this case, the client is able to check TPA's work at any point in time by verifying TPA's logs, and then she can judge that if TPA did his auditing work correctly in the past. As shown in [12], such client's checking against TPA can be much less frequent than the TPA's POR audits against CSS, since the client can batch-check a number of accumulated TPA logs all at once.

To support dynamic updates, we apply a similar idea to [14] within our solution, which is to place all accumulated update operations into an erasure-coded buffer at CSS side, rather than immediately executing these update operations upon the outsourced encoded data. As shown in Figure 1, the CSS-side storage is organized in two different buffers

denoted with U (i.e., unencoded buffer) and E (i.e., erasure-coded buffer). Buffer U will independently store an up-to-date copy of all the raw data blocks, which are organized by our proposed bv23Tree, to support the efficient batch reads from cloud storage without struggling with the erasure code. On the other side, buffer E is further divided into two parts, ED and EO, which store the whole outsourced original encoded data blocks and all the accumulated encoded update operations, respectively. In case of data loss, client can recover the up-to-date copy of the whole outsourced raw data by decoding the entire buffer E and combining both of ED and EO, so the periodic POR audits only need to be performed upon buffer E for the retrievability guarantee.

As will be described in Section 4.1, both ED and EO constitute a complete hierarchical storage structure where all levels have the same size, which are different from the levels of exponentially growing capacity in [14]. After the client performs a batch of update operations upon buffer U, benefitting from the same-sized levels of DOPOR, this batch of update operations can be wholly encoded and then directly placed into EO to fill up the corresponding level for improved computation cost, without executing the rebuilding of a level as in [14] that incurs $O(\log n)$ amortized cost for each update operation. More importantly, based on the same-sized levels, our DOPOR solution can build upon the public verification POR scheme of [8], the aggregation technique of which provides the support for the client's checking against malicious TPA.

Finally, after every n update operations, when the size of EO grows to the same size as ED, buffer E will be rebuilt. The rebuilding of E will rewrite the whole ED with the encoded version of all the up-to-date raw data blocks and meanwhile empty the whole EO. Since E is only rebuilt once in every n update operations, the amortized complexity of rebuilding E will be $O(1)$ per update operation. However, different from the existing schemes [14, 18, 19] that require $O(n)$ client-side temporary memory for such rebuilding, as shown in Section 4.3, based on the ability of batch reads and the same-sized levels, the client of DOPOR only requires $O(\lambda)$ client-side memory to gradually rebuild E, where λ is the security parameter that is independent of the data size n. So, DOPOR further reduces the required client's memory when rebuilding and thus is suitable for the client-side mobile devices.

2.3. System Model. Formally, the complete definition of DOPOR system can be described by the following ten protocols:

(i) **GenKey**$(1^{\mathscr{k}}) \rightarrow \{\mathrm{sk}_{\mathbb{E}}, \mathrm{pk}_{\mathbb{E}}\}$: when inputting the security parameter $\mathscr{k}$, this protocol outputs a pair of public-private keys for each entity $\mathbb{E} \in \{\text{client}, \text{CSS}, \text{TPA}\}$.

(ii) **GenTags**$(\mathrm{sk}_{\text{client}}, F) \rightarrow \{B, \Phi\}$: when inputting the client's secret key $\mathrm{sk}_{\text{client}}$ and the original file F that is an ordered set of raw data blocks $\{m_i\}$, this protocol encodes F into the encoded file B and outputs B as an ordered set of codeword blocks $\{b_i\}$. It also outputs the tags set $\Phi = \{\sigma_i\}$, where each σ_i is computed based on $\mathrm{sk}_{\text{client}}$ and b_i.

(iii) **OutsourceData**$(F, B, \Phi) \rightarrow \{\Psi, \mathbb{P}\}$: when inputting the original file F, the encoded file B, and tags Φ, it outputs the bv23Tree Ψ that is constructed based on F. After outsourcing all the input data and Ψ to CSS, this protocol also outputs the public parameters set $\mathbb{P}$ for POR audits.

(iv) **ReadBlocks**$(x_{\text{root}}, \Pi, \mathscr{st}) \rightarrow \{\mathscr{M}, \text{false}\}$: when inputting the root hash x_{root} of the bv23Tree, the set of any k blocks indices $\Pi = \{a_1, a_2, \ldots, a_k\}$, and the CSS state $\mathscr{st}$, this protocol outputs the appointed data blocks set $\mathscr{M} = \{m_{a_1}, m_{a_2}, \ldots, m_{a_k}\}$, or false otherwise.

(v) **PerformUpdates**$(x_{\text{root}}, \mathcal{SO}, \mathrm{sk}_{\text{client}}, \mathscr{st}, \mathscr{p}) \rightarrow \{(x_{\text{root}^*}, \mathscr{st}^*, \mathscr{p}^*), \text{false}\}$: when inputting the tree root hash x_{root}, the set of update operations $\mathcal{SO}$, client's secret key $\mathrm{sk}_{\text{client}}$, the CSS state $\mathscr{st}$, and the state pointer $\mathscr{p}$ that is related to $\mathscr{st}$, it outputs a new root hash x_{root^*}, a new CSS state $\mathscr{st}^*$, and a new pointer $\mathscr{p}^*$ showing that all the operations in $\mathcal{SO}$ are correctly executed in a batch, or false otherwise.

(vi) **Rebuild**$(x_{\text{root}}, \mathrm{sk}_{\text{client}}, \mathscr{st}, \mathscr{p}) \rightarrow \{(\mathscr{st}^*, \mathscr{p}^*), \text{false}\}$: when inputting the tree root hash x_{root}, client's secret key $\mathrm{sk}_{\text{client}}$, CSS state $\mathscr{st}$, and state pointer $\mathscr{p}$, it outputs a new CSS state $\mathscr{st}^*$ and a new pointer $\mathscr{p}^*$ showing that the rebuilding is completed, or false otherwise.

(vii) **RandomChal**$(\mathbb{B}, t, \mathbb{P}) \rightarrow \{\mathrm{Q}^{(t)}\}$: when inputting the bitcoin source $\mathbb{B}$, the time t, and the public parameters $\mathbb{P}$, it outputs a bitcoin-based challenge $Q^{(t)}$ for POR audit.

(viii) **GenProof**$(Q^{(t)}, \mathscr{st}) \rightarrow \{\rho^{(t)}\}$: when inputting the challenge $Q^{(t)}$ and the CSS state $\mathscr{st}$, it outputs CSS's proof $\rho^{(t)}$ to enable TPA to perform a POR audit.

(ix) **PORAudit**$(\mathbb{P}, Q^{(t)}, \rho^{(t)}) \rightarrow \{\Lambda^{(t)}, \mathfrak{D}_{\text{TPA}}\}$: when inputting public parameters set $\mathbb{P}$, challenge $Q^{(t)}$, and CSS's proof $\rho^{(t)}$, it outputs TPA's log $\Lambda^{(t)}$ and decision $\mathfrak{D}_{\text{TPA}}$. $\mathfrak{D}_{\text{TPA}}$ is true if TPA audit passes, or false otherwise.

(x) **CheckLogs**$(\mathbb{B}, T, \Lambda^{(T)}, \mathbb{P}, \mathrm{sk}_{\text{client}}) \rightarrow \{\mathfrak{D}_{\text{client}}\}$: when inputting the bitcoin source $\mathbb{B}$, a point-in-time set $T = \{t_1, t_2, \ldots, t_k\}$, the corresponding TPA's logs set $\Lambda^{(T)} = \{\Lambda^{(t)}\}_{t \in T}$, the public parameters $\mathbb{P}$, and client's secret key $\mathrm{sk}_{\text{client}}$, this protocol outputs a client's decision $\mathfrak{D}_{\text{client}}$. $\mathfrak{D}_{\text{TPA}}$ is true if client's batch-checking upon $\Lambda^{(T)}$ can pass, or false otherwise.

3. Balanced Authenticated Data Structure

Within the rb23Tree of [6], for verifying a leaf, the client must retrieve from the adversary a corresponding proof path that consists of $O(\log n)$ *marks*. However, the limitation of rb23Tree method is that a lot of duplicated information will exist in the proof paths of different leaves, which will waste too much communication cost when the client verifies many leaves one after another by retrieving different proof paths. To solve this problem, we propose the *batch-verifications 2-3 Tree*, called *bv23Tree*, for efficiency.

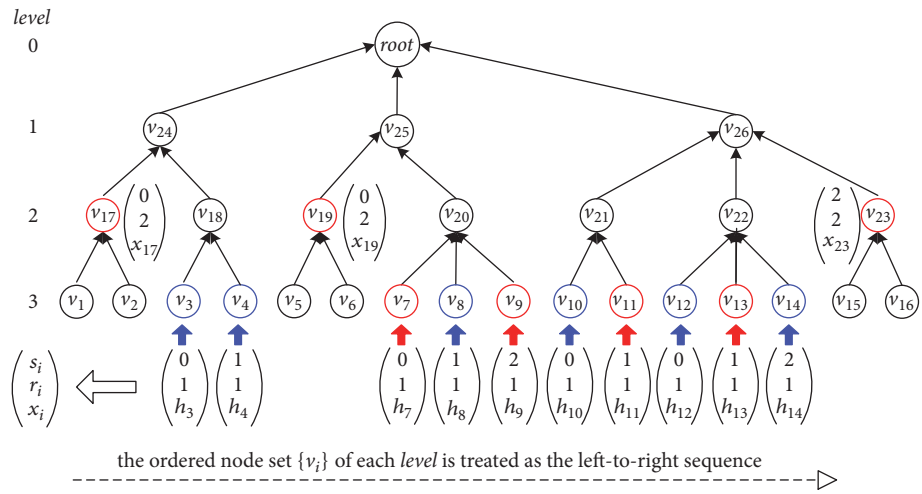


FIGURE 2: An example of bv23Tree. Besides the verified ordered leaves set $\mathscr{L} = \{v_3, v_4, v_8, v_{10}, v_{12}, v_{14}\}$, the verifier only needs to retrieve the necessary auxiliary tree nodes (colored in red), which are organized into a special table structure as will be shown later. Then, the verifier can batch-verify the indices and values of these leaves in $\mathscr{L}$ all together.

3.1. Batch-Verifications 2-3 Tree. Following the definition of 2-3 Tree [20], each nonleaf node of bv23Tree can have two or three children. Let F be an original data file consisting of n raw blocks $F = \{m_1, m_2, \ldots, m_n\}$. With the hash function $h(\cdot)$, the bv23Tree on file$F$ can be constructed by storing at each tree node v a 3-element tuple (s_v, r_v, x_v), defined as follows:

(i) s_v is the *status* of node v. Let $\mathscr{P}$ denote the parent node of v. Let $\mathrm{ch}_1, \mathrm{ch}_2, \mathrm{ch}_3$ be the three left-to-right children of $\mathscr{P}$, respectively. Specifically, if $\mathscr{P}$ only has two children, then $\mathrm{ch}_3 = \text{null}$. So, s_v is defined as

$$s_v = \begin{cases} 0, & \text{if } v \text{ is } \mathrm{ch}_1; \\ 1, & \text{if } v \text{ is } \mathrm{ch}_2; \\ 2, & \text{if } \mathrm{ch}_3 \neq \text{null}, \ v \text{ is } \mathrm{ch}_3. \end{cases} \quad (1)$$

(ii) r_v is the *rank value* of node v, which is similar to the concept defined in existing rank-based MHT scheme [16]. Namely, r_v stores the number of leaf nodes that belong to the subtree with node v as the root. If v is a leaf node, we define $r_v = 1$.

(iii) x_v represents the *authentication hash value* of node v. The value of r_v is defined with different cases.

Case 0. $v = \text{null}$; then

$$x_v = \text{null}. \quad (2)$$

Case 1. v is the ith leaf node; then

$$x_v = h(m_i). \quad (3)$$

Case 2. v is a nonleaf node with children $\mathrm{ch}_1, \mathrm{ch}_2$, and ch_3 as above (sometimes ch_3 will be null):

$$x_v = h\left(r_v \parallel x_{\mathrm{ch}_1} \parallel x_{\mathrm{ch}_2} \parallel x_{\mathrm{ch}_3}\right), \quad (4)$$

where $\parallel$ denotes the concatenation operation.

We show an example of bv23Tree in Figure 2, which is constructed on 16 file blocks $\{m_i\}_{1 \le i \le 16}$. Next, we use the concise shorthand for some symbols. Given a bv23Tree node v_i, we use s_i, r_i, and x_i to denote s_{v_i}, r_{v_i}, and x_{v_i}, respectively, so each tree node v_i and its corresponding 3-element tuple (s_i, r_i, x_i) are not distinguished. And we also use h_i to denote the hash value $h(m_i)$ for convenience.

3.2. Batch Queries. The integrity (i.e., authenticity and freshness) of file blocks can be protected by the corresponding hash values h_i $(1 \le i \le n)$ stored in leaves, while the integrity of the leaves themselves will be protected by bv23Tree. Now, suppose that the whole file blocks $\{m_1, m_2, \ldots, m_n\}$ and a bv23Tree on all blocks have been stored at CSS. Client wants to verify the integrity of any k ordered file blocks $m_{a_1}, m_{a_2}, \ldots, m_{a_k}$ $(1 \le a_1 < a_2 < \cdots < a_k \le n)$ read from CSS and thus batch-queries CSS by issuing the ordered indices set $\Pi = \{a_1, a_2, \ldots, a_k\}$ that appoints k ordered leaves $\{v_{a_1}, v_{a_2}, \ldots, v_{a_k}\}$. Then, CSS calls Algorithm 1 to generate the corresponding *proof table* and responds to client with the k appointed leaves and their proof table.

An example of proof table is shown in Table 1. Without loss of generality, suppose the number of the levels of a bv23Tree is l. Due to the property of balanced tree, the number l must be $O(\log n)$ complexity. Let $T_{k \times l}$ denote the proof table of any k appointed leaves; then $T_{k \times l}$ has the following characteristics:

(i) $T_{k \times l}$ contains k rows and l columns, respectively (i.e., $T_{k \times l}$ consists of $k \times l$ items).

(ii) Each item $T_{i,j}$ $(1 \le i \le k, 1 \le j \le l)$ can have one or two components, and each component can be a tree node v_i or a mark θ $(2 \le \theta \le k)$ or null. Since θ only denotes a pointer that points to the θth row of the table $T_{k \times l}$ itself, the communication cost of a mark θ is little when compared to the cost of a node v_i.

(iii) The more the leaves are batch-verified, the more the null that exists in $T_{k \times l}$. And no matter how many

Proof_Table$(\Psi, \Pi) \rightarrow T_{k\times l}$. With the bv23Tree Ψ of l levels, and the k ordered indices set $\Pi = \{a_1, a_2, \ldots, a_k\}$ that appoint k ordered leaves $\{v_{a_1}, v_{a_2}, \ldots, v_{a_k}\}$, this algorithm generates the corresponding proof table $T_{k\times l}$.

(1) initialize node array $A[1 \cdots k]$ by $A[i] \leftarrow v_{a_i}$ $(i = 1, 2, \ldots, k)$;
(2) {$A[i]$ tracks the parent of the current node v as below}
(3) $j \leftarrow 1$; {j tracks the column of the proof table $T_{k\times l}$}
(4) **while** $j \leq l$ **do**
(5) **for** $i = 1, 2, \ldots, k$ **do**
(6) **if** $A[i]$ = null **then**
(7) $T_{i,j} \leftarrow$ null;
(8) **else** {$A[i] \neq$ null}
(9) {let v denote the current tree node stored in $A[i]$}
(10) **if** node v only has one sibling node, which is denoted by sv **then**
(11) **if** sv exists in $A[\theta]$, in this case $i < \theta \leq k$ **then**
(12) $T_{i,j} \leftarrow \theta$; $A[\theta] \leftarrow$ null;
(13) **else** {sv does not exist in current array $A[\ ]$}
(14) $T_{i,j} \leftarrow sv$;
(15) **end if**
(16) **else** {node v has two sibling nodes, which are denoted by (sv_1, sv_2)}
(17) {note that the sequence of (sv_1, sv_2) must follow the left-to-right principle as in Figure 2, e.g., two siblings of node v_8 must be denoted by (v_7, v_9)}
(18) **if** only sv_1 (or sv_2) exists in $A[\theta]$ **then**
(19) $T_{i,j} \leftarrow (\theta, sv_2)$ (or $T_{i,j} \leftarrow (sv_1, \theta)$);
(20) $A[\theta] \leftarrow$ null;
(21) **else if** both sv_1 and sv_2 exist in $A[\theta_1]$ and $A[\theta_2]$, respectively, **then**
(22) $T_{i,j} \leftarrow (\theta_1, \theta_2)$; $A[\theta_1] \leftarrow$ null; $A[\theta_2] \leftarrow$ null;
(23) **else** both sv_1 and sv_2 do not exist in current array $A[\ ]$
(24) $T_{i,j} \leftarrow (sv_1, sv_2)$;
(25) **end if**
(26) **end if**
(27) $A[i] \leftarrow$ the parent node of the current node v;
(28) **end if**
(29) **end for**
(30) $j \leftarrow j + 1$;
(31) **end while**
(32) **return** the proof table $T_{k\times l}$;

Algorithm 1: Algorithm for generating proof table $T_{k\times l}$.

Table 1: An example of proof table.

Table item $T_{i,j}$	$j = 1$	$j = 2$	$j = 3$
$i = 1$ (row 1)	2	v_{17}	(3, 4)
$i = 2$ (row 2)	Null	Null	Null
$i = 3$ (row 3)	(v_7, v_9)	v_{19}	Null
$i = 4$ (row 4)	v_{11}	$(5, v_{23})$	Null
$i = 5$ (row 5)	$(v_{13}, 6)$	Null	Null
$i = 6$ (row 6)	Null	Null	Null

The corresponding proof table $T_{6\times 3}$ is generated based on the bv23Tree and the appointed ordered leaves set $\mathscr{L} = \{v_3, v_4, v_8, v_{10}, v_{12}, v_{14}\}$ of Figure 2.

leaves are batch-verified, each necessary auxiliary tree node v_i only appears once in $T_{k\times l}$.

In the context of the batch verifications upon k appointed leaves, compared to the proof path method as in [6], the communication cost of the proof table $T_{k\times l}$ is much less than that of transferring different proof paths of k leaves, respectively. This is because the proof table avoids the limitation of proof paths that the repetitive node information (e.g., node hash values) will exhibit in the proof paths of different leaves with high probability, so there is a plenty of null in the proof table to save the communication cost. Furthermore, compared to the 8-element tuple *mark* in proof path of [6], each tree node v_i included in the proof table is only related to a 3-element tuple, which further reduces the communication cost.

3.3. Batch Verifications. Upon receiving from CSS the k appointed leaves set $\mathscr{L} = \{v_{a_1}, v_{a_2}, \ldots, v_{a_k}\}$ and the required proof table $T_{k\times l}$, client can run Algorithm 2 to batch-verify the indices and values of these k leaves in $\mathscr{L}$ all at once, by using her local metadata x_{root}. An example of batch verifications upon multiple leaves is shown in Table 2.

Within Algorithm 2, the function Append(·) is to merge two different sets of numbers while preserving the order of these numbers. For example, given two sets $\mathscr{IS}_1 = \{4\}$ and $\mathscr{IS}_2 = \{6, 8, 10\}$, then Append$(\mathscr{IS}_1, \mathscr{IS}_2) = \{4, 6, 8, 10\}$. In addition, the operator notation "$\oplus$" is to add a number to every element of a set. For example, let $\mathscr{IS}_3 = \{4, 6, 8, 10\}$;

Batch_Verify($x_{\text{root}}, \Pi, \mathscr{L}, T_{k\times l}$) $\rightarrow$ {true, false}. This algorithm can batch-verify not only the hash values of all k ordered leaves $\mathscr{L} = \{v_{a_1}, v_{a_2}, \ldots, v_{a_k}\}$ provided by CSS, but also that the indices of these k leaves are exactly matched with the appointed indices set $\Pi = \{a_1, a_2, \ldots, a_k\}$.

(1) **for** each leaf $v_{a_i} \coloneqq (s_{a_i}, r_{a_i}, x_{a_i})$, $i = 1, 2, \ldots, k$, **do**
(2) initialize $\mathscr{R}_i \leftarrow r_{a_i}$; $\mathscr{H}_i \leftarrow x_{a_i}$; $\mathscr{IS}_i \leftarrow \{1\}$;
(3) **end for**
(4) initialize *Row_Set* $\leftarrow \{1, 2, \ldots, k\}$;
(5) **for** $j = 1, 2, \ldots, l$ **do**
(6) **for** each i in *Row_Set* **do**
(7) **if** item $T_{i,j}$ only has one component $\mathscr{C}$ **then**
(8) **call** *Deal_One_Component* ($\mathscr{C}$);
(9) **else** {$T_{i,j}$ has two components ($\mathscr{C}_1, \mathscr{C}_2$)}
(10) **call** *Deal_Two_Components* ($\mathscr{C}_1, \mathscr{C}_2$);
(11) **end if**
(12) **end for**
(13) **end for**
(14) **if** $\mathscr{H}_1 = x_{\text{root}}$ and $\mathscr{IS}_1 = \{a_1, a_2, \ldots, a_k\}$ **then**
(15) **return** true;
(16) **else**
(17) **return** false;
(18) **end if**

Function *Deal_One_Component* ($\mathscr{C}$)

(1) **if** $\mathscr{C}$ is null **then**
(2) **continue**;
(3) **else if** $\mathscr{C}$ is a node $v_t \coloneqq (s_t, r_t, x_t)$ **then**
(4) $\mathscr{R}_i \leftarrow \mathscr{R}_i + r_t$;
(5) **if** $s_t = 0$ **then**
(6) $\mathscr{IS}_i \leftarrow \mathscr{IS}_i \oplus r_t$;
(7) $\mathscr{H}_i \leftarrow h(\mathscr{R}_i \parallel x_t \parallel \mathscr{H}_i)$;
(8) **else** {$s_t = 1$}
(9) $\mathscr{H}_i \leftarrow h(\mathscr{R}_i \parallel \mathscr{H}_i \parallel x_t)$;
(10) **end** if
(11) **else** {$\mathscr{C}$ is a mark θ}
(12) $\mathscr{IS}_\theta \leftarrow \mathscr{IS}_\theta \oplus \mathscr{R}_i$;
(13) $\mathscr{R}_i \leftarrow \mathscr{R}_i + \mathscr{R}_\theta$;
(14) $\mathscr{H}_i \leftarrow h(\mathscr{R}_i \parallel \mathscr{H}_i \parallel \mathscr{H}_\theta)$;
(15) $\mathscr{IS}_i \leftarrow \text{Append}(\mathscr{IS}_i, \mathscr{IS}_\theta)$;
(16) **delete** θ from the *Row_Set*;
(17) **end if**

Function *Deal_Two_Components* ($\mathscr{C}_1, \mathscr{C}_2$)

(1) **if** $\mathscr{C}_1$ and $\mathscr{C}_2$ are two nodes v_t and v_τ, respectively, **then**
(2) {note that $v_t \coloneqq (s_t, r_t, x_t)$, $v_\tau \coloneqq (s_\tau, r_\tau, x_\tau)$}
(3) assign $\mathbb{S} \leftarrow \mathbb{S} \in \{0, 1, 2\}$ and $\mathbb{S} \notin \{s_t, s_\tau\}$;
(4) $\mathscr{R}_i \leftarrow \mathscr{R}_i + r_t + r_\tau$;
(5) **if** $\mathbb{S} = 0$ **then**
(6) $\mathscr{H}_i \leftarrow h(\mathscr{R}_i \parallel \mathscr{H}_i \parallel x_t \parallel x_\tau)$;
(7) **else if** $\mathbb{S} = 1$ **then**
(8) $\mathscr{IS}_i \leftarrow \mathscr{IS}_i \oplus r_t$; $\mathscr{H}_i \leftarrow h(\mathscr{R}_i \parallel x_t \parallel \mathscr{H}_i \parallel x_\tau)$;
(9) **else** {$\mathbb{S} = 2$}
(10) $\mathscr{IS}_i \leftarrow \mathscr{IS}_i \oplus (r_t + r_\tau)$; $\mathscr{H}_i \leftarrow h(\mathscr{R}_i \parallel x_t \parallel x_\tau \parallel \mathscr{H}_i)$;
(11) **end if**
(12) **else if** $\mathscr{C}_1$ is a node v_t, and $\mathscr{C}_2$ is a mark θ, **then**
(13) {in terms of the left-to-right principle, the value of s_t can only be 0 or 1 in this case}
(14) **if** $s_t = 0$ **then**
(15) $\mathscr{IS}_\theta \leftarrow \mathscr{IS}_\theta \oplus \mathscr{R}_i$;
(16) $\mathscr{IS}_i \leftarrow \text{Append}(\mathscr{IS}_i, \mathscr{IS}_\theta)$;
(17) $\mathscr{IS}_i \leftarrow \mathscr{IS}_i \oplus r_t$;
(18) $\mathscr{R}_i \leftarrow \mathscr{R}_i + r_t + \mathscr{R}_\theta$;
(19) $\mathscr{H}_i \leftarrow h(\mathscr{R}_i \parallel x_t \parallel \mathscr{H}_i \parallel \mathscr{H}_\theta)$;
(20) **else** {$s_t = 1$}

Algorithm 2: Continued.

(21) $\mathcal{IS}_\theta \leftarrow \mathcal{IS}_\theta \oplus (\mathcal{R}_i + r_\ell)$;
(22) $\mathcal{IS}_i \leftarrow \text{Append}(\mathcal{IS}_i, \mathcal{IS}_\theta)$;
(23) $\mathcal{R}_i \leftarrow \mathcal{R}_i + r_\ell + \mathcal{R}_\theta$;
(24) $\mathcal{H}_i \leftarrow h(\mathcal{R}_i \parallel \mathcal{H}_i \parallel x_\ell \parallel \mathcal{H}_\theta)$
(25) **end if**
(26) delete θ from the *Row_Set*;
(27) **else if** $\mathcal{C}_1$ is a mark θ, and $\mathcal{C}_2$ is a node v_ℓ, **then**
(28) $\mathcal{IS}_\theta \leftarrow \mathcal{IS}_\theta \oplus \mathcal{R}_i$;
(29) $\mathcal{IS}_i \leftarrow \text{Append}(\mathcal{IS}_i, \mathcal{IS}_\theta)$;
(30) $\mathcal{R}_i \leftarrow \mathcal{R}_i + \mathcal{R}_\theta + r_\ell$;
(31) $\mathcal{H}_i \leftarrow h(\mathcal{R}_i \parallel \mathcal{H}_i \parallel \mathcal{H}_\theta \parallel x_\ell)$;
(32) delete θ from the *Row_Set*;
(33) **else** {$\mathcal{C}_1$ and $\mathcal{C}_2$ are two marks θ_1 and θ_2, respectively}
(34) $\mathcal{IS}_{\theta_2} \leftarrow \mathcal{IS}_{\theta_2} \oplus \mathcal{R}_{\theta_1}$;
(35) $\mathcal{IS}_{\theta_1} \leftarrow \text{Append}(\mathcal{IS}_{\theta_1}, \mathcal{IS}_{\theta_2})$
(36) $\mathcal{IS}_{\theta_1} \leftarrow \mathcal{IS}_{\theta_1} \oplus \mathcal{R}_i$;
(37) $\mathcal{IS}_i \leftarrow \text{Append}(\mathcal{IS}_i, \mathcal{IS}_{\theta_1})$;
(38) $\mathcal{R}_i \leftarrow \mathcal{R}_i + \mathcal{R}_{\theta_1} + \mathcal{R}_{\theta_2}$;
(39) $\mathcal{H}_i \leftarrow h(\mathcal{R}_i \parallel \mathcal{H}_i \parallel \mathcal{H}_{\theta_1} \parallel \mathcal{H}_{\theta_2})$;
(40) delete θ_1 and θ_2 from the *Row_Set*;
(41) **end if**

Algorithm 2: Algorithm for batch-verifying all appointed leaves in $\mathcal{L}$.

Table 2: An example of batch verifications.

Variables	Initialization	$j = 1$	$j = 2$	$j = 3$
$\mathcal{H}_1$	x_3	x_{18}	x_{24}	x_{root}
$\mathcal{IS}_1$	{1}	{1, 2}	{3, 4}	{3, 4, 8, 10, 12, 14}
$\mathcal{H}_2$		x_4		
$\mathcal{IS}_2$	{1}		{2}	
$\mathcal{H}_3$	x_8	x_{20}	x_{25}	
$\mathcal{IS}_3$	{1}	{2}	{4}	{8, 10, 12, 14}
$\mathcal{H}_4$	x_{10}	x_{21}	x_{26}	
$\mathcal{IS}_4$	{1}	{1}	{1, 3, 5}	{6, 8, 10}
$\mathcal{H}_5$	x_{12}		x_{22}	
$\mathcal{IS}_5$	{1}	{1, 3}	{3, 5}	
$\mathcal{H}_6$		x_{14}		
$\mathcal{IS}_6$	{1}		{3}	

The process of batch verifications upon the appointed ordered leaves set $\mathcal{L} = \{v_3, v_4, v_8, v_{10}, v_{12}, v_{14}\}$ of Figure 2, according to Algorithm 2 and the proof table $T_{6\times3}$ of Table 1.

then $\mathcal{IS}_3 \oplus 4 = \{8, 10, 12, 14\}$. Algorithm 2 applies each nonnull item $T_{i,j}$ of $T_{k\times l}$ to iteratively compute tuple ($\mathcal{R}_i$, $\mathcal{H}_i$, $\mathcal{IS}_i$). If the returned leaves set $\mathcal{L}$ and table $T_{k\times l}$ are right, we will get the following results after the outermost for-loop of Algorithm 2 is finished:

(i) Value $\mathcal{H}_1$ is equal to x_{root}, that is, the authentication hash value of the root of bv23Tree.

(ii) Value $\mathcal{IS}_1$ is exactly the same as $\{a_1, a_2, \ldots, a_k\}$, that is, the indices set of k appointed ordered leaves in $\mathcal{L}$.

At a high level, Algorithm 2 is also the way to gradually construct the *partial bv23Tree*, which precisely covers all appointed leaves in $\mathcal{L}$, the paths from these appointed leaves to the root, and all the siblings of the nodes on these paths. Based on this partial bv23Tree, the batch updates upon outsourced original raw blocks $\{m_1, m_2, \ldots, m_n\}$ can be supported, as shown in Algorithms 3 and 4.

3.4. Batch Updates. Three basic types of dynamic update operations are modification (M), insertion (I), and deletion (D) [9]. Any block-level update operation $\mathcal{O}$ can be defined by the form of $\mathcal{O} :=$ "$\mathcal{U}$ a m", where $\mathcal{U} \in \{M, I, D\}$ denotes operation type, a is the index of targeted block, and m is the new data block that will be exactly stored according to the targeted index a (m is null for deletion). For example, "M 2 m" is to modify the 2nd block to m, "I 3 m" is to insert m after the 3rd block, and "D 4 null" is to delete the 4th block.

In the setting of dynamism, the update operations should be performed not only on the data blocks, but also on the bv23Tree. Note that only the insertion and deletion can

Batch_Updates $(\mathcal{SO}, \Psi, F) \rightarrow \{\mathcal{L}, T_{k\times l}, x_{\text{root}^*}\}$. Input parameters $\mathcal{SO} = \{\mathcal{O}_1, \mathcal{O}_2, \ldots, \mathcal{O}_c\}$ are a batch of update operations, Ψ is the bv23Tree, and $F = \{m_1, m_2, \ldots, m_n\}$ are the whole outsourced original blocks. This algorithm outputs an ordered leaves set $\mathcal{L}$, the proof table $T_{k\times l}$, and the updated root hash value x_{root^*} from the final state bv23Tree Ψ^*.

(1) extract from $\mathcal{SO}$ the largest ordered targeted indices set $\Pi = \{a_1, a_2, \ldots, a_k\}$, by removing the duplicate indices;
(2) read leaves set $\mathcal{L} = \{v_{a_1}, v_{a_2}, \ldots, v_{a_k}\}$ from Ψ;
(3) obtain $T_{k\times l} \leftarrow$ *Proof_Table*(Ψ, Π);
(4) update the file blocks set F according to the sequential executions of all update operations $\{\mathcal{O}_1, \mathcal{O}_2, \ldots, \mathcal{O}_c\}$;
(5) perform each of the update operations in sequence on Ψ and then obtain the final state Ψ^*; more specifically, transform Ψ in terms of each operation, and update the status s_v, rank r_v, and hash value x_v of the affected tree nodes during each transformation; (for the modification operation without transformation, only need to update the hash values of the nodes on the path from the targeted leaf to the root)
(6) **return** $\{\mathcal{L}, T_{k\times l}, x_{\text{root}^*}\}$, where x_{root^*} is the authentication hash value of the root of final state Ψ^*;

Algorithm 3: Algorithm for CSS to perform the batch updates.

Verify_Updates $(x_{\text{root}}, \mathcal{SO}, \mathcal{L}, T_{k\times l}, x_{\text{root}^*}) \rightarrow \{\text{true}, \text{false}\}$. Input parameters x_{root} is client local metadata, update operations set $\mathcal{SO} = \{\mathcal{O}_1, \mathcal{O}_2, \ldots, \mathcal{O}_c\}$ are generated by client herself, $\mathcal{L}$, $T_{k\times l}$, x_{root^*} are provided by CSS as computed in Algorithm 3. This algorithm outputs true if the batch updates are successful, or false otherwise.

(1) extract $\Pi = \{a_1, a_2, \ldots, a_k\}$ from $\mathcal{SO}$ as in Algorithm 3;
(2) **if** *Batch_Verify* $(x_{\text{root}}, \Pi, \mathcal{L}, T_{k\times l}) =$ true **then**
(3) construct partial bv23Tree with x_{root} as the root hash;
(4) **else** {Batch_Verify$(x_{\text{root}}, \Pi, \mathcal{L}, T_{k\times l}) =$ false}
(5) **return** false;
(6) **end if**
(7) perform each update operation $\mathcal{O}_i$ of $\mathcal{SO}$ upon above partial bv23Tree by the same transformations as in Algorithm 3, and then compute the final state root hash $x_{\text{root}'}$;
(8) **if** $x_{\text{root}'} = x_{\text{root}^*}$ **then**
(9) replace local x_{root} with x_{root^*}, **return** true;
(10) **else** $\{x_{\text{root}'} \neq x_{\text{root}^*}\}$
(11) **return** false;
(12) **end if**

Algorithm 4: Algorithm for client to verify the result of batch updates.

cause the structure transformation of bv23Tree, and the maintenance of this transformation is essentially identical to the maintenance of a standard 2-3 tree [20], except that the updating of each affected tree node v should be considered in terms of the 3-element tuple (s_v, r_v, x_v). As in Figure 3, we give an example for the structure transformation of bv23Tree after repetitively inserting (or deleting) the appointed data blocks at the same index position.

Now, suppose that client caches a batch of ordered update operations $\mathcal{SO} \coloneqq \{\mathcal{O}_1, \mathcal{O}_2, \ldots, \mathcal{O}_c\}$, which refer to an ordered indices set $\Pi = \{a_1, a_2, \ldots, a_k\}$, $k \leq c$, by the removal of duplicate indices. To batch-update the remote data, client first issues $\mathcal{SO}$ to CSS and obtains the returned ordered leaves set $\mathcal{L} = \{v_{a_1}, v_{a_2}, \ldots, v_{a_k}\}$ along with the proof table $T_{k\times l}$. As shown in Section 3.3, during the process of executing Algorithm 2 to batch-verify $\mathcal{L}$, client can construct the corresponding partial bv23Tree. Then, after sequentially performing each operation of $\mathcal{SO}$ upon the partial bv23Tree, client can compute by herself what would be the authentication hash value of the final state tree root. Finally, if CSS outputs the same final state root hash value as the one computed by client herself, client outputs true, meaning that CSS correctly performs the batch updates according to $\mathcal{SO}$. Otherwise, client outputs false. As shown in Algorithms 3 and 4, we outline the *Batch_Updates* algorithm performed by CSS and the *Verify_Updates* algorithm performed by client, respectively.

4. DOPOR Solution

4.1. Cloud Server Storage Configuration. As shown in Figure 1, there are two different buffers U and E at CSS side for outsourced data storage. Client's original file F, consisting of n raw blocks $F = \{m_1, m_2, \ldots, m_n\}$, will be separately stored into U and E with different formats, detailed as follows.

U (Unencoded Buffer). To support the efficient reads, buffer U always stores the up-to-date copy of the raw data blocks, and thus the update operations issued from client must be immediately performed upon the appointed blocks of U. All the blocks in U are organized by the bv23Tree as proposed in Section 3, the batch-verifications property of which enables client to batch-read a group of raw blocks from U for improved performance.

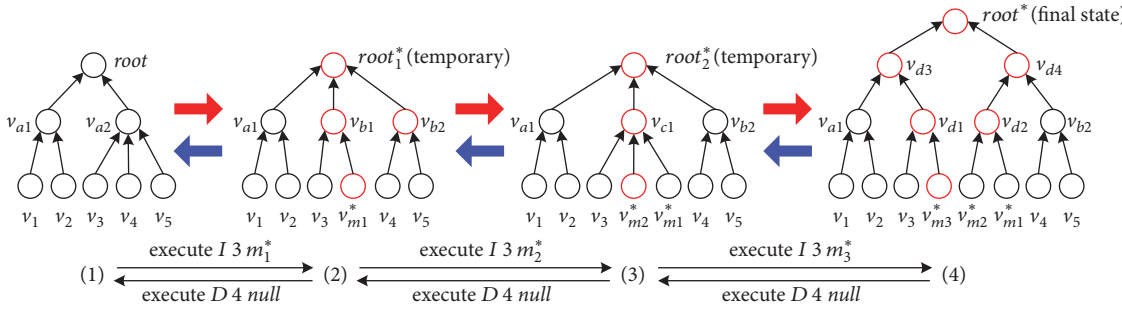


FIGURE 3: From (1) to (4) is the structure transformation of bv23Tree after performing three insertion operations $\{I\ 3\ m_1^*, I\ 3\ m_2^*, I\ 3\ m_3^*\}$ in order. And the reverse transformation from (4) to (1) is based on the sequential executions of three identical deletion operations.

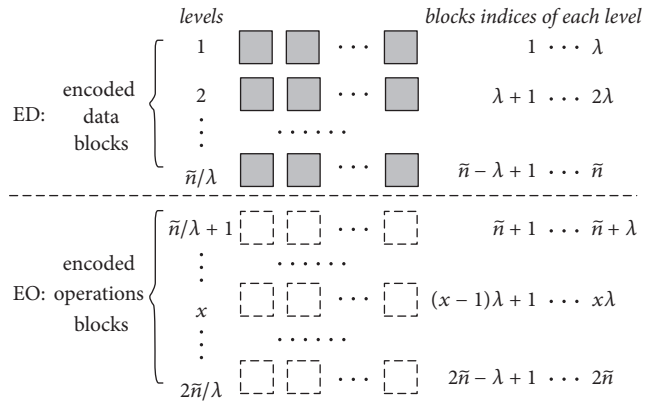


FIGURE 4: CSS-side hierarchical storage structure for erasure-coded buffer E, where each level stores λ encoded blocks along with tags.

E (Erasure-Coded Buffer). Buffer E is organized by the same-sized hierarchical structure, as in Figure 4, where each level has the same size to hold λ encoded blocks with the blocks tags (λ is the security parameter). Moreover, E is equally divided into two parts ED and EO with the same capacity. ED is to store encoded data blocks, and EO is to store encoded operations blocks. At the beginning, client applies erasure code to encode the entire file F into $\tilde{n}$ encoded data blocks, computes the blocks tags, and sequentially stores them into ED in terms of the blocks indices as shown in Figure 4. Subsequently, the content of ED will not be changed until buffer E is rebuilt. As shown in Section 3.4, each update operation $\mathcal{O}$ is of the form "$\mathcal{U}\ a\ m$", which can also be regarded as an operation block. In this case, after a batch of update operations $\mathcal{SO} = \{\mathcal{O}_1, \mathcal{O}_2, \ldots, \mathcal{O}_c\}$ ($c < \lambda$) are performed upon U, client will encode $\mathcal{SO}$ into λ encoded operations blocks and store them along with the blocks tags into a corresponding level of EO. So, EO is empty in the initial state and will be incrementally filled up level by level over time. At last, once EO is full, the rebuilding of E is triggered, as will be described later.

At a high level, by decoding the whole E and sequentially performing the accumulated update operations of EO upon the original data blocks of ED, the latest version of the whole client's outsourced raw data can be recovered. Therefore, the periodic POR audits only need to be deployed against the buffer E, which functions as a backup storage to provide the retrievability guarantee when data loss occurs. To resist the mentioned selective deletion attack as before, every POR audit will sample challenged blocks from each filled level of E. Because each level is entirely encoded, malicious CSS must corrupt a significant portion of the encoded blocks of one level to actually cause the data loss. However, if malicious CSS corrupts that many encoded blocks of one level, it cannot pass the POR audits with overwhelming probability.

4.2. Initialization. We also work in the bilinear setting, where G is a multiplicative cyclic group of prime order p and g is a generator of G. Let $e : G \times G \to G_T$ be the same nondegenerate bilinear map as in [8] that has the following property: for any $\varkappa, \omega \in G$ and $a, b \in \mathbb{Z}_p$, $e(\varkappa^a, \omega^b) = e(\varkappa, \omega)^{ab}$. Let $H : \{0,1\}^* \to G$ be the secure BLS hash function. The initialization of our scheme is described as follows.

(1) **GenKey**(1^{κ}): each entity $\mathbb{E} \in \{\text{client, CSS, TPA}\}$ generates a signing key pair ($\text{ssk}_{\mathbb{E}}$, $\text{spk}_{\mathbb{E}}$) for their respective signatures. In addition, client samples a random element $\alpha \xleftarrow{R} \mathbb{Z}_p$ and computes $\omega \leftarrow g^{\alpha}$. α is kept secret by client but ω is public. So, client's private key $\text{sk}_{\text{client}} = (\alpha, \text{ssk}_{\text{client}})$ and her public key $\text{pk}_{\text{client}} = (\omega, \text{spk}_{\text{client}})$.

(2) **GenTags**($\mathrm{sk}_{\mathrm{client}}, F$): client applies erasure code to encode $F = \{m_i\}_{1\le i\le n}$ into $\tilde{n}$ codeword blocks $B = \{b_i\}_{1\le i\le \tilde{n}}$, and each b_i is s sectors long: $b_i := \{b_{ij}\}_{1\le j\le s}$. Client then generates a name fid for F and samples s elements $u_1, \ldots, u_s \xleftarrow{R} G$. For each index i, $1 \le i \le \tilde{n}$, with her secret key α in $\mathrm{sk}_{\mathrm{client}}$, client computes for b_i the corresponding tag $\sigma_i \leftarrow (H(\mathrm{fid} \parallel i) \cdot \prod_{j=1}^{s} u_j^{b_{ij}})^{\alpha}$ and attaches σ_i to b_i.

(3) **OutsourceData**(F, B, Φ): based on $F = \{m_1, m_2, \ldots, m_n\}$, client generates the corresponding bv23Tree Ψ with the root hash x_{root}, as shown in Section 3.1. Then, client outsources $\{F, \Psi\}$ into the buffer U and outsources all encoded data blocks $B = \{b_i\}_{1\le i\le \tilde{n}}$ (along with their tags $\Phi = \{\sigma_i\}_{1\le i\le \tilde{n}}$) into the ED part of buffer E, the layout of which is shown in Figure 4. In addition, let $\mathfrak{sp}$ be a state pointer that denotes the number of the filled levels of buffer E. Clearly, the range of $\mathfrak{sp}$ is $\tilde{n}/\lambda \le \mathfrak{sp} \le 2\tilde{n}/\lambda$, as in Figure 4. Let $c\ell$ denote the number of the challenged blocks from a filled level, and let $\mathbb{P} = \{\mathfrak{sp}, c\ell, g, \omega, \mathrm{fid}, u_1, \ldots, u_s\}$ be the public parameters set for POR audits. Finally, client keeps $\mathbb{P}$ and x_{root} locally, sends $\mathbb{P}$ to TPA, and deletes $\{F, \Psi, B, \Phi\}$ from her local storage.

4.3. Data Access Mechanisms. Based on the bv23Tree of buffer U and the hierarchical configuration of same-sized levels of buffer E, DOPOR supports batch updates that enable the client to perform a batch of update operations upon the outsourced storage, which is suitable for the common scenario of [14] where writes are frequent. Now, after completing the initialization of DOPOR, the client can access and update her outsourced data by the following three protocols.

(1) **ReadBlocks**($x_{\mathrm{root}}, \Pi, \mathfrak{st}$). With her local root hash x_{root} of bv23Tree, the client can batch-read any k appointed raw blocks from CSS, by sending the ordered blocks indices set $\Pi = \{a_1, a_2, \ldots, a_k\}$ as the query to CSS. Here, let buffer U be the CSS state $\mathfrak{st}$. In terms of Section 3, upon receiving Π, CSS accesses appointed raw blocks $\mathscr{M} = \{m_{a_1}, m_{a_2}, \ldots, m_{a_k}\}$ and the tree leaves $\mathscr{L} = \{v_{a_1}, v_{a_2}, \ldots, v_{a_k}\}$ from U, generates $T_{k\times l} \leftarrow$*Proof_Table* (Ψ, Π), and returns $\{\mathscr{M}, \mathscr{L}, T_{k\times l}\}$ to the client. Then, the client batch-verifies the authenticity of $\mathscr{L}$ by calling *Batch_Verify* $(x_{\mathrm{root}}, \Pi, \mathscr{L}, T_{k\times l})$ and finally checks the integrity of all raw blocks of $\mathscr{M}$ according to the corresponding hash values stored in $\mathscr{L}$.

(2) **PerformUpdates**($x_{\mathrm{root}}, \mathcal{SO}, \mathrm{sk}_{\mathrm{client}}, \mathfrak{st}, \mathfrak{sp}$). Suppose that the client keeps $O(\lambda)$ local storage to cache c ($c < \lambda$) ordered update operations $\mathcal{SO} := \{\mathcal{O}_1, \mathcal{O}_2, \ldots, \mathcal{O}_c\}$. Then, the client sends $\mathcal{SO}$ to CSS for performing these c operations in a batch. As shown in Section 3.4, on receiving $\mathcal{SO}$, with the raw data F and bv23Tree Ψ stored in buffer U, CSS can execute *Batch_Updates* $(\mathcal{SO}, \Psi, F)$ to return to the client the results $\{\mathscr{L}, T_{k\times l}, x_{\mathrm{root}^*}\}$ for batch updates, and the client can call *Verify_Updates* $(x_{\mathrm{root}}, \mathcal{SO}, \mathscr{L}, T_{k\times l}, x_{\mathrm{root}^*})$ to authenticate these returned results.

If the above results pass the client's authentication, the client then applies an erasure code to encode $\mathcal{SO}$ into λ encoded operations blocks $\{b_{i_1}, b_{i_2}, \ldots, b_{i_\lambda}\}$. Based on local state pointer $\mathfrak{sp}$, client first computes the indices $\{i_1, i_2, \ldots, i_\lambda\}$ of these λ encoded blocks in order $i_\tau = \mathfrak{sp}\cdot\lambda + \tau$, $1 \le \tau \le \lambda$, and computes the tag σ_{i_τ} for each b_{i_τ} by the same tag formula as in **GenTags**$(\cdot)$ of initialization. Secondly, client outsources these λ blocks $\{b_{i_1}, b_{i_2}, \ldots, b_{i_\lambda}\}$ along with their tags into the $(\mathfrak{sp}+1)$th level of buffer E (i.e., the corresponding empty level of EO). Finally, client sends to TPA the updated state pointer $\mathfrak{sp}^* = \mathfrak{sp} + 1$ with her signature and empties her $O(\lambda)$ local storage for caching the next c ordered update operations. Overall, through this protocol, the CSS state $\mathfrak{st}$ consists of U and EO as above.

(3) **Rebuild**($x_{\mathrm{root}}, \mathrm{sk}_{\mathrm{client}}, \mathfrak{st}, \mathfrak{sp}$). Once in every n update operations, when EO of buffer E gets filled (i.e., $\mathfrak{sp}$ is equal to $2\tilde{n}/\lambda$ as in Figure 4), the periodic rebuilding for E is triggered. Since buffer U stores the up-to-date copy of all raw data blocks, client can carry out this rebuilding based on U for the performance improvement, instead of decoding the whole E and applying all operations of EO on the original data of ED. (ED and EO will not be decoded and combined, unless client detects data corruption within CSS and wants to retrieve the whole data.)

Benefitting from the ability of batch-reading multiple blocks from CSS and our hierarchical configuration, client can rebuild E with only $O(\lambda)$ local memory that is the same size as a level of the hierarchical structure, which is the significant improvement when compared to existing schemes [14, 18, 19] that require such rebuilding with $O(n)$ client local memory. In this protocol, the CSS state $\mathfrak{st}$ consists of U and E. At the beginning, client runs **ReadBlocks**$(\cdot)$ to batch-read the first c ($c < \lambda$) raw blocks from U. After encoding these c blocks into λ codeword blocks, client computes the corresponding λ tags by the same way as in **GenTags**$(\cdot)$, where the indices of λ codeword blocks of each level are shown in Figure 4. Then, client outsources to CSS these λ codeword blocks and tags that will be stored in the first level of ED. Subsequently, client can empty her $O(\lambda)$ local memory and batch-read from U the next batch of c blocks, which are processed and outsourced to CSS by the same procedures as above, except that each batch of λ processed codeword blocks and tags will be stored in the corresponding different level of ED (e.g., the second batch will be stored in the second level of ED, etc.). After the last batch of blocks from U are processed and outsourced, client authorizes CSS to update the whole ED with all above outsourced codeword blocks and tags, and CSS simultaneously empties the whole EO. Finally, client publishes the updated state pointer $\mathfrak{sp}^*$ to TPA, meaning that the rebuilding is over. Overall, the amortized bandwidth of rebuilding E is $O(1)$ per update operation, since this rebuilding is executed only once every n update operations.

4.4. Outsourced Proof of Retrievability (OPOR)

(1) **RandomChal**($\mathbb{B}, t, \mathbb{P}$). The periodic random challenges are generated based on the time-based bitcoin pseudorandom

source $\mathbb{B}$. More specifically, given the current time t, the tool getblockhash(t) [12] from the bitcoin source $\mathbb{B}$ can output the hash of the latest block that has arisen since time t in the bitcoin block chain. As shown in [12], for a future time t, no adversary can predict the hash of a bitcoin block that will arise in the future. In addition, for a past time t, the hash of previous bitcoin block, returned by getblockhash(t), is objective and irrefutable against any adversary. Thus, let $\text{coin}^{(t)}$ denote the output of getblockhash(t); then $\text{coin}^{(t)}$ can be considered as a secure pseudorandom coin for time t.

With the public parameter $\mathfrak{z}, \mathfrak{c} \in \mathbb{P}$, TPA can generate the random POR challenge of length $\mathfrak{z} \cdot \mathfrak{c}$, by calling the same probabilistic algorithm Sample($\text{coin}^{(t)}, \mathfrak{z} \cdot \mathfrak{c}$) as in [12] that $\text{coin}^{(t)}$ is obtained from getblockhash(t) for the different time t. Specifically, to ensure that a POR challenge can sample the same amount of $\mathfrak{c}$ blocks from each filled level of buffer E (i.e., ED and EO) at CSS, TPA first calls Sample($\text{coin}^{(t)}, \mathfrak{c}$) to choose a random $\mathfrak{c}$-elements subset $I^* = \{s_{t,1}, s_{t,2}, \ldots, s_{t,\mathfrak{c}}\}$ of $[1, \lambda]$, and then it computes the $\mathfrak{z} \cdot \mathfrak{c}$-elements set I as follows:

$$I = \left\{\zeta \cdot \lambda + \tau \mid 0 \le \zeta \le \mathfrak{z} - 1,\ \tau \in I^*\right\}, \tag{5}$$

Moreover, for each $i \in I$, as in [12], TPA also depends on Sample($\text{coin}^{(t)}, \mathfrak{z} \cdot \mathfrak{c}$) to choose a corresponding random element $\alpha_i \xleftarrow{R} \mathbb{Z}_p$. Finally, let $Q^{(t)} \coloneqq \{(i, \alpha_i)\}_{i \in I}$ denote an $\mathfrak{z} \cdot \mathfrak{c}$-elements set, which is regarded as the POR challenge at time t and sent to CSS by TPA.

(2) **GenProof**($Q^{(t)}, \mathfrak{z}t$). Here, let buffer E be the CSS state $\mathfrak{z}t$. Upon receiving challenge $Q^{(t)}$, for each index i specified by $Q^{(t)}$, CSS reads the corresponding block $b_i \coloneqq \{b_{ij}\}_{1 \le j \le s}$ and tag σ_i from E. Now CSS performs exactly as in the public verification scheme of [8] by computing the values $\mu_j^{(t)}$, $1 \le j \le s$, along with $\sigma^{(t)}$ as follows:

$$\begin{aligned} \mu_j^{(t)} &\longleftarrow \sum_{(i,\alpha_i) \in Q^{(t)}} \left(\alpha_i \cdot b_{ij}\right) \in \mathbb{Z}_p, \\ \sigma^{(t)} &\longleftarrow \prod_{(i,\alpha_i) \in Q^{(t)}} {\sigma_i}^{\alpha_i} \in G. \end{aligned} \tag{6}$$

At last, CSS sends to TPA its response $\rho^{(t)} \parallel \text{sig}_{\text{CSS}}$, where $\rho^{(t)} \coloneqq (\mu_1^{(t)}, \mu_2^{(t)}, \ldots, \mu_s^{(t)}, \sigma^{(t)})$, and sig_{CSS} is CSS's signature to provide nonrepudiation.

(3) **PORAudit**($\mathbb{P}, Q^{(t)}, \rho^{(t)}$). Based on the public parameters set $\mathbb{P}$ and the challenge $Q^{(t)}$, TPA can compute his own auditing parameter $\xi^{(t)}$ as follows:

$$\xi^{(t)} \longleftarrow \prod_{(i,\alpha_i) \in Q^{(t)}} H\left(\text{fid} \parallel i\right)^{\alpha_i} \in G. \tag{7}$$

Then, after TPA parses the CSS's response to obtain $\rho^{(t)}$, TPA will audit $\rho^{(t)}$ by checking the following equation:

$$e\left(\sigma^{(t)}, g\right) \stackrel{?}{=} e\left(\xi^{(t)} \cdot \prod_{j=1}^{s} {u_j}^{\mu_j^{(t)}}, \omega\right). \tag{8}$$

If this verification does not pass, TPA informs the client of this abnormal situation, meaning that the data loss occurs. Finally, TPA must generate and store the following log $\Lambda^{(t)}$ that corresponds to the challenge at time t:

$$\Lambda^{(t)} \coloneqq \left(t, \text{coin}^{(t)}, \xi^{(t)}, \rho^{(t)} \parallel \text{sig}_{\text{CSS}}\right). \tag{9}$$

(4) **CheckLogs**($\mathbb{B}, T, \Lambda^{(T)}, \mathbb{P}, \text{sk}_{\text{client}}$). To protect against malicious TPA who might violate the above auditing process or even collude with CSS, the client can verify TPA's work by checking TPA's logs. However, instead of checking the accumulated TPA's logs one by one, client is able to batch-check multiple TPA's logs all together, so such a client's batch-checking against TPA is only seldom performed in practice, as shown in [12].

Client can check the latest TPA's log for a minimal check, since this log reflects the latest status of retrievability for the outsourced data [12]. More generally, to perform a batch-checking against TPA, client selects a point-in-time set $T = \{t_1, t_2, \ldots, t_k\}$ and sends T to TPA, where each $t \in T$ marks the time of a past challenge.

Upon receiving T, for each $t \in T$, in terms of $\rho^{(t)}$ and $\xi^{(t)}$ stored in log $\Lambda^{(t)}$, TPA computes

$$\begin{aligned} \mu_j^{(T)} &\longleftarrow \sum_{t \in T} \mu_j^{(t)} \in \mathbb{Z}_p \quad \text{for } 1 \le j \le s, \\ \sigma^{(T)} &\longleftarrow \prod_{t \in T} \sigma^{(t)} \in G, \\ \xi^{(T)} &\longleftarrow \prod_{t \in T} \xi^{(t)}. \end{aligned} \tag{10}$$

Then, TPA responds to client with his proof $\varrho^{(T)} \coloneqq (\mu_1^{(T)}, \mu_2^{(T)}, \ldots, \mu_s^{(T)}, \sigma^{(T)}, \xi^{(T)})$, which is also signed by TPA.

Based on the public parameters $\mathfrak{z}$ and $\mathfrak{c}$, for each $t \in T$, the client is able to reconstruct alone each past challenge $Q^{(t)}$ as described in the protocol **RandomChal**($\cdot$), by using Sample($\text{coin}^{(t)}, \mathfrak{z} \cdot \mathfrak{c}$) with the pseudorandom $\text{coin}^{(t)}$ obtained from getblockhash(t). So, client can compute her own checking parameter $\eta^{(T)}$ as follows:

$$\eta^{(T)} \longleftarrow \prod_{t \in T} \prod_{(i,\alpha_i) \in Q^{(t)}} H\left(\text{fid} \parallel i\right)^{\alpha_i} \in G. \tag{11}$$

After verifying TPA's signature on the proof $\varrho^{(T)}$, client first checks that whether $\eta^{(T)}$ is equal to $\xi^{(T)}$ of $\varrho^{(T)}$. If $\eta^{(T)} \neq \xi^{(T)}$, client outputs false, confirming that TPA was irresponsible for the past POR audits. Finally, client checks the following equation with her secret key $\alpha \in \text{sk}_{\text{client}}$:

$$\left(\eta^{(T)} \cdot \prod_{j=1}^{s} {u_j}^{\mu_j^{(T)}}\right)^{\alpha} \stackrel{?}{=} \sigma^{(T)}. \tag{12}$$

If the above client's check fails, client outputs false, which means that there exists collusion among TPA and CSS

and that the data corruption has occurred within CSS. The correctness of (12) is demonstrated as follows:

$$\begin{aligned}\sigma^{(T)} &= \prod_{t\in T}\sigma^{(t)} = \prod_{t\in T}\prod_{(i,\alpha_i)\in Q^{(t)}}\sigma_i^{\alpha_i}\\ &= \prod_{t\in T}\prod_{(i,\alpha_i)\in Q^{(t)}}\left(H(\text{fid}\parallel i)^{\alpha_i}\cdot\prod_{j=1}^{s}u_j^{\alpha_i\cdot b_{ij}}\right)^{\alpha}\\ &= \left(\prod_{t\in T}\prod_{(i,\alpha_i)\in Q^{(t)}}H(\text{fid}\parallel i)^{\alpha_i}\right.\\ &\quad\left.\cdot\prod_{j=1}^{s}u_j^{\sum_{t\in T}\sum_{(i,\alpha_i)\in Q^{(t)}}(\alpha_i\cdot b_{ij})}\right)^{\alpha} = \left(\eta^{(T)}\cdot\prod_{j=1}^{s}u_j^{\mu_j^{(T)}}\right)^{\alpha}.\end{aligned} \tag{13}$$

5. Security Analysis

Similar to the analysis of [8, 12], we evaluate the soundness of our DOPOR scheme according to three parts: unforgeability, liability, and extractability.

Theorem 1 (unforgeability). *It is computationally infeasible for any adversary $\mathcal{A}$ to forge a proof that can pass verifier's check, if the Computational Diffie-Hellman (CDH) problem and the Discrete Logarithm (DL) problem are hard.*

Proof. Since CSS does not check any proof throughout the whole process of executing DOPOR, there are only two cases to be discussed.

Case 1. TPA plays the role of verifier to check the proof returned from CSS during executing the protocols **GenProof**(·) and **PORAudit**(·) as shown in Section 4.4. In this case, observe that both CSS and TPA perform exactly the same as the BLS-based public verification scheme of [8], so the unforgeability guarantee immediately follows from the work of [8]. As shown in [21], the BLS scheme is secure when the CDH problem is hard in bilinear groups, based on which the unforgeability of BLS-based public scheme has been proven in [8] and thus omitted here.

Case 2. The client acts as the verifier to check TPA's logs as in the protocol **CheckLogs**(·) of Section 4.4. To pass the client's check with (12), TPA should return the correct proof $(\mu_1^{(T)}, \mu_2^{(T)}, \ldots, \mu_s^{(T)}, \sigma^{(T)})$. Now, assume that TPA is able to forge the proof. As shown in [8], due to the security of BLS scheme, the BLS-based homomorphic verifiable tag σ_i is unforgeable, and thus the aggregated tag $\sigma^{(T)}$ is also unforgeable. So, the only choice for TPA is to generate the forged aggregated block, denoted with $\widetilde{\mu}_1^{(T)}, \widetilde{\mu}_2^{(T)}, \ldots, \widetilde{\mu}_s^{(T)}$, as the response to client's check. Then, for (12) to be satisfied, we have

$$\left(\eta^{(T)}\cdot\prod_{j=1}^{s}u_j^{\widetilde{\mu}_j^{(T)}}\right)^{\alpha} = \sigma^{(T)}. \tag{14}$$

In addition, according to the correct proof, we have

$$\left(\eta^{(T)}\cdot\prod_{j=1}^{s}u_j^{\mu_j^{(T)}}\right)^{\alpha} = \sigma^{(T)}. \tag{15}$$

Note that $\eta^{(T)}$ is the parameter computed by client herself, and the security of $\eta^{(T)}$ is ensured by the security of bitcoin pseudorandom source of [12]. Based on the security of BLS scheme, we can learn that

$$\begin{aligned}&\prod_{j=1}^{s}u_j^{\mu_j^{(T)}} = \prod_{j=1}^{s}u_j^{\widetilde{\mu}_j^{(T)}} \Longrightarrow\\ &\prod_{j=1}^{s}u_j^{\Delta\mu_j^{(T)}} = 1,\end{aligned} \tag{16}$$

where $\Delta\mu_j^{(T)} = \widetilde{\mu}_j^{(T)} - \mu_j^{(T)}$ and $\widetilde{\mu}_j^{(T)} \neq \mu_j^{(T)}$.

For any two given elements $g_1, g_2 \in G$, we have $u_j = g_1^{p_j}g_2^{q_j} \in G$, where $p_j, q_j \in \mathbb{Z}_p$. Hence, (16) is transformed as follows:

$$\begin{aligned}&\prod_{j=1}^{s}\left(g_1^{p_j}g_2^{q_j}\right)^{\Delta\mu_j^{(T)}} = 1 \Longrightarrow\\ &g_2 = g_1^{-(\sum_{j=1}^{s}p_j\cdot\Delta\mu_j^{(T)})/(\sum_{j=1}^{s}q_j\cdot\Delta\mu_j^{(T)})}.\end{aligned} \tag{17}$$

Obviously, (17) means that malicious TPA can solve the DL problem, which is in conflict with the assumption that DL problem is hard. Therefore, it is infeasible for TPA to forge a proof to pass the client's check. This completes our proof. □

Theorem 2 (liability). *If any adversary $\mathcal{A}$ attempts to cheat or frame the honest entity who has been well behaving, the honest entity can output incontestable evidence to confirm the misbehavior of adversary $\mathcal{A}$ in case of lawsuit.*

Proof. It is clear that if the honest entity can protect against the collusion of the other two malicious entities, then this honest entity can certainly protect against any single malicious entity. Hence, to prove Theorem 2, it suffices to consider the following three cases where only one entity is honest.

Case 1. Honest client defends against the collusion of CSS and TPA. Obviously, CSS has incentive to collude with TPA only when the outsourced data corruption has occurred at cloud side. In this case, once the corrupted data blocks are challenged, according to Theorem 1, both CSS and TPA cannot forge an effective proof to pass client's check against TPA's log, unless they can solve the DL problem (but this probability is negligible). Therefore, the false output by client when executing the protocol **CheckLogs**(·) is the incontestable evidence to identify the collusion of CSS and TPA.

Case 2. Honest TPA defends against the collusion of client and CSS. As shown in the protocol **PORAudit**(·) of Section 4.4, TPA completes his auditing work by computing the auditing parameter $\xi^{(t)}$ and verifying (8), where all these processes are reproducible and undeniable for the malicious entities.

More specifically, on the one hand, the nonrepudiation of parameter $\xi^{(t)}$ is derived from the objectivity of challenge $Q^{(t)}$, which is computed based on the secure bitcoin pseudorandom source and the public parameters $\{\mathcal{sp}, \mathcal{cl}\}$. On the other hand, all other inputs involved in verifying (8) are also undeniable for both client and CSS; for example, $(\mu_1^{(t)}, \mu_2^{(t)}, \ldots, \mu_s^{(t)}, \sigma^{(t)})$ are signed by CSS's signature, and $\{g, \omega, u_1, \ldots, u_s\}$ are the public parameters confirmed by all entities. Hence, the honest TPA can provide his logs $\Lambda^{(t)}$ in case of lawsuit, which includes all above incontestable evidence enabling the playback of all past TPA's auditing work to prove the innocence of TPA.

Case 3. Honest CSS defends against the collusion of client and TPA. When malicious client colludes with TPA to falsely accuse CSS of corrupting the ith block b_i, CSS can output the intact $b_i = \{b_{ij}\}_{1 \le j \le s}$ and its tag σ_i as the incontestable evidence. As shown in [8], each tag σ_i constructed and outsourced by client herself is unforgeable. Based on the security of BLS signature scheme, as long as the above b_i and σ_i output by CSS satisfy (18), then CSS is innocent.

$$e(\sigma_i, g) \stackrel{?}{=} e\left(H(\text{fid} \parallel i) \cdot \prod_{j=1}^{s} u_j^{b_{ij}}, \omega\right). \tag{18}$$

This completes the proof of Theorem 2. □

According to Theorem 2, all the three entities have to behave properly in DOPOR. In this case, the extractability during the TPA' audits against CSS can immediately follow from the work of [8], since the procedure of TPA's audits of DOPOR corresponds to the public verification scheme of [8]. As for the extractability of performing **CheckLogs**(·) protocol, we have the following theorem.

Theorem 3 (extractability). *During the client's checking against TPA, if client does not output false after checking TPA's logs, then there exists a deterministic extraction algorithm, based on which client can extract the challenged file blocks by the repetitive interactions with TPA.*

Proof. According to Theorem 1, to pass client's check during the execution of protocol **CheckLogs**(·), TPA has to respond to client with the correct proof $\varrho^{(T)}$ that includes the aggregated block $(\mu_1^{(T)}, \mu_2^{(T)}, \ldots, \mu_s^{(T)})$, and each $\mu_j^{(T)}$ is a linear equation of the following form:

$$\mu_j^{(T)} = \sum_{t \in T} \mu_j^{(t)} = \sum_{t \in T} \sum_{(i, \alpha_i) \in Q^{(t)}} \left(\alpha_i \cdot b_{ij}\right), \tag{19}$$

where T is a set of point-in-times chosen by client herself, and all the coefficients α_i are dominated by these point-in-times as shown in the protocol **RandomChal**(·) of Section 4.4.

Now, suppose client chooses the appropriate point-in-times in the past to generate different set T and checks TPA's logs for a polynomial number of times by sending different T to TPA; then, client can get a total of s systems of linear equations that are built upon the challenged target blocks. Finally, by solving these s systems, client can extract all target blocks. □

When referring to dynamic updates, as shown in Section 4.3, the procedure of performing updates is divided into two parts: (1) performing the batch updates upon buffer U according to the algorithms *Batch_Updates* (·) and *Verify_Updates* (·), as shown in Section 3.4, the essence of which is based on the property of batch verifications of bv23Tree, while the security of this property is ensured by Theorem 4; (2) outsourcing a batch of encoded operations blocks and their tags to buffer E, the security of which is directly ensured by the security of the unforgeable tags, the periodic executions of POR audits against buffer E, and the erasure code scheme.

Theorem 4. *Assuming the existence of a collision-resistant hash function $h(\cdot)$, for any k ordered indices set $\Pi = \{a_1, a_2, \ldots, a_k\}$ appointed by the client, the corresponding proof table $T_{k \times l}$ generated using the bv23Tree ensures the integrity of all the k appointed leaves $\mathscr{L} = \{v_{a_1}, v_{a_2}, \ldots, v_{a_k}\}$ returned from CSS with overwhelming probability.*

Proof. As shown in Section 3.2, upon receiving the appointed ordered indices set $\Pi = \{a_1, a_2, \ldots, a_k\}$, CSS should respond to client with the corresponding leaves $\mathscr{L} = \{v_{a_1}, v_{a_2}, \ldots, v_{a_k}\}$ and proof table $T_{k \times l}$. Now, suppose that CSS tries to act dishonestly; then, the possible ways for CSS to misbehave can be covered by the following two cases.

Case 1. Malicious CSS either forges some leaves within $\mathscr{L}$ or forges some items within the proof table $T_{k \times l}$. As shown in Section 3.3, client possesses the public tree root hash x_{root} and will verify both $\mathscr{L}$ and $T_{k \times l}$ by calling the algorithm *Batch_Verify* (·), the procedure of which is to recalculate the public x_{root} by iteratively hashing the values of all tree nodes included in $\mathscr{L}$ and $T_{k \times l}$ in terms of the specified order. Apparently, the above forged $\mathscr{L}$ or $T_{k \times l}$ can enable *Batch_Verify*(·) to output the same root hash value as x_{root}, meaning that CSS is able to find the collisions against the hash function $h(\cdot)$, which contradicts the assumption that $h(\cdot)$ is collision-resistant. Therefore, it is a negligible probability for malicious CSS to forge $\mathscr{L}$ or $T_{k \times l}$.

Case 2. Malicious CSS launches the replacing attack; that is, CSS returns the replaced $\widetilde{\mathscr{L}}$ where some appointed leaves are replaced with other existing leaves of bv23Tree and the corresponding proof table $\widetilde{T}_{k \times l}$ that is correctly generated based on $\widetilde{\mathscr{L}}$. Without loss of generality, suppose that $\widetilde{\mathscr{L}} = \{v_x, v_y, \ldots, v_{a_k}\}$, where v_x and v_y are not the appointed leaves; that is, $x \neq a_1$ and $y \neq a_2$. In this case, although the final hash value output by *Batch_Verify* (·) is equal to the public x_{root}, the final value of the variable $\mathcal{IS}_1$ within *Batch_Verify* (·) will be $\{x, y, \ldots, a_k\}$ instead of the specified $\{a_1, a_2, \ldots, a_k\}$, which contradicts the expected results as shown in Section 3.3. So, client will still output false meaning that above malicious attack is detected by client.

In short, if there exists a collision-resistant hash function, malicious CSS has to return all the appointed leaves along

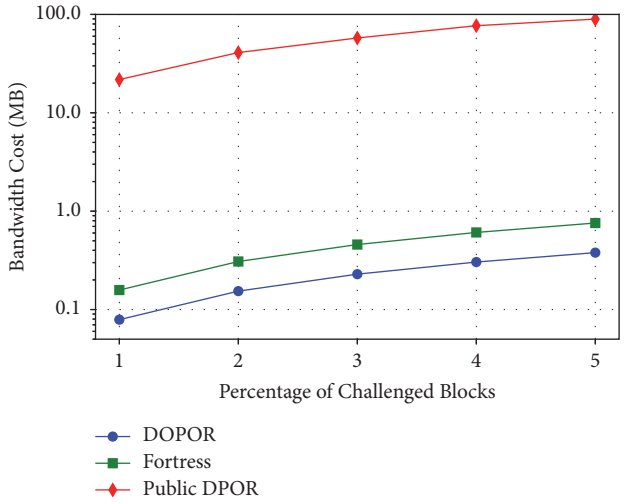


Figure 5: TPA-CSS bandwidth cost for a POR audit, with respect to the fraction of challenged blocks of the total number of encoded blocks.

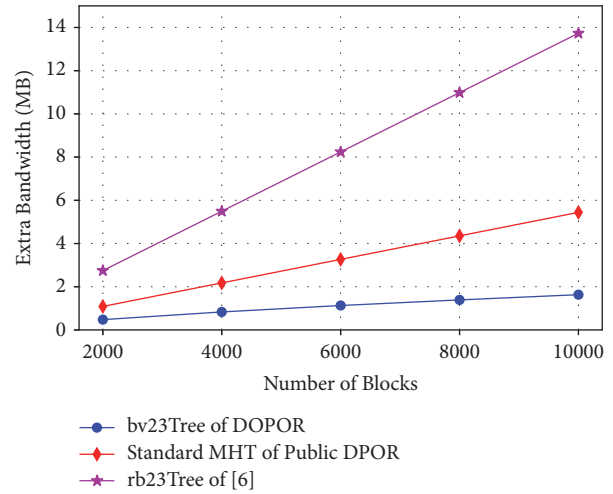


Figure 6: Extra bandwidth cost consumed by client for verifying the leaves of all blocks batch-read from CSS, according to different authenticated data structures.

with the correct proof table to pass client's batch verifications upon these leaves. □

6. Performance Evaluation

Our experiments were deployed using Python language on the Linux system with Intel Xeon E5-2609 CPU running at 2.40 GHz, 16 GB of RAM, and 7200 RPM 600 GB Serial ATA drive with a 32 MB buffer. The cryptographic operations were implemented based on the Python Cryptography Toolkit [22] and Pypbc library [23], and we used the 80-bit security parameter that means the order p of group G is of 160-bit length. We chose 1 GB raw data file F for testing and relied on the (9, 12) erasure code for encoding. For ease of comparison, all block sizes are set to 4 KB as in [6, 14]. Our results are an average of 20 rounds.

6.1. POR Audits Cost. During each POR audit, since the number of challenged blocks $|I|$ is far less than the total number of encoded file blocks $\widetilde{n}$ (e.g., the percent $|I|/\widetilde{n} = 1\%$ as in [12]), the time consumed in proof computation (or proof verification) will not be the bottleneck for CSS (or TPA). The POR audit phase of DOPOR corresponds to the execution of the public verification construction of [8], the efficient computation performance of which has been confirmed as shown in previous studies [5, 6, 10]. Therefore, the computation time of POR audit phase is not the primary concern in our DOPOR scheme, and we will focus on evaluating the bandwidth cost of this phase. Figure 5 depicts the total TPA-CSS bandwidth cost for executing POR audit once, for various percents of challenged blocks. Here, with regard to the given parameters $c\ell$ and λ in DOPOR, the percentage of challenged blocks is equal to $c\ell/\lambda$. It is obvious that the public DPOR of [14] results in a large communication overhead since it must transfer all challenged blocks during each audit, which greatly affects the bandwidth performance. By relying on the technologies of blockless verification and homomorphic authenticators (tags) to compress the proof size, the bandwidth costs of both our DOPOR and the static OPOR (i.e., Fortress) of [12] are only dominated by the sizes of challenges released from TPA and thus gradually increase with the percents of challenged blocks. Note that the bandwidth cost of DOPOR is always less than that of Fortress, since TPA only needs to send a single challenge for each audit in DOPOR, but there are two parallel challenges for TPA to be sent in Fortress. This is due to the fact that, during each audit, Fortress requires CSS to respond with two different responses [12]: one is used by TPA for auditing CSS and the other will be used by client for checking TPA's work. And thus these two responses correspond to two parallel challenges in Fortress. However, as shown in Section 4.4, DOPOR enables CSS to respond with only one response, which is based on the public key cryptosystem to support both TPA's auditing and client's checking. So, within DOPOR, there is only one challenge corresponding to the above sole CSS's response.

6.2. Read Cost. When client reads a batch of raw blocks from CSS, the integrity of these blocks is guaranteed by the authenticated data structure of the up-to-date buffer U. In Figure 6, we evaluate the extra bandwidth cost (i.e., not including the bandwidth of transferring the blocks themselves) incurred on client side for batch-verifying the leaves of all returned raw blocks with the proposed proof table of bv23Tree, when compared to the costs of the rb23Tree method [6] and the standard MHT method [14] that can only verify all appointed leaves by transferring their respective proof paths. As shown, with the increasing number of blocks batch-read from CSS, the extra bandwidth cost caused by rb23Tree is much higher than MHT, since the basic component of proof path of rb23Tree is an 8-element tuple *mark* with a larger size. However, owing to the proof table that avoids transferring all repetitive node values within different proof paths, our bv23Tree incurs the lowest extra bandwidth cost among the three dynamic structures. Likewise, as shown

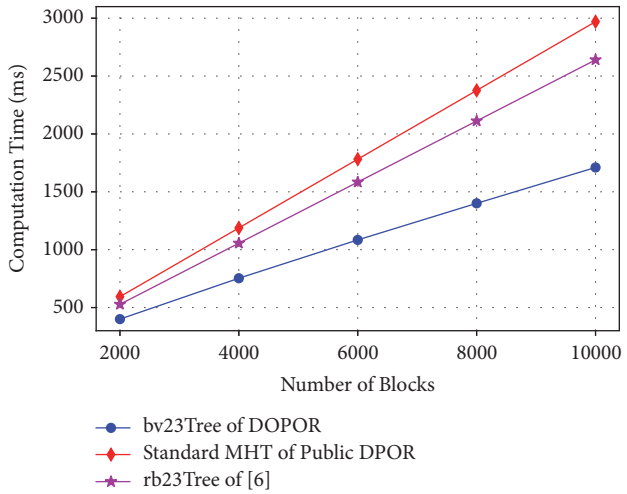


Figure 7: Computation time spent by client for verifying the integrity of all blocks batch-read from CSS, according to different authenticated data structures.

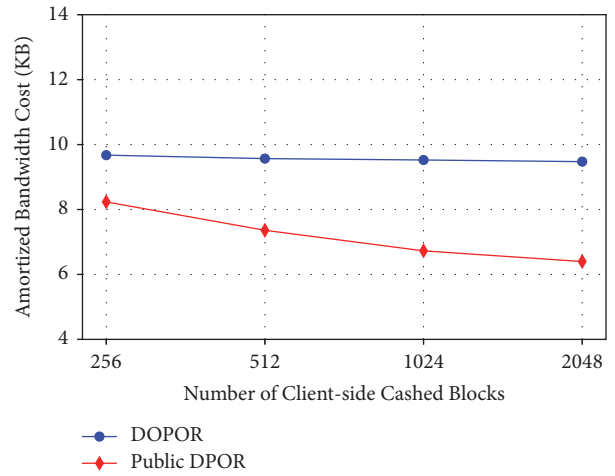


Figure 8: Client-CSS amortized bandwidth cost for each block when writing all cached blocks in a batch to CSS.

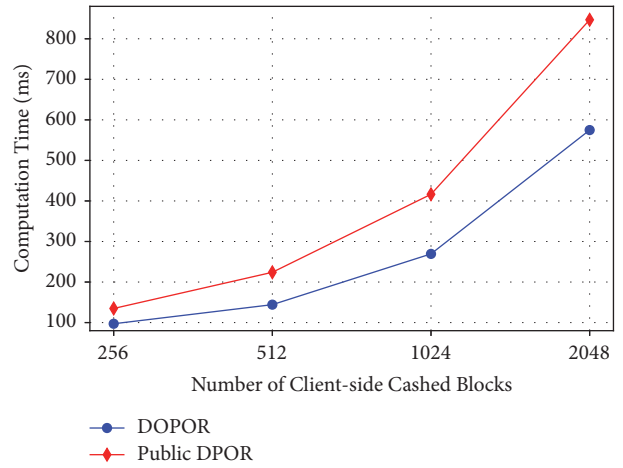


Figure 9: Computation time spent by CSS when writing all cached blocks in a batch to CSS.

in Figure 7, based on bv23Tree, client also further reduces the computation time spent for verifying the integrity of all returned blocks, due to the fact that the proof table enables client to batch-verify all returned leaves together just by computing the tree root hash once, avoiding the straightforward way of rb23Tree and standard MHT that client has to verify different leaves one after another by repeatedly computing the tree root hash with different proof paths.

In conclusion, our results show that the more the blocks batch-read from CSS, the more the repetitive node values omitted for transferring and computing according to proof table, and thus the client will save more costs based on bv23Tree for improving the performance of reads.

6.3. Write Cost. Now, we evaluate the performance of writes for DOPOR and the public DPOR of [14], both of which apply the client-side cache measure for performing writes; that is, client will cache locally a group of raw blocks (contained in the update operations of modification or insertion as in Section 3.4) and write these blocks in a batch to CSS, as shown in the protocol **PerformUpdates**(·) of Section 4.3. For DOPOR of this experiment, the parameter c is the number of client-side cached blocks, which determines the parameter λ according to the erasure-coding rate.

Figure 8 depicts the client-CSS amortized bandwidth cost for writing each 4 KB raw block. With the increasing number of client-side cached blocks, our results show that DOPOR incurs 17%~48% more amortized bandwidth than the public DPOR, due to the fact that DOPOR needs to transfer the additional encoded operations blocks besides the raw blocks. However, recall that the required bandwidth cost for frequent POR audits in the public DPOR is orders of magnitude higher than that in DOPOR (Figure 5), and DOPOR achieves a stronger security level by protecting against malicious TPA and collusion than the public DPOR. Furthermore, as shown in Figure 9, the public DPOR incurs an average of 45% more computation time at CSS side than DOPOR, since during performing writes the public DPOR must rebuild the corresponding levels of an MHT-based hierarchical structure located on CSS's disk, which results in a lot of additional disk I/O time when compared to DOPOR that does not need to do such rebuilding.

6.4. Client-Side Checking Cost. As shown in [12], although client should not be embroiled in the most frequent POR audits, it is necessary to give client the capability of checking TPA's past work to protect against the malicious TPA.

Since both DOPOR and the Fortress scheme of [12] adopt the aggregation technology to compress the proof size, the client-TPA bandwidth costs during checking TPA are alike for these two schemes, so in this experiment we focus on measuring the client's computation time of two investigated schemes when batch-checking TPA's logs, as shown in Figure 10. Here, one log corresponds to a past POR audit performed by TPA with challenging 1% of the total

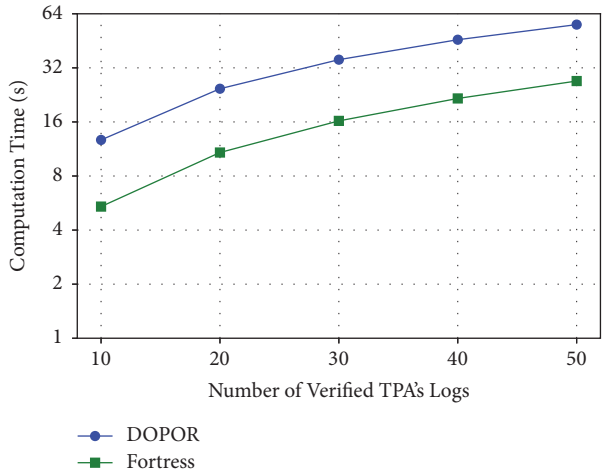


Figure 10: Computation time spent by client for batch-checking TPA's logs (including the time to access the bitcoin source).

encoded blocks. Compared to Fortress, DOPOR requires more time at client side for checking TPA, due to the fact that the exponentiation operation on the elliptic curve of DOPOR incurs more computation cost than the module operation of Fortress. However, recall that DOPOR enables the efficient data dynamics that cannot be supported in Fortress, so the additional client-side time cost incurred by DOPOR, that is, the distance between the line of DOPOR and that of Fortress as in Figure 10, can be regarded as the price for dynamism under the OPOR setting. Indeed, this additional dynamism cost can be tolerated by client to a great extent, since the client's checking against TPA is an optional verification and is only seldom executed in practice [12]; for example, client might just batch-check TPA's logs once in several months or even a year.

7. Related Work

Nowadays, with the rapid development of cloud computing, more and more cloud applications are designed upon the big data stored at CSS side, such as the service quality evaluation [24] and cloud service recommendation [25, 26]. However, how to guarantee the storage security of the big data is a critical challenge for mobile clients in the setting of cloud computing. Proof of Retrievability (POR) is a kind of security measure that builds upon cryptographic proofs to ensure the correctness and retrievability of client's big data outsourced to cloud. Juels and Kaliski Jr. [7] proposed the first POR scheme by utilizing the "sentinels" technique, where client can conceal some sentinel blocks among other original data blocks for remote POR audits before outsourcing her data. But this proposal can only support a limited number of POR audits, since performing the audits will expose the corresponding sentinels, so the frequent audits cannot be sustained once all sentinels are exhausted. Based on the pseudorandom functions (PRFs) and BLS signatures [21], Shacham and Waters [8] proposed two improved POR schemes with private verification and public verification, respectively. Both of these schemes enable an unlimited number of audits against CSS and simultaneously compress the response of CSS into one aggregated block along with a small authenticator value for optimized auditing bandwidth. Subsequently, Dodis et al. [27] generalized the constructions of [7, 8] by combining the concepts of POR with the coding and complexity theory. In view of the importance of data dynamics, Cash et al. [18] provided a Dynamic POR (DPOR) scheme based on the ORAM technique. Because ORAM will incur the heavy bandwidth overhead for client when performing dynamic updates under the POR setting, by replacing ORAM with the FFT-based constructible code and the hierarchical storage structure, Shi et al. [14] designed a more efficient private DPOR scheme than that of [18] and simultaneously applied the MHT structure to turn this private DPOR into a public DPOR scheme. Furthermore, with the observations made upon previous POR studies, Etemad and Küpçü [19] proposed a general framework to construct efficient DPOR and defend against the selective deletion attack described in [14].

On the other hand, Provable Data Possession (PDP), first proposed by Ateniese et al. [17], is a closely related research direction that focuses on ensuring the integrity of outsourced data. The difference between POR and PDP is that POR applies the erasure code but PDP does not. As shown in [19], the security level of POR is stronger than that of PDP, since POR ensures that *the whole* outsourced data can be retrieved by client, when compared to PDP that only guarantees the integrity of *most of* the outsourced data. Given that some existing public auditing schemes [5, 15, 16] are designed without involving erasure code, these schemes can be classed as the variants of PDP. Zhu et al. [4] presented a cooperative PDP (CPDP) scheme for distributed Multicloud Storage setting. Wang et al. [5] designed the random masking technology to protect client's outsourced data from leaking to TPA during the audits. Erway et al. [13] proposed the first Dynamic PDP (DPDP) scheme to support efficient data updates using Skip List. And then variant authenticated structures were proposed for data dynamics, such as the standard MHT method [9], rank-based MHT [15], multireplica MHT [16], and rb23Tree [6]. However, all these authenticated structures can only verify different leaves one by one, which is an inefficient way for client when there are many leaves that need to be verified.

In addition, as shown in [12], when referring to public verification (auditing), the potential security risk is that TPA might also be malicious. But this risk has not been considered by all the above public schemes. Outsourced Proof of Retrievability (OPOR), proposed by Armknecht et al. [12], is the first scheme to protect against malicious TPA under the public POR setting. However, the OPOR construction of [12] only supports the static data, which is the limitation that should be further solved.

8. Conclusions

As a stronger security model in the context of remote data auditing, Outsourced Proof of Retrievability (OPOR) focuses on dealing with the dilemma that client hopes to resort to TPA for assessing the storage security of her outsourced

data, while TPA might be malicious and collude with CSS to cheat client. In this paper, we propose a concrete DOPOR scheme to support data dynamics under the environment of OPOR. Our DOPOR scheme is constructed based on a newly designed authenticated data structure, called bv23Tree, which not only relies on the property of balanced tree to guarantee the expected logarithmic complexity in any case of dynamic updates, but also enables client to batch-verify multiple appointed leaves all together for improved performance. Under the setting of employing erasure code, by separating the updated data from the original data and adopting the hierarchical structure of same-sized levels to uniformly store all encoded data, DOPOR can efficiently support batch reads and updates upon outsourced storage according to the feature of batch verifications of bv23Tree. When compared to the state of the art, our experiments show that DOPOR incurs a lower bandwidth cost for frequent TPA's audits than the original static OPOR scheme, and the overall performance of DOPOR for reads and writes is comparable to that of existing public Dynamic POR scheme.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work is supported by NSFC (Grant no. 61502044).

References

[1] S. Guarino, E. S. Canlar, M. Conti, R. Di Pietro, and A. Solanas, "Provable Storage Medium for Data Storage Outsourcing," *IEEE Transactions on Services Computing*, vol. 8, no. 6, pp. 985–997, 2015.

[2] J. Yu, K. Ren, and C. Wang, "Enabling cloud storage auditing with verifiable outsourcing of key updates," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 6, pp. 1362–1375, 2016.

[3] K. He, J. Chen, R. Du, Q. Wu, G. Xue, and X. Zhang, "DeyPoS: Deduplicatable Dynamic Proof of Storage for Multi-User Environments," *IEEE Transactions on Computers*, vol. 65, no. 12, pp. 3631–3645, 2016.

[4] Y. Zhu, H. Hu, G.-J. Ahn, and M. Yu, "Cooperative provable data possession for integrity verification in multicloud storage," *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 12, pp. 2231–2244, 2012.

[5] C. Wang, Q. Wang, K. Ren, and W. Lou, "Privacy-preserving public auditing for data storage security in cloud computing," in *Proceedings of the IEEE INFO-COM*, pp. 525–533, March 2010.

[6] Z. Ren, L. Wang, Q. Wang, and M. Xu, "Dynamic proofs of retrievability for coded cloud storage systems," *IEEE Transactions on Services Computing*, vol. PP, no. 99, 2015.

[7] A. Juels and B. S. Kaliski Jr., "Pors: proofs of retrievability for large files," in *Proceedings of the 14th ACM Conference on Computer and Communications Security (CCS '07)*, pp. 584–597, ACM, Alexandria, VA, USA, November 2007.

[8] H. Shacham and B. Waters, "Compact proofs of retrievability," in *Advances in Cryptology—ASIACRYPT 2008: Proceedings of the 14th International Conference on the Theory and Application of Cryptology and Information Security, Melbourne, Australia, December 2008*, vol. 5350 of *Lecture Notes in Computer Science*, pp. 90–107, Springer, Berlin, Germany, 2008.

[9] Q.-A. Wang, C. Wang, K. Ren, W.-J. Lou, and J. Li, "Enabling public auditability and data dynamics for storage security in cloud computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 5, pp. 847–859, 2011.

[10] C. Wang, S. S. M. Chow, Q. Wang, K. Ren, and W. Lou, "Privacy-preserving public auditing for secure cloud storage," *IEEE Transactions on Computers*, vol. 62, no. 2, pp. 362–375, 2013.

[11] H. Tian, Y. Chen, C. C. Chang, H. Jiang, Y. Huang, and J. Liu, "Dynamic-hash-table based public auditing for secure cloud storage," *IEEE Trans. Services Comput*, 2015.

[12] F. Armknecht, J.-M. Bohli, G. O. Karame, Z. Liu, and C. A. Reuter, "Outsourced proofs of retrievability," in *Proceedings of the 21st ACM Conference on Computer and Communications Security, CCS 2014*, pp. 831–843, USA, November 2014.

[13] C. Erway, A. Küpçü, C. Papamanthou, and R. Tamassia, "Dynamic provable data possession," in *Proceedings of the Proceedings of the 16th ACM Conference on Computer and Communications Security (CCS '09)*, pp. 213–222, ACM, Chicago, Ill, USA, November 2009.

[14] E. Shi, E. Stefanov, and C. Papamanthou, "Practical dynamic proofs of retrievability," in *Proceedings of the 2013 ACM SIGSAC Conference on Computer and Communications Security, CCS 2013*, pp. 325–336, Germany, November 2013.

[15] C. Liu, J. Chen, L. T. Yang et al., "Authorized public auditing of dynamic big data storage on cloud with efficient verifiable fine-grained updates," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 9, pp. 2234–2244, 2014.

[16] C. Liu, R. Ranjan, C. Yang, X. Zhang, L. Wang, and J. Chen, "MuR-DPA: Top-Down Levelled Multi-Replica Merkle Hash Tree Based Secure Public Auditing for Dynamic Big Data Storage on Cloud," *IEEE Transactions on Computers*, vol. 64, no. 9, pp. 2609–2622, 2015.

[17] G. Ateniese, R. Burns, R. Curtmola et al., "Provable data possession at untrusted stores," in *Proceedings of the 14th ACM Conference on Computer and Communications Security (CCS '07)*, pp. 598–609, Virginia, Va, USA, November 2007.

[18] D. Cash, A. Küpçü, and D. Wichs, "Dynamic Proofs of Retrievability via Oblivious RAM," in *Advances in Cryptology – EUROCRYPT 2013*, vol. 7881 of *Lecture Notes in Computer Science*, pp. 279–295, Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[19] M. Etemad and A. Küpçü, "Generic efficient dynamic proofs of retrievability," in *Proceedings of the 8th ACM Cloud Computing Security Workshop, CCSW 2016*, pp. 85–96, Austria.

[20] A. V. Aho, J. E. Hopcroft, and J. D. Ullman, *The design and analysis of computer algorithms*, Addison-Wesley Publishing Co., Reading, Mass.-London-Amsterdam, 1975.

[21] D. Boneh, B. Lynn, and H. Shacham, "Short signatures from the Weil pairing," in *Advances in cryptology—ASIACRYPT 2001 (Gold Coast)*, vol. 2248 of *Lecture Notes in Comput. Sci.*, pp. 514–532, Springer, Berlin, 2001.

[22] "Python Cryptography Toolkit (PyCrypto)," 2014, https://pypi.python.org/pypi/pycrypto.

[23] "Python 3 bindings for libpbc (Pypbc)," 2017, https://github.com/debatem1/pypbc.

[24] L. Qi, W. Dou, Y. Zhou, J. Yu, and C. Hu, "A context-aware service evaluation approach over big data for cloud applications," *IEEE Transactions on Cloud Computing*, 1 page, 2015.

[25] L. Qi, X. Xu, X. Zhang et al., "Structural balance theory-based e-commerce recommendation over big rating data," *IEEE Transactions on Big Data*, 2016.

[26] L. Qi, X. Zhang, W. Dou, and Q. Ni, "A Distributed Locality-Sensitive Hashing-Based Approach for Cloud Service Recommendation From Multi-Source Data," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 11, pp. 2616–2624, 2017.

[27] Y. Dodis, S. Vadhan, and D. Wichs, "Proofs of Retrievability via Hardness Amplification," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics): Preface*, vol. 5444, pp. 109–127, 2009.

Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 5634561, 16 pages
https://doi.org/10.1155/2018/5634561

Research Article

Flexible, Secure, and Reliable Data Sharing Service Based on Collaboration in Multicloud Environment

Qiang Wei,[1] **Huaibin Shao**,[2] **and Gongxuan Zhang**[1]

[1]*School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, China*
[2]*Jiangxi G.H. Quality & Security Technology Co., Ltd., China*

Correspondence should be addressed to Qiang Wei; weiqiangarticle@163.com

Received 11 January 2018; Accepted 15 March 2018; Published 30 April 2018

Academic Editor: Ao Zhou

Due to the abundant storage resources and high reliability data service of cloud computing, more individuals and enterprises are motivated to outsource their data to public cloud platform and enable legal data users to search and download what they need in the outsourced dataset. However, in "Paid Data Sharing" model, some valuable data should be encrypted before outsourcing for protecting owner's economic benefits, which is an obstacle for flexible application. Specifically, if the owner does not know who (user) will download which data files in advance and even does not know the attributes of user, he/she has to either remain online all the time or import a trusted third party (TTP) to distribute the file decryption key to data user. Obviously, making the owner always remain online is too inflexible, and wholly depending on the security of TTP is a potential risk. In this paper, we propose a flexible, secure, and reliable data sharing scheme based on collaboration in multicloud environment. For securely and instantly providing data sharing service even if the owner is offline and without TTP, we distribute all encrypted split data/key blocks together to multiple cloud service providers (CSPs), respectively. An elaborate cryptographic protocol we designed helps the owner verify the correctness of data exchange bills, which is directly related to the owner's economic benefits. Besides, in order to support reliable data service, the erasure-correcting code technic is exploited for tolerating multiple failures among CSPs, and we offer a secure keyword search mechanism that makes the system more close to reality. Extensive security analyses and experiments on real-world data show that our scheme is secure and efficient.

1. Introduction

With the arrival of data times, both individuals and enterprises are producing huge amount of data every day, which profoundly influences our living style. On one hand, the explosively increasing data creates the limits for data storage and data transmission; on the other hand, data has become an important productivity resource that could help users make data-driven decisions by some computational methods, such as giving optimized scheme by operational research [1, 2], giving accurate prediction by regression analysis [3], or carrying out geometrical computation [4]. Therefore, it is necessary to design an efficient data sharing model to share huge amount of data among different individuals or organizations. Obviously, cloud computing is certainly a prefect data sharing tool for its vast storage space and reliable distributed storage system.

However, even if there are various advantages of cloud computing, security and privacy concerns are the primary obstacles to wide adoption [5]. In terms of data sharing, the data owner would like to outsource his/her data files to the remote public cloud servers and enable other users to find and download what they are interested in from cloud dataset. Although the security issues of data transmission have been deeply researched [6], which guarantees that the adversary is difficult to steal data privacy during transmission process, some potential risks still exist. For example, "curious" cloud service providers may steal valuable data by themselves for some economic benefits. To address such problem, a general approach is that the owner encrypts his/her data files with self-chosen keys before outsourcing [7]. But, in this case, how the owner flexibly and securely shares data with users is still not fully investigated.

Previous works have proposed two kinds of encrypted cloud data sharing models. *The first one* is that the data owner

encrypts their outsourced data with specific encryption key(s) so that only specific authorized data users could access and decrypt encrypted data. For example, the identity-based encryption (IBE) schemes [8, 9] and proxy-based reencryption (PRE) schemes [10, 11] enable the owner to encrypt outsourced data based on the users' identification or their public key information. Moving a step forward, the attribute-based encryption (ABE) schemes [12, 13] could achieve a more flexible encryption model that enables data owner to encrypt data for user groups with certain attributes' combinations instead of one specific user. The ABE has become one of the most popular methods for access control field. *The second one* is that the data owner directly shares the decryption key information with authorized users. This method is widely used in most searchable encryption (SE) schemes [14, 15]; it enables the owner to give the decryption key after knowing who (data user) will download which data files of him/her. Obviously, this method provides a significant advantage in "Paid Data Sharing" model ("Paid Data Sharing" model: users should pay for the downloaded data files according to their amount and prices).

Unfortunately, both methods still have some limits when utilizing them in real applications. *Firstly*, in real data sharing system (like 4shared.com and mymedwall.com), data owner cannot determine which outsourced data files will be downloaded by which data users (even the users' attributes) before outsourcing. So the solutions based on the first method like IBE, PRE, and ABE are not feasible anymore, because they need the owner to possess the specific information of users for each data file. *Secondly*, under the "Paid Data Sharing" model, it is unreasonable to make owner give all the decryption keys to the authorized users in system initialization phase. That is to say, if we follow the second method, the owner has to remain online all the time for users instantly obtaining corresponding decryption key(s). However, in reality, it is common that the data owner is the individual or resource-limited enterprises that find it difficult to always remain online.

Aside from the two limits we mentioned above, in "Paid Data Sharing" scenario, it is significantly important to correctly record the details of each data exchange transaction (called "bill"), which is directly related to the benefits of owner. An alternative solution to address all these problems is to import a trusted third party (TTP) in the system; the TTP is responsible for storing and distributing the encryption keys to authorized users and maintaining the bills simultaneously. Obviously, there is no doubt that data confidentiality and bill correctness are both well guaranteed by TTP, even if the owner is completely offline. But the security of such solution relies wholly on the TTP; once the TTP is broken, all plaintext form of data will be leaked. Besides, it is not easy to find a trusted entity as TTP in practical network environment. The similar views have been early proposed in many previous works, such as the key escrow problem in ABE schemes: the security of these schemes depends majorly on the key authority (like our TTP), which will make the confidentiality of outsourced data at a potential security risk [12, 16]. From the above, it is necessary to design a flexible and secure data sharing scheme in "Paid Data Sharing" scenario, which supports offline data owner without TTP.

In this paper, we discuss a "Paid Data Sharing" scheme that can be simply described as follows: a data owner encrypts his/her multiple data files and outsources them to public cloud platform, so that all authorized users could search what they are interested in and then pay and download them from cloud server. Finally, users decrypt downloaded encrypted data locally. Note that our scheme is more practical and flexible compared with previous works. Specifically, the owner does not know which data users (even their attributes) will download and decrypt which outsourced data files of him/her, while the owner cannot always keep online and there is no trusted third party in the system. For providing immediate data sharing service when the owner is offline, the owner needs to utilize the secret sharing approach to split and encrypt the file decryption key and then outsource the encrypted key blocks to multiple cloud service providers (denoted as CSPs hereafter). Except from decryption key, the owner also encrypts and distributes data blocks encoded with the erasure-correcting code technology in multiple CSPs, so that even if there are data losses occurring in some CSPs, the data redundancies can help to successfully recover the complete outsourced data for supporting reliable data sharing service. Meanwhile, an elaborately designed cryptographic protocol guarantees the correctness of each of data transactions (bills); if there are some entities deceiving the data owner for the bill content, this will be detected by the system as a high probability. Besides, we import a semitrusted proxy to provide a secure and efficient keyword-based query function that enables data user to search what he/she needs in outsourced data files before downloading. Our major contributions are shown as follows:

(i) For the first time, we explore the problems of such a practical data sharing scenario: the data owner does not know which data users (even the attributes) will download his/her encrypted files before data outsourcing, and the owner cannot always remain online simultaneously. Besides, there is no TTP in our system.

(ii) We design an elaborately cryptographic protocol that could guarantee the correctness of each of data exchange transactions (bills), so that the deceptive behavior from some semitrusted entities will be detected as a high probability.

(iii) Thorough security analysis and experiments on the real-world dataset show that our proposed scheme is secure and efficient.

The remainder of this paper is organized as follows. In Section 2, we give some necessary notations of our scheme and describe the system model, threat model, and design goals. Section 3 introduces some background knowledge so that readers could understand our later scheme easily, followed by Section 4, in which we propose the details of our scheme. And then we analyze the security and evaluate the performance of our proposed scheme in Section 5. The related

works are discussed in Section 6. Finally, Section 7 covers the conclusion.

2. Problem Formulation

2.1. Notations. We first introduce some necessary notations that will make our later description convenient, shown as follows:

(i) F: the collection of m data files that would be outsourced to multiple cloud service providers, denoted as $F = \{F_1, F_2, \ldots, F_m\}$, where F_i, $i \in \{1, 2, \ldots, m\}$, is the ith file in the collection

(ii) D_i: the encoded results for data file F_i. Assuming that (n, t) Reed-Solomon code is exploited for encoding data in our scheme, D_i can be denoted as the collection of totally n blocks: $D_i = \{D_{i,1}, \ldots, D_{i,t}, \ldots, D_{i,n}\}$, where $D_{i,j}$, $j \in \{1, 2, \ldots, n\}$, represents the jth data block

(iii) C_i: the ciphertext form of D_i: $C_i = \{C_{i,1}, C_{i,2}, \ldots, D_{i,n}\}$

(iv) K: the encryption key collection for data file collection F: $K = \{K_1, K_2, \ldots, K_m\}$, where K_i, $i \in \{1, 2, \ldots, n\}$, denotes the encryption key for F_i

(v) S_i: the encoded results for K_i. Similar to encoding method of D_i, $S_i = \{S_{i,1}, \ldots, S_{i,t}, \ldots, S_{i,n}\}$, where $S_{i,j}$, $j \in \{1, 2, \ldots, n\}$, represents the jth key block

(vi) $S^*_{i,j}$ and $S^{**}_{i,j}$: the ciphertext form of key block $S_{i,j}$ and the reencrypted form of key block $S_{i,j}$, respectively, where $S^*_{i,j}$ is encrypted by owner and $S^{**}_{i,j}$ is reencrypted by CSP.

(vii) B_i: the bill for data file $F_i(C_i)$, which consists of each transaction of F_i. The transaction generally includes the content about who download $F_i(C_i)$ at what price

(viii) W: the dictionary whose elements are extracted distinct keywords from all data files

(ix) Q: the query sent by user, denoted as the collection of multiple query keywords

(x) TD: the trapdoor generated from Q (i.e., encrypted form of Q)

2.2. System Model. There are totally 4 entities involved in our secure keyword search scheme, data owner, multicloud alliance, semitrusted proxy, and data users, as illustrated in Figure 1.

Data owner would like to share his/her valuable data with authorized users through public cloud platform. But he/she does not know who will download which data files before outsourcing and cannot remain online all the time. To ensure that only authorized users could use his/her outsourced data, owners will share a secret value L with all authorized data users in the initialization phase. For achieving the reliability, in the process of data upload, the owner firstly exploits the erasure-correcting code approach (like Reed-Solomon [17]) to split each data file F_i and its corresponding encryption key K_i as multiple block collection (i.e., D_i and S_i); then the owner encrypts D_i and S_i and sends encrypted data/key blocks (i.e., $\{C_{i,j}, S^*_{i,j}\}$) to all cloud service providers, respectively (each CSP will receive one encrypted data block and one encrypted key block). The owner also generates a searchable encrypted index for all files and outsources them to proxy, so that users could search what they are interested in before downloading. After carrying out all above upload tasks, the data owner could be offline.

Proxy is a semitrusted entity that maintains the encrypted index submitted by data owner. For improving the search efficiency, the index is an inverted list that was tailor-made for our scheme. Upon receiving query request from data user, the proxy searches on its local index and sends all identifications of encrypted data/key blocks that were included in searched results (we called these identifications "data request") to the multicloud alliance. Besides, since the proxy knows who download which outsourced data files (and their corresponding decryption key), it is also responsible for updating each bill B_i for each data file $F_i(C_i)$ and sending corresponding bills to the data user.

Multicloud alliance is composed of multiple cloud service providers (CSPs). It is responsible for storing the encrypted data blocks and encrypted key blocks. Upon receiving the data request from proxy, each CSP will firstly update the bills corresponding to encrypted data blocks it stores; then the randomly chosen t CSPs (chosen by user; the details are shown in next paragraph) reencrypt the corresponding encrypted key block with its updated bill and users public key. Finally, they send the encrypted data blocks as well as reencrypted key blocks together to the data user.

Data users would like to send a search request to the semitrusted proxy. The search request includes the following 3 items: the trapdoor TD, search parameter k, and t randomly chosen identifications of CSPs, where parameter k denotes that the user needs the top-k ranked results and CSPs' identifications represent user needs to download data blocks and key blocks from these t CSPs. After receiving corresponding encrypted data blocks and reencrypted key blocks from t CSPs, user could carry out the following three operations to recover the original plaintext form of data files: (1) with the bills sent by proxy and its secret key, user decrypts the reencrypted key blocks as their encrypted form that owner sends to CSPs (i.e., $S^{**}_{i,j} \rightarrow S^*_{i,j}$); (2) with the secret value L given by owner, user decrypts the encrypted key blocks as unencrypted one (i.e., $S^*_{i,j} \rightarrow S_{i,j}$); (3) with all plaintext forms of key blocks, users recover the decryption key K_i (K_i is also regarded as decryption key because we use symmetrical encryption mechanism in the scheme) and then decrypt the encrypted data blocks and recover the plaintext form of data file F_i.

2.3. Threat Model. From the system model we described above, the threat model can be defined in terms of the security settings of cloud service providers and proxy as follows.

Cloud service providers (CSPs) are "mixed-trusted" ones. (1) Most of them are "honest but curious" (i.e., semitrusted), which carry out the designed protocol honestly but are curious about the plaintext form of outsourced data file and

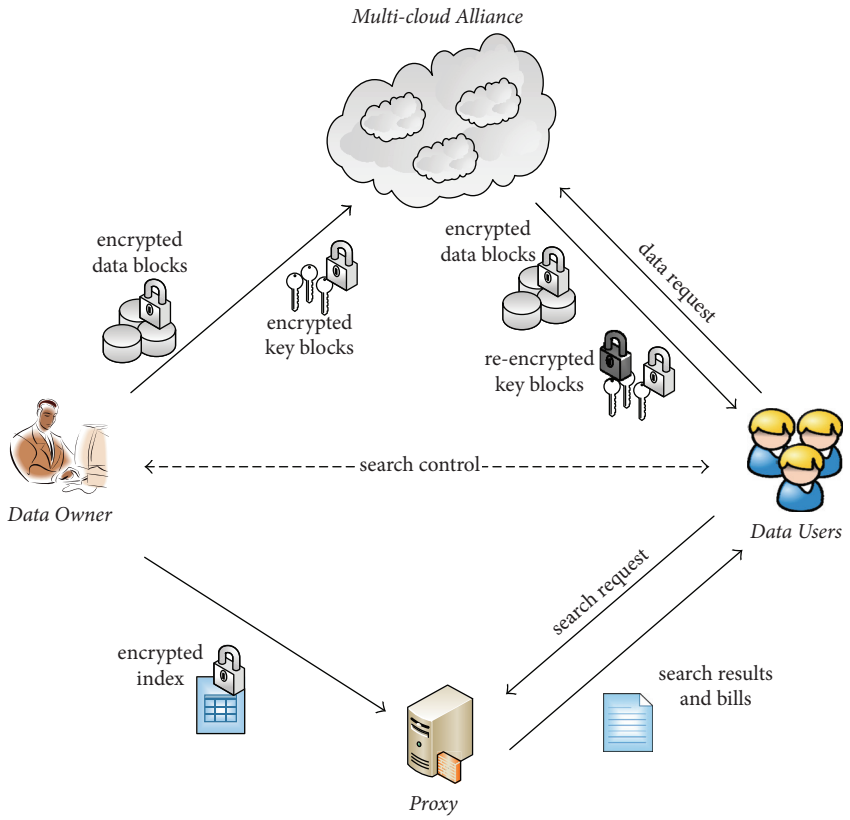


Figure 1: The architecture of our proposed data sharing system.

even collude with the proxy for deceiving the data user in terms of bills for stealing the economic benefits of data owner. (2) A small part of them are completely trusted, which will not collude with other CSPs or the proxy for destroying the data confidentiality or the correctness of bills. Note that the threat model we set for CSPs is based on the reality that there are many companies providing cloud computing service at present, and some of them are big companies that significantly value their reputation.

Proxy is "honest but curious," which executes the designed protocol honestly but is curious about the plaintext form of outsourced index and query keywords sent by user. Besides, the proxy would like to collude with CSPs for forging the bills.

Users are "honest but curious" ones who execute the designed protocol honestly but are curious about the plaintext form of outsourced data that other users downloaded. However, we assume that users do not collude with CSPs for revealing the secret value L or forging the bills, and they also would not reveal L to the proxy.

2.4. Design Goals. Our proposed data sharing scheme should achieve the following 4 design goals.

2.4.1. Security. Under the threat model we defined above, the adversary should not obtain the additional information of data owner and data user except the message it receives in protocol. Besides, the bills that record the data sharing information also should be confirmable ones to protect the economic benefits for data owner. Specifically, there are 3 aspects of security, shown as follows:

(i) *Data confidentiality*: the confidentiality of each of outsourced data blocks should not be revealed by any adversary

(ii) *Index/query privacy*: the plaintext form of outsourced index and encrypted query keyword (trapdoor) should not be revealed by any adversary

(iii) *Confirmable bills*: any adversary cannot deceive data owner successfully in terms of the forging data exchange bills

2.4.2. Reliability. Our scheme can provide reliable data sharing service even if the integrity of data blocks or key blocks of some CSPs is destroyed.

2.4.3. Flexibility. In the premise that the owner does not know who will download which outsourced data files before data outsourcing, the proposed scheme should guarantee achieving a real-time data sharing service even if data owner is offline, that is, data owner in our scheme is flexible enough.

2.4.4. Efficient. The response time of downloading the interested outsourced data files (the details of response time are shown in latter scheme) should be in an acceptable range.

Namely, for the entities who do not possess abundant computational resources (like the owner, user, and proxy), the computational overhead of them should be relatively small.

3. Preliminaries

In this section, we briefly introduce 3 pieces of basic knowledge utilized in our proposed scheme.

3.1. Erasure-Correcting Code. The erasure-correcting code is utilized to tolerate multiple failures in distributed storage systems. Simply, an integrate data file can be encoded as n blocks and only t of them can recover the original data file, where $t \leq n$. In this paper, if we store the encoded results of data file or encryption key (i.e., n data/key blocks) into n different CSPs, respectively, the original data file or encryption key can survive the failure of any $n - t$ of the n CSPs without any data loss. As for encoding method, the widely used Reed-Solomon (RS) erasure-correcting code is utilized in our scheme; taking the (n, t) RS code as an example, we briefly give the process of encoding as follows; more details are shown in literature [17].

For the data file F, we firstly let $F = (F_1, F_2, \ldots, F_t)$, where each data block F_j, $j \in \{1, 2, \ldots, t\}$, can be regarded as an l-length column vector whose elements all belong to $GF(2^p)$. Then we randomly generate a $t \times n$ Vandermonde matrix A as the dispersal matrix; according to a sequence of elementary row transformations, matrix A can be rewritten as the form of $(E \mid P)$, where E is a $t \times t$ identity matrix and P is the secret parity generation matrix with size $t \times (n-t)$, shown as follows:

$$A = (E \mid P) = \begin{pmatrix} 1 & 0 & \cdots & 0 & p_{11} & p_{12} & \cdots & p_{1(n-t)} \\ 0 & 1 & \cdots & 0 & p_{21} & p_{22} & \cdots & p_{2(n-t)} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & p_{t1} & p_{t2} & \cdots & p_{t(n-t)} \end{pmatrix}. \quad (1)$$

With the dispersal matrix A, we can encode the original data file F as follows:

$$\begin{aligned} D = F \cdot A &= F_i \cdot (E \mid P) \\ &= (D_1, D_2, \ldots, D_t, D_{t+1}, \ldots, D_n) \\ &= (F_1, F_2, \ldots, F_t, D_{t+1}, \ldots, D_n), \end{aligned} \quad (2)$$

where D are the encoded results composed with n blocks. Since $A = (E \mid P)$ is derived from a Vandermonde matrix, any t out of the n columns form an invertible matrix; that is, a $t \times t$ matrix whose t column vectors are distinct column vectors randomly sampled from D can be used to recover F.

3.2. Discrete Logarithm Assumption. Firstly, the discrete logarithm problem can be described as follows: for $x \in Z_p$, given g and $g^x \in G$, output x, where G is a multiplicative cyclic group in large prime order p and g is the generator of group G.

The discrete logarithm assumption means that it is computationally infeasible to solve the discrete logarithm problem in G.

3.3. Bilinear Maps. Assuming that G_1, G_2, and G_T are three multiplicative cyclic groups of prime order p; g_1 and g_2 are the generators of G_1 and G_2, respectively. Then a bilinear map $\hat{e} : G_1 \times G_2 \to G_T$ satisfies the following properties:

(i) *Bilinear*: for any $u \in G_1$, $v \in G_2$, and two random numbers $a, b \in Z_p$, $\hat{e}(u^a, v^b) = \hat{e}(u, v)^{ab}$

(ii) *Nondegenerate*: $\hat{e}(u, v) \neq 1$

(iii) *Computable*: there is a polynomial-time algorithm for computing the map $\hat{e}(u, v)$

4. The Proposed Scheme

In this section, we propose a feasible data sharing scheme that could achieve all the design goals we defined in Section 2.4.

4.1. Setup

4.1.1. Parameters Generation. First of all, the system generates and publishes the following parameters in initialization phase:

(i) G and G_T: two multiplicative cyclic groups of prime order p, where g is the generator of G

(ii) L: one of the secret values shared between owner and users. All users have the same L

(iii) r: the other secret value shared between owner and user. Different users have different r

(iv) $\hat{e}$: a bilinear map $G_1 \times G_1 \to G_T$. Its specific properties are shown in Section 3.3

(v) Z_p^*: a set of nonzero elements in q-order integer group

(vi) $H(\cdot)$: a secure collision-resistant hash function that could map the message with any length into an element of Z_p^*

(vii) $H_b(\cdot)$: a secure collision-resistant hash function that could map the exchange bill with any length into an element of G

(viii) $H_l(\cdot)$: a secure collision-resistant hash function that could map any element in G_T into a fix length (in our scheme, the fix length is 128-bit) binary number

(ix) A: the $t \times n$ dispersal matrix used for encoding data file and encryption key

4.1.2. Symmetric Key Generation. The data owner utilizes the traditional symmetric cryptographic mechanism (like AES algorithm) to encrypt each data block. Obviously, for protecting data confidentiality in "Paid Data Sharing" model, different data files (i.e., data block collection) should be encrypted with different encryption keys. Therefore, the owner generates encryption key K_i for each data file F_i. Besides, the secret values L and r are also generated in this

phase (shown as the "search control" in Figure 1), where L is a isometric binary number as K_i. For example, in terms of AES algorithm, K_i and L are both 128-bit length binary numbers.

4.1.3. Public/Secret Keys Generation. Upon receiving the data request from the proxy, t randomly chosen CSPs will firstly reencrypt the encrypted key blocks and then send the corresponding encrypted data blocks and reencrypted key blocks together to the user. For releasing from the heavy load of maintaining certificates produced by multiple users, we exploit the idea of identity-based encryption (IBE) to encrypt key information for user. Specifically, each authorized user should prepare a pair of public and secret keys, where the hash value of each user's identification (H(uID)) is the public key and the value $r \cdot H$(uID) is each user's secret key (as we mentioned above, different users have different secret values r). Besides, each data user publishes the parameter $P_u = g^r$ in the whole system.

4.2. Build Index. To enable data users to search on encrypted cloud data, the data owner needs to generate a searchable index and submit it to the semitrusted proxy. The proxy maintains the index and obtains the search results for the search request sent by user according to the index. For providing efficient search, we exploit the inverted index structure widely used in many related works [18, 19] to construct index for all outsourced data files, and each term in the inverted index is the encrypted form of corresponding keyword. The specific steps for building such inverted index are shown as follows.

4.2.1. Dictionary Generation. The data owner firstly needs to construct the dictionary that contains all distinct keywords extracted from all outsourcing data files. In this paper, we exploit the method proposed by Cash et al. [20] to extract keyword and then generate the dictionary W, which includes a total of 4 steps: (1) segmenting words, (2) removing stop words, (3) stemming words, and (4) merging keywords. Due to the more detailed steps shown in [20], we omit them here.

4.2.2. Secure Index Construction. According to the dictionary generated above, the plaintext form of inverted index can be built by data owner. Formally, it can be shown as Table 1, where $w_1, w_2, \ldots$ in the table are keywords of dictionary W; that is, the size of index is equal to the length of W. Besides, the document list in the table (i.e., $F_1, F_3, \ldots$) represents the documents containing corresponding keyword, and the identifications of encrypted data block collections and encrypted key block collections (i.e., C_1.ID, C_3.ID, ... and S_1^*.ID, S_3^*.ID, ...) denote the identifications of corresponding encrypted data/key blocks stored in the multicloud alliance, where C_i.ID $= \{C_{i,1}$.ID, $C_{i,2}$.ID, ..., $C_{i,n}$.ID$\}$ (S_i^*.ID can be denoted as the similar form). In practice, the inverted index does not contain the second column "document list" of Table 1; we show it here only for easily understanding the structure of index.

After constructing the plaintext form of inverted index, in order to protect the index privacy, owner should encrypt the keywords in the index before outsourcing them to semitrusted proxy. We design an efficient encryption algorithm based on the assumption that the discrete logarithm problem in large prime order cyclic group is difficult (the detailed explanation is shown in Section 3.2). The specific encryption approach is shown as follows:

$$w^* = \left(g^{H(L\|w)} \bmod p\right)^{-1}, \tag{3}$$

where w^* represents the ciphertext form of keyword w, g and $H(\cdot)$ are public parameters generated in system initialization phase (see Section 4.1.1), L is the secret value shared between owner and all users, and the operator "$\|$" denotes cascade. Note that the encrypted index only replaces the plaintext form of keywords with encrypted keywords, but the block list and the structure of index do not change at all.

Finally, data owner outsources the encrypted inverted index (denoted as I) generated above to the proxy.

4.3. Upload Encrypted Data/Key Blocks. Now we show how the data owner generates and uploads his/her data blocks and key blocks to the multicloud alliance.

4.3.1. Encrypted Data Blocks Generation. For providing the reliable and secure data sharing service, before outsourcing the data file to CSPs, we should encode and encrypt each original data file based on erasure-correcting code approach, so that our system could tolerate multiple failures (i.e., the data integrities of some CSPs in the multicloud alliance are destroyed). In our scheme, we utilized the (n, t) Reed-Solomon erasure-correcting code for data file encoding, where $t < n$ and n is the total number of CSPs in the multicloud alliance (according to the principle of RS code, the parameters t and n should satisfy $t \le n$, but $t = n$ means that user could recover the integrated data file if and only if there is no data loss of all CSPs. Therefore, for tolerating multiple failures in multicloud alliance, we let $t < n$).

Firstly, we generate plaintext form of n data blocks for each data file F_i in the file collection F; the specific calculation steps are the same as (2); we directly give the encoded result D_i for F_i as follows:

$$D_i = F_i \cdot A = \left(D_{i,1}, D_{i,2}, \ldots, D_{i,t}, D_{i,t+1}, \ldots, D_{i,n}\right). \tag{4}$$

Secondly, in order to protect the data confidentiality, data owner should encrypt all data block collections with corresponding encryption key K_i. In this paper, the encryption approach we utilized is the traditional AES algorithm, and the encrypted results of D_i are denoted as C_i:

$$D_i \xrightarrow[K_i]{\text{AES}} C_i = \left\{C_{i,1}, C_{i,2}, \ldots, C_{i,n}\right\}. \tag{5}$$

For each data file that needs to be outsourced, the owner encodes it and encrypts it as (4) and (5) show and then uploads all the m (i.e., the number of data files) encrypted data block collections $\{C_1, C_2, \ldots, C_m\}$ to the multicloud alliance, where each data block collection is composed of n data blocks; they should be randomly dispersed to n CSPs, respectively.

Table 1: The form of inverted index constructed by data owner.

Keyword	Document list	Identifications of encrypted data block collections	Identifications of encrypted key block collections
w_1	$F_1, F_3, \ldots$	$C_1.\mathrm{ID}, C_3.\mathrm{ID}, \ldots$	$S_1^*.\mathrm{ID}, S_3^*.\mathrm{ID}, \ldots$
w_2	$F_2, F_5, \ldots$	$C_2.\mathrm{ID}, C_5.\mathrm{ID}, \ldots$	$S_2^*.\mathrm{ID}, S_5^*.\mathrm{ID}, \ldots$
$\ldots$	$\ldots$	$\ldots$	$\ldots$

Input: the outsourced data file collection F; the encryption key collection K; the secret binary number L.
Output: the encrypted data/key blocks that should be outsourced to multi-cloud alliance.
(1) **for** each data file F_i in F **do**
(2) the owner encode it with (n, t) - RS code:
$$F_i \xrightarrow{\text{encode}} D_i = \{D_{i,1}, D_{i,2}, \ldots, D_{i,n}\}$$
(3) **for** each data block $D_{i,j}$ in D_i **do**
(4) the owner encrypts it with K_i: $D_{i,j} \xrightarrow[K_i]{\text{encrypt}} C_{i,j}$
(5) **end for**
(6) **end for**
(7) **for** each encryption (decryption) key K_i in K **do**
(8) the owner splits K_i as t isometric binary number:
$$K_i \xrightarrow{\text{split}} \{K_{i,1}, K_{i,2}, \ldots, K_{i,n}\}$$
(9) the owner encode it with (n, t) - RS code:
$$\{K_{i,1}, \ldots, K_{i,n}\} \xrightarrow{\text{encode}} S_i = \{S_{i,1}, \ldots, S_{i,n}\}$$
(10) **for** each key block $S_{i,j}$ in S_i **do**
(11) the owner encrypts it with L:
$$S_{i,j} \xrightarrow[L]{\text{encrypt}} S_{i,j}^* = S_{i,j} \oplus L$$
(12) **end for**
(13) **end for**
(14) **return** $\{C_i, S_i^*\}$, $i \in \{1, 2, \ldots, m\}$

Algorithm 1: *Upload*(F, K, L).

4.3.2. Encrypted Key Blocks Generation. In a similar way, we can also encode the encryption key K_i as n key blocks. However, since we use the AES algorithm for encrypting, K_i is only a 128-bit binary number that is not easily encoded. We will first split all K_i, $i \in \{1, 2, \ldots, m\}$, as t isometric binary number based on the XOR approach. Specifically, we randomly choose $t - 1$ 128-bit length binary vectors, denoted as $\{K_{i,1}, K_{i,2}, \ldots, K_{i,t-1}\}$; then the tth key blocks can be calculated as follows:

$$K_{i,t} = K_{i,1} \oplus K_{i,2} \oplus \cdots \oplus K_{i,t-1} \oplus K_i. \tag{6}$$

Obviously, we can recover K_i if and only if we possess all these t blocks.

Then we utilize the RS code to add $(n - t)$ redundant isometric binary number, shown as follows:

$$\begin{aligned} S_i &= \{K_{i,1}, K_{i,2}, \ldots, K_{i,t-1}, K_{i,t}\} \cdot A \\ &= (S_{i,1}, S_{i,2}, \ldots, S_{i,t}, S_{i,t+1}, \ldots, S_{i,n}), \end{aligned} \tag{7}$$

where S_i is a collection that contains n key blocks and any t blocks of them could be used to recover the original t key blocks we generated in (6).

Finally, the owner encrypts all n key blocks of each S_i as (8) and randomly disperses them to n CSPs, respectively, as the upload process of data blocks:

$$S_{i,t}^* = S_{i,t} \oplus L. \tag{8}$$

The whole process of uploading encrypted data/key blocks is shown as Algorithm 1.

4.4. Search. When the data user would like to search what he/she needs in the outsourced dataset, he/she will first generate trapdoor TD and randomly choose t CSPs to download data. Then users send TD, k, and the t identifications of CSPs together (called search request) to the proxy. Upon receiving the search request, the proxy performs search algorithm on the outsourced inverted index I and obtains the search results. If we call the data files that possess the query keywords "target files," then, according to the structure of index I, the search results are the identifications of encrypted data/key blocks corresponding to the target files. Finally, the proxy informs multicloud alliance to send corresponding data to users based on the search results (called data request).

4.4.1. Trapdoor Generation. Firstly, the data user needs to generate trapdoor TD based on several query keywords. In fact, the trapdoor consists of all encrypted query keywords. The algorithm for encrypting one query keyword w_q can be shown as follows:

$$w_q^* = \left\{ g^{(1+r)\cdot H(L\|w_q)} \bmod p, g^{r\cdot H(L\|w_q)} \bmod p \right\}, \quad (9)$$

where w_q^* is the ciphertext form of query keyword w_q; from the equation above, we see it is a two-tuple, $g, H(\cdot)$ are public parameters the system generates in the initialization phase, L is the secret value shared between owner and all users, and r is the secret value possessed by different owners.

4.4.2. Search Process. With trapdoor TD and index I, the proxy could find which encrypted index keywords in I match the encrypted query keywords in TD; then it could obtain the corresponding encrypted data/key blocks. For ranking the search results, we follow the basic principle that "more match, higher ranking"; that is to say, for one data file (i.e., encrypted data blocks), the more keywords occur in the data file and query simultaneously, the higher score the data file will obtain. For example, the data file $F_i(C_i)$ has the keywords w_1, w_2, and the data file $F_i'(C_i')$ has the keywords w_2, w_3; if the trapdoor is generated from w_1 and w_2, then the search score of $F_i(C_i)$ is 2 but the search score of $F_i'(C_i')$ is 1. According to the search score of each data file (encrypted data blocks collection), the proxy obtains identifications of the top-k results. The algorithm for determining whether a query keyword matches the index keyword is shown as follows.

We assume that the proxy could recognize identification of data user who sends the search request. Taking one query keyword w_q and one index keyword w as an example, if (10) holds, we say that w_q matches w.

$$\left(g^{(1+r)\cdot H(L\|w_q)} \bmod p \right) \cdot w^* = g^{r\cdot H(L\|w_q)} \bmod p, \quad (10)$$

where w_q^* is the encrypted query keyword in the trapdoor TD and w^* is the encrypted index keyword in I.

Correctness Proof. We can prove the correctness of (10) according to (9) and (3) as follows. If $w_q = w$,

$$\begin{aligned} &\left(g^{(1+r)\cdot H(L\|w_q)} \bmod p \right) \cdot w^* \\ &= \left(g^{(1+r)\cdot H(L\|w_q)} \bmod p \right) \cdot \left(g^{H(L\|w_q)} \bmod p \right)^{-1} \\ &= g^{r\cdot H(L\|w_q)} \bmod p. \end{aligned} \quad (11)$$

After obtaining search results, the proxy sends data request to t CSPs that are randomly chosen by user. And it also sends the data exchange transaction to all n CSPs, so that all CSPs in multicloud alliance could update their self-maintained bills. Finally, the proxy updates the bills it maintained (i.e., adding the exchange transaction to corresponding bills) and sends each bill B_i that has been updated to the user. Note that these bills will be used for decrypting encryption key in latter download process (see Section 4.5).

4.5. Download Encrypted Data/Key Blocks. For the t randomly chosen CSPs, they will firstly update the bills corresponding to data files (i.e., encrypted data blocks) that need to be downloaded and then reencrypt corresponding encrypted key blocks with updated bills and the public key of users. Finally, they send the encrypted data blocks with reencrypted key blocks together to the user. For example, assuming that the encrypted data block collection C_i is what user needs, for one of the t CSPs that stores one of the encrypted data blocks $C_{i,j}$ and encrypted key blocks $S_{i,j}$, it firstly encrypts $K_{i,j}$ with the bill B_i' (For distinguishing the bills proxy sends to user and the bills CSPs maintain, we denote the bill maintained by one CSP that stored $C_{i,j}$ as B_i' and denote the bill proxy sends to user as B_i) and sends the two-tuples $\{C_{i,j}, S_{i,j}^{**}\}$ to the user, where $S_{i,j}^{**}$ is the encrypted form of key block $S_{i,j}$. We follow the idea of identity-based encryption (IBE) to design the encryption/decryption algorithm for encrypted key blocks, shown as follows.

4.5.1. Key Blocks Reencryption. For one of the t CSPs that store $C_{i,j}$ and $S_{i,j}^*$, it randomly chooses $q \in Z_p^*$; then the CSP calculates 3 variates $\alpha_{i,j}$, $\mu_{i,j}$, and $\upsilon_{i,j}$ as shown in the following equations:

$$\begin{aligned} \alpha_{i,j} &= \widehat{e}\left(H_b \left(B_i' \right)^{H(\text{uID})}, P_u \right), \\ \mu_{i,j} &= g^q, \\ \upsilon_{i,j} &= S_{i,j}^* \oplus H_l \left(\alpha_{i,j}^q \right), \end{aligned} \quad (12)$$

where the six parameters $\{\widehat{e}, H_b(\cdot), H_l(\cdot), H(\cdot), g, P_u\}$ are all predefined in the *Setup* phase of our scheme.

Note that the reencrypted form of key block $S_{i,j}$ (i.e., $S_{i,j}^{**}$) is denoted as the two-tuples $\{\mu_{i,j}, \upsilon_{i,j}\}$ and each of t CSPs will send it together with encrypted data block $C_{i,j}$ to the user. Due to different identifications, different users would receive different form of $S_{i,j}^{**}$.

4.5.2. Key Blocks Decryption. Upon user receiving all t encrypted key blocks (i.e., $\{\mu_{i,j}, \upsilon_{i,j}\}$) from t CSPs (here, we assume that user could identify the identifications of all t CSPs who send data), he/she could decrypt them with his/her secret key, $r \cdot H(\text{uID})$, shown as follows:

$$S_{i,j}^* = \upsilon_{i,j} \oplus H_l \left(\widehat{e}\left(H_b \left(B_i \right)^{r\cdot H(\text{uID})}, \mu_{i,j} \right) \right), \quad (13)$$

where $\upsilon_{i,j}$, $\mu_{i,j}$ are included in the encrypted form of $S_{i,j}$, r is the secret value shared between owner and the user, B_i is the bill for C_i sent by the proxy, and $H_l(\cdot)$, $H_b(\cdot)$ are public parameters we prepared in our *Setup* phase. Finally, user could decrypt $S_{i,j}^*$ with L.

$$S_{i,j} = S_{i,j}^* \oplus L. \quad (14)$$

4.5.3. File Decryption. For obtaining plaintext form of original data file F_i, after decrypting all t encrypted key blocks, the data user could obtain the original unencrypted encryption

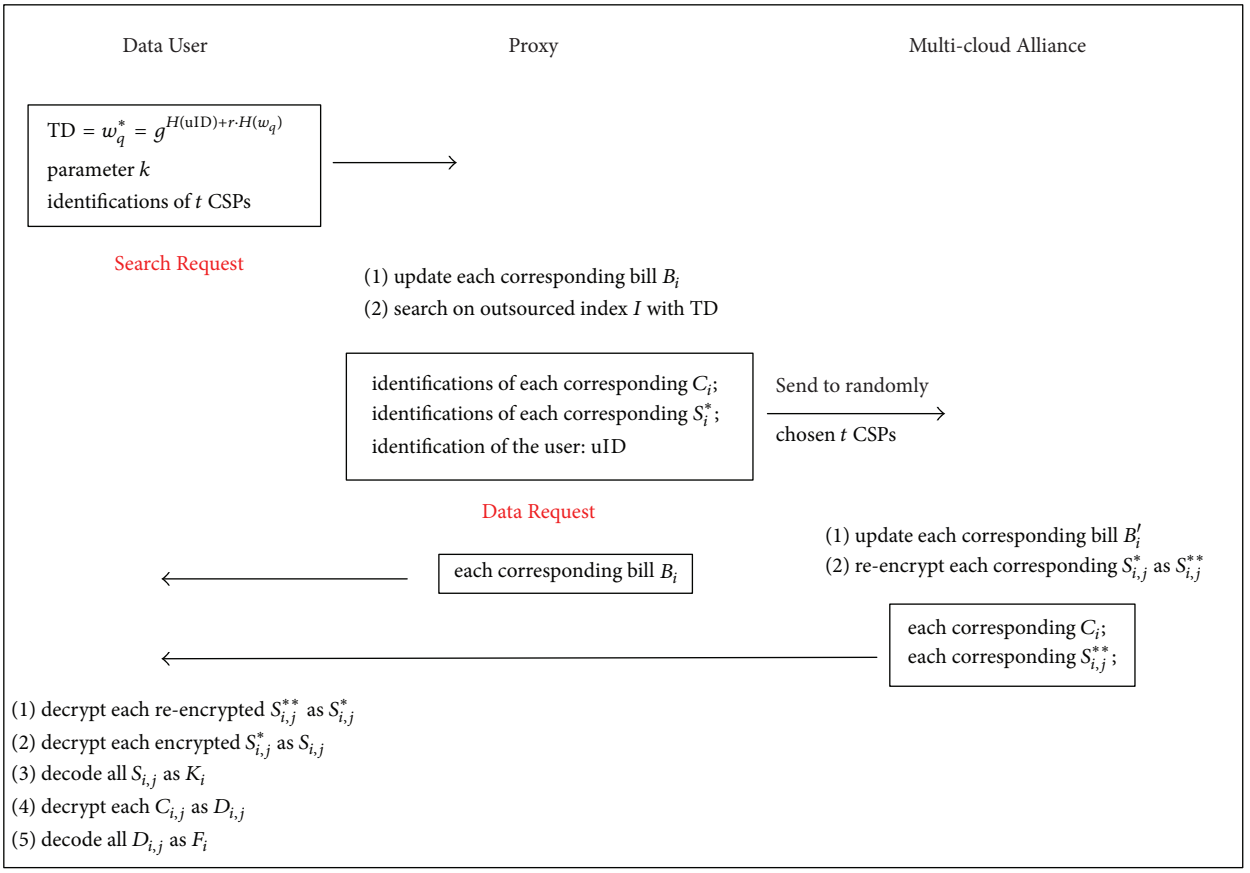


Figure 2: The processes of search and download encrypted data/key blocks.

key K_i by decoding t unencrypted key blocks with dispersal matrix A. Then user decrypts each of encrypted data blocks in C_i and decode them in the same way. Now user could obtain the plaintext form of original data file F_i. The detailed processes of searching and downloading encrypted data/key blocks are summarized in Figure 2.

4.6. Correctness Analysis. In this section, we will show why data user could correctly decrypt the encrypted key blocks from each CSP if and only if the two bills of one corresponding data file (i.e., the bill from proxy and the bill form CSP) are exactly equal to each other. Without loss of generality, taking the key block $S_{i,j}$ as an example, according to the encryption given by (12) and the decryption process shown in (13) and (14), we can prove that $S_{i,j}$ could be correctly decrypted when $B_i = B_i'$, as shown as follows.

Firstly, the reencrypted form of $S_{i,j}(S_{i,j}^{**})$ is shown as follows:

$$\begin{aligned} S_{i,j}^{**} &= \left\{\mu_{i,j}, \upsilon_{i,j}\right\} \\ &= \left\{g^q, S_{i,j}^* \oplus H_l\left(\widehat{e}\left(H_b\left(B_i'\right)^{H(\text{uID})}, P_u\right)^r\right)\right\} \\ &= \left\{g^q, S_{i,j} \oplus L \oplus H_l\left(\widehat{e}\left(H_b\left(B_i'\right)^{H(\text{uID})}, P_u\right)^r\right)\right\}. \end{aligned} \tag{15}$$

Secondly, if and only if $B_i = B_i'$, we can correctly decrypt $S_{i,j}^*$, as shown as follows:

$$\begin{aligned} &\upsilon_{i,j} \oplus H_l\left(\widehat{e}\left(H_b\left(B_i\right)^{r\cdot H(\text{uID})}, \mu_{i,j}\right)\right) \\ &\quad = S_{i,j}^* \oplus H_l\left(\alpha_{i,j}^q\right) \oplus H_l\left(\widehat{e}\left(H_b\left(B_i\right)^{r\cdot H(\text{uID})}, \mu_i\right)\right) \\ &\quad = S_{i,j}^* \oplus H_l\left(\widehat{e}\left(H_b\left(B_i'\right)^{H(\text{uID})}, P_u\right)^q\right) \\ &\qquad \oplus H_l\left(\widehat{e}\left(H_b\left(B_i\right)^{r\cdot H(\text{uID})}, g^q\right)\right) \\ &\quad = S_{i,j}^* \oplus H_l\left(\widehat{e}\left(H_b\left(B_i'\right)^{H(\text{uID})}, P_u\right)^q\right) \\ &\qquad \oplus H_l\left(\widehat{e}\left(H_b\left(B_i\right)^{H(\text{uID})}, P_u\right)^q\right) = S_{i,j}^*. \end{aligned} \tag{16}$$

Obviously, if we successfully decrypt $S_{i,j}^*$, we can correctly decrypt $S_{i,j}^*$ as secret binary number L; we omit the correctness proof here.

4.7. Summary of Scheme. Now we summarize our proposed scheme in terms of reliability and flexibility. The detailed analyses about security and efficiency are shown in Section 5.

4.7.1. Reliability. The (n, t) Reed-Solomon (RS) erasure-correcting code guarantees the reliability of our system. If

there are some errors, such as data loss (key blocks or data blocks) occurring in some CSPs, the properties of erasure-correcting code could help the system to recover the complete data in time. In practice, when the owner deploys cloud services, some mature technology like virtual machines (VM) replication also could be used to enhance the data reliability, and the efficient issues like huge network resource consumption in failure recovery process have been deeply researched [21] to improve the quality of data service. That is to say, many technological means could guarantee the reliability of our system.

4.7.2. Flexibility. The owner outsources the (1) encrypted data blocks, (2) encrypted key blocks, and (3) encrypted index to the multicloud alliance and proxy, respectively. Therefore, data user could search what he/she needs in outsourced cloud data and download them as well as corresponding encrypted key blocks and finally decrypt the key and data. The whole data sharing process does not need the owner to remain online after finishing the upload process. It obviously brings significant flexibility to data owner.

5. Security and Performance Analysis

5.1. Security Analysis. We analyze the security of our system about the 3 aspects we mentioned in Section 2.4.1: data confidentiality, index and query privacy, and confirmable bills.

5.1.1. Data Confidentiality. The AES encryption algorithm is utilized to encrypt the outsourced data in our scheme. In general, we consider that the AES algorithm with random 128-bit key is secure; that is, the only way to destroy the data confidentiality is to deduce the 128-bit encryption (decryption) key. Except from completely guessing the key, there are two other kinds of attacks that adversary can carry out based on the threat model we defined in Section 2.3. Assuming that the adversary wants to deduce one of the decryption keys K_i for the encrypted data block collection C_i, the specific attacks can be shown as follows:

(i) *Attack-1*: after receiving all encrypted data blocks and encrypted key blocks, there are more than t CSPs colluding together for deducing one of the encryption keys K_i; that is, the adversary can totally obtain more than t encrypted data/key blocks

(ii) *Attack-2*: when the t CSPs send t encrypted data blocks $(\{\ldots, C_{i,j}, \ldots\})$ and t reencrypted key blocks $(\{\ldots, S_{i,j}^{**}, \ldots\})$ to one user, other users would like to eavesdrop the sending information to deduce K_i and further obtain the plaintext form of data file F_i

As for *Attack-1*, the secret binary number L is used to protect each encrypted key block $S_{i,j}^{*}$. According to the encryption process shown in (8), since the users in our system would not collude with any CSP, the adversary can only deduce the 128-bit L based on completely guessing way. Such difficulty is equal to guessing AES key; that is, our system is secure in this case.

As for *Attack-2*, all users have the same secret value L which can successfully decrypt the encrypted key blocks (e.g., $S_{i,j}^{*} \rightarrow S_{i,j}$), but the "curious" ones have to decrypt the reencrypted key blocks firstly (e.g., $S_{i,j}^{**} \rightarrow S_{i,j}^{*}$). In our scheme, the secure identity-based encryption algorithm proposed by Boneh and Franklin [22] is used to protect the confidentiality of reencrypted key blocks (i.e., $S_{i,j}^{**}$). Recalling the reencryption process we proposed in (12), the only difference compared with classical IBE algorithm in [22] is that we import the hash value of bill B_i into encryption, but it does not influence the security of encryption algorithm. The detailed security analysis of IBE is shown in literature [22], so we omit it here.

The above two kinds of attacks are the two strongest attacks under the threat model we defined; that is, if our system can successfully resist the above two attacks, other weaker attacks also cannot work well. In summary, the data confidentiality can be protected well in our scheme.

5.1.2. Index/Query Privacy. As for index privacy, since we only encrypt the keywords in the index, so if we can prove the security of index keywords encryption algorithm, we say that the index privacy is well protected. Recall the form of encrypted keywords in index:

$$w^{*} = \left(g^{H(L\|w)} \bmod p\right)^{-1}. \tag{17}$$

Firstly, the discrete logarithm problem in Z_p with large prime p prevents the proxy from knowing the hash value $H(L \| w)$, even if the proxy could guess the encrypted keyword by trying all keywords in dictionary, but under the security assumption, in our threat model, users would not collude with proxy or any other adversary, which means that the proxy cannot obtain the secret value L. Therefore the proxy cannot deduce any plaintext information about encrypted keywords in index.

As for query privacy, before we prove the security of query keywords encryption, we first define a traditional challenger-adversary game. The challenger firstly gives a public key pk to the adversary and then the adversary randomly chooses two query keywords w_0 and w_1 and sends them to the challenger. Upon receiving w_0 and w_1, the challenger randomly chooses w_s ($s = 0$ or $s = 1$) with equal probability 1/2 and then sends $w_s^{*} = \mathrm{Enc}(w_s, pk)$ to the adversary. The adversary tries to guess the value of s and outputs its guess s'. The adversary's advantage of this game is the probability $\Pr[s = s'] - 1/2$. Generally, the encryption algorithm is regarded as semantically secure against a chosen plaintext attack (CPA) if the adversary's advantage is negligible.

Theorem 1. *If DDH (Decisional Diffie-Hellman) problem is hard, our keywords encryption is semantically secure.*

Proof. We assume that there is a polynomial time algorithm $\mathscr{A}$ that has a nonnegligible advantage ε as the adversary in the game described above. Then we define a DDH adversary $\mathscr{B}$ that could access $\mathscr{A}$ and achieves a nonnegligible advantage. $\mathscr{B}$ is given the tuple $(g, g^{r}, g^{H(L\|w)}, T)$ as input. Now, $\mathscr{B}$ tries

to guess whether T is equal to $g^{r \cdot H(L\|w)}$ or T is a random value; we denote $\gamma = 0$ as $T = g^{r \cdot H(w)}$ and $\gamma = 1$ as T is random.

Firstly, $\mathcal{B}$ gives the tuple (g, g^r) to $\mathcal{A}$ as input and then $\mathcal{A}$ randomly chooses keywords w_0 and w_1 and sends them to $\mathcal{B}$.

Secondly, $\mathcal{B}$ sets a bit s randomly and sends ciphertext $C_1 = g^r$, $C_2 = g^{H(L\|w_s) \cdot T}$ to $\mathcal{A}$.

Thirdly, $\mathcal{A}$ sends $\mathcal{B}$ a bit s', which is its guess for s. $\mathcal{B}$ guesses that $\gamma = 0$ if and only if $s' = s$. □

Now we analyze the two situations of γ:

(i) If $\gamma = 0$ (i.e., $T = g^{r \cdot H(L\|w_s)}$), then $C_2 = g^{H(L\|w_s)} \cdot T$ is a valid encryption for keywords encryption. In this case, $\mathcal{A}$ will guess correctly with probability $1/2 + \varepsilon$.

(ii) If $\gamma = 1$ (i.e., T is random), the keywords encryption adversary $\mathcal{A}$ receives the value $g^{H(L\|w_s)} \cdot T^{H(L\|w_s)}$, where T is random value. In this case, s is hidden to $\mathcal{A}$, so the probability that $\mathcal{A}$ will guess it is simply 1/2.

Therefore, the probability that $\mathcal{B}$ could successfully solve the DDH problem is shown as follows:

$$\Pr = \frac{1}{2}\left(\frac{1}{2} + \varepsilon\right) + \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{2} + \frac{\varepsilon}{2}. \tag{18}$$

Since ε is nonnegligible, this demonstrates that $\mathcal{B}$ violates the assumption that DDH is hard.

5.1.3. Confirmable Bills. The method that regards the exchange bill as a part of encryption key for reencrypting key blocks protects the bills from being forged. From the analysis we mentioned in Section 4.6, we find that the user could not decrypt the reencryption key blocks successfully with a high probability under the threat model we defined in Section 2.3. Specifically, in our scheme, user randomly chooses t CSPs from the whole multicloud alliance for retrieving encryption key K_i. According to the security assumption we defined above, a small part of CSPs can be completely trusted; that is to say, if we choose proper parameters for n and t, it is highly possible that there is at least one trusted CSP in the randomly chosen t CSPs. As we say in Section 4.5.2, the user could identify the identifications of all t CSPs that send data, and user would not collude with proxy; therefore these t CSPs are randomly chosen by user; there does not exist the situation where proxy chooses t compromised CSPs that are not wholly chosen by user. Hence, even if the proxy colludes with some CSPs to forge bill B_i for C_i, there is at least one reencrypted key block that cannot be correctly decrypted by data user (see (16)), not to mention to obtain the correct unencrypted data file F_i. Of course, once the user fails to decrypt the data file, he/she will report an error to the system, which could help the owner detect whether the proxy or some CSPs forge the bills immediately.

From the above analysis, the probability that all randomly chosen CSPs are semitrusted (i.e., adversary could successfully deceive the owner in terms of forging the one of the bills) can be calculated as follows:

$$\Pr = \frac{C_{n \cdot (1-P_T)}^t}{C_n^t}, \tag{19}$$

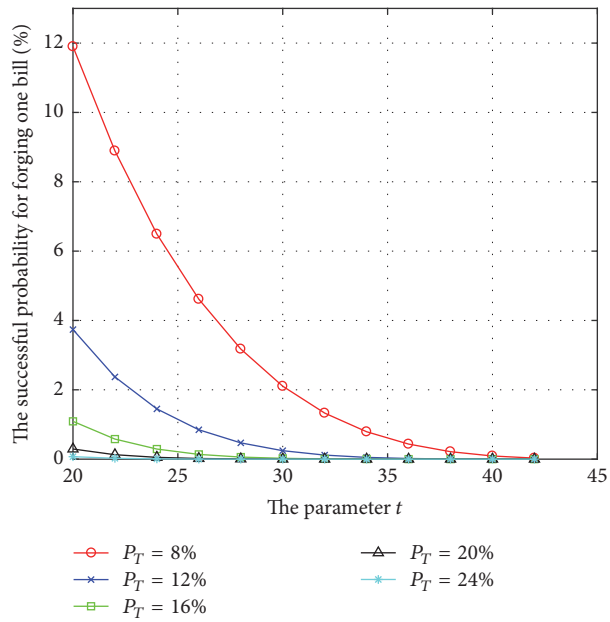


FIGURE 3: The successful probability for forging one bill.

where P_T denotes the proportion of trusted CSPs in whole multicloud alliance, n is the total number of CSPs in the whole multicloud alliance, and t is the encoding parameter that owner chooses. We assume that there are a total of 50 CSPs in the multicloud alliance and then analyze the successful probability for forging one bill based on different P_T and parameter t as Figure 3 shows.

From Figure 3, we can see that larger t and P_T will lead to lower probability that adversary can successfully deceive the owner for one forged bill. When $t \geq 28$ and $P_T \geq 16\%$, the probability that owner suffers such collusive attack is negligible. However, we have to say that even if the owner chooses proper t and the multicloud alliance is relatively secure (i.e., higher P_T), there still exists the situation where the randomly chosen t CSPs are all semitrusted. In this case, all the key blocks would be correctly decrypted. In fact, there is a tradeoff between the high reliability and high security (the "security" here particularly means the correctness of exchange bills) of our data sharing scheme. On one hand, an extreme situation is to choose $n = t$ as the encoding parameter; under our threat model that a small part of CSPs is wholly trusted, the attack above will never be carried out successfully. But if there are some errors like data losses in some CSPs, the system could not give the complete original data file to the user. On the other hand, if t is far less than n, the probability that proxy successfully colludes with the semitrusted CSPs will be high. In a word, the contradiction between reliability and security is still not fully solved in this paper. We will continue to research this problem in the future.

5.2. Performance Analysis. We implemented the proposed scheme using by C language in Windows 10 operation system and tested its efficiency in real-world data files: the Request for Comments (RFC) [23]. Since the upload process executed

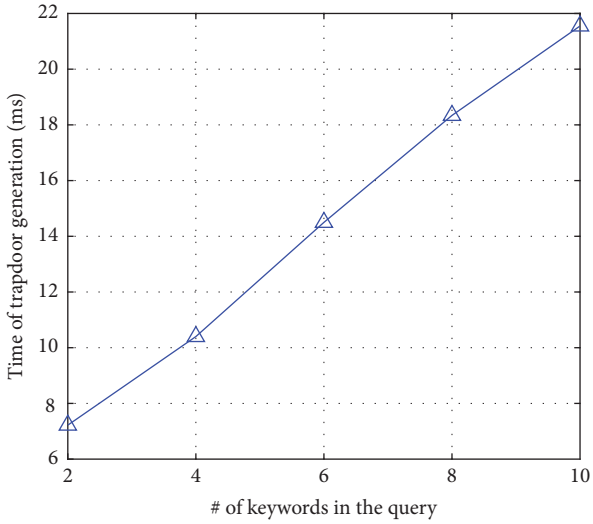


Figure 4: The time of trapdoor generation.

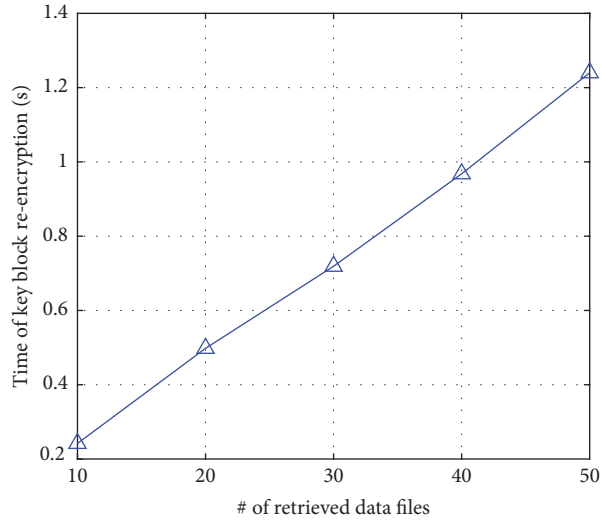


Figure 5: The time of key block reencryption.

by owner can be regarded as the precomputation that will be executed only once, we only focus on the time cost of users and the communication cost of the proxy. Specifically, it includes (1) the time of trapdoor generation, (2) the time of search, (3) the time of key blocks reencryption, (4) the time of reencrypted key blocks decryption, and (5) the traffic analysis of proxy. Of course, the energy consumption is also an important indicator to evaluate the performance of such data-intensive cloud service [24], but due to the restricted experimental conditions, we would not discuss it in this paper. Our experiments are conducted on a system with an Intel(R) Core (TM) i5-3470 processor running at 3.74 GHz, 16.00 GB of RAM, and a 7200 RPM Western Digital 1 TB Serial ATA drive. All results represented the mean of 20 trials.

5.2.1. Time of Trapdoor Generation. According to (9), the encryption algorithm for generating trapdoor for each query keyword totally includes two modular exponentiation operations; that is, the time complexity of trapdoor generation in our schemes is $O(N^2)$, where N denotes the size of dictionary. Figure 4 shows that the time of trapdoor generation increases linearly with N increasing.

5.2.2. Time of Search. The proxy should traverse the inverted index I and rank the data files according to the number of query keywords in the file. Figure 6(a) shows that, with the same number of query keywords and data files, the search time increases almost linearly with the size of dictionary increasing. Figure 6(b) shows that, with the same number of data files and the same size of dictionary, the search time increases almost linearly with the number of query keywords increasing. But the search time cost is not significantly affected by the number of data files due to the structure of our inverted index; specifically, after locating query keywords, all candidates are filtered out for ranking, while the rank operation will not cost too much time, as shown in Figure 6(c).

5.2.3. Time of Key Blocks Reencryption. Each of the t randomly chosen CSPs will reencrypt one encrypted key block for one data file as given in (12). Since the t CSPs reencrypt encrypted key blocks simultaneously, the waiting time for user is only related to the number of data files that will be downloaded by data user (top-k ranked results). That is, the time complexity of key blocks reencryption is $O(k)$. As shown in Figure 5, with k increasing, the time of reencryption increases almost linearly.

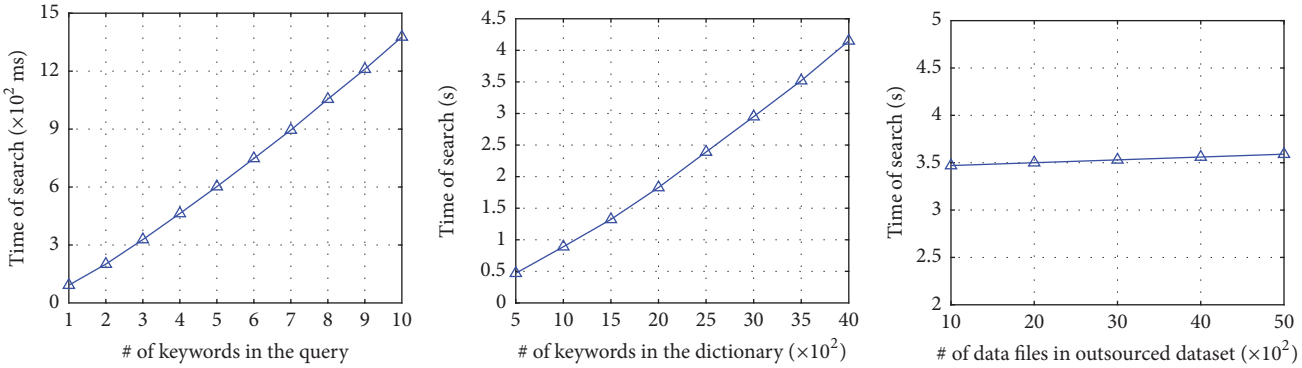
5.2.4. Time of Reencrypted Key Blocks Decryption. Upon receiving all reencrypted key blocks corresponding to retrieved k data files (i.e., encrypted data block collections), the user needs to decrypt them as given in (13). Obviously, the user totally decrypts $t \cdot k$ reencrypted key blocks. Figure 7 shows that, with the increasing of parameter t or k, the time of decryption increases almost linearly.

5.2.5. Traffic Analysis of Proxy. Since proxy plays as "Interaction Center" in our system, it is necessary to analyze its traffic. The whole communication process of proxy can be divided into two phases. Specifically, the first one is "upload"; the proxy only receives the encrypted index from data owner; the second one is "download"; the proxy receives search request from data user, sends data request to multicloud alliance, and sends all corresponding bills to data user.

According to the above two phases, the complexity of communication of proxy can be shown in Table 2. We assume that the encrypted keyword is 128 bytes, the identification of encrypted data/key block or CSP is about 128 bytes, and the size of a bill is about 512 bytes. Then we set the number of documents $m = 5000$, the number of CSPs in whole multicloud alliance (i.e., the encode parameter n) is 50, the encode parameter $t = 35$, the size of dictionary $|W| = 12000$, the number of keywords in query $|Q| = 5$, and the search parameter $k = 50$. Based on such parameter settings, we can obtain that the whole communication cost of proxy in upload

Table 2: The traffic analysis of proxy.

	Data owner	Data user	Multicloud
Upload	$O(\|W\| + m + 2mn)$	0	0
Download	0	$O(t + \|Q\| + k)$	$O(2kt)$

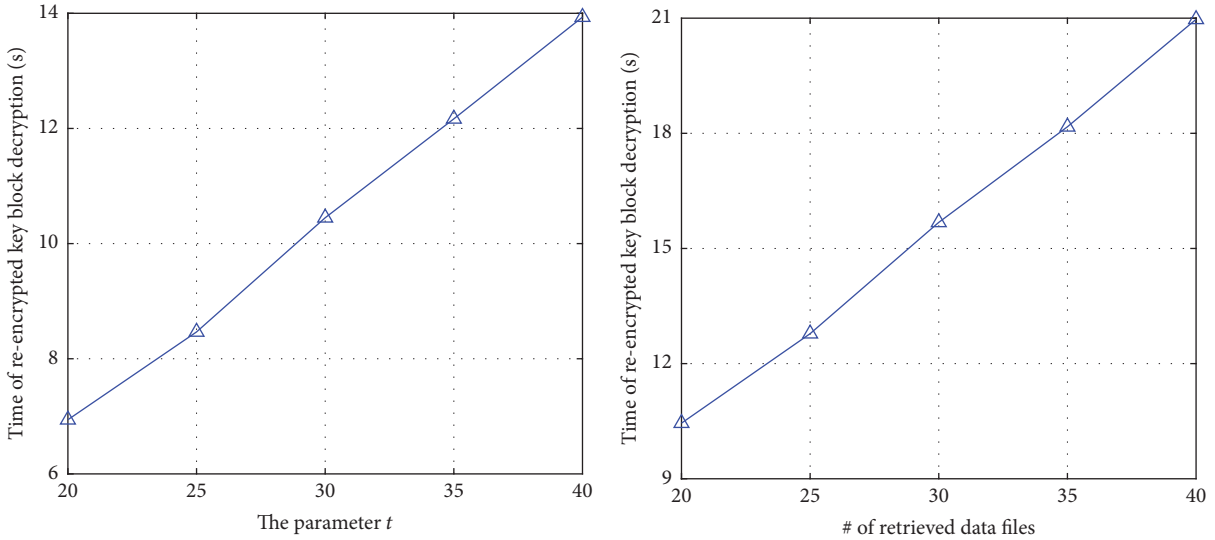


(a) For different number of query keywords with the same size of dictionary, $|W| = 4000$, and for the same number of query data files, $n = 2000$

(b) For different size of dictionary with the same number of query keywords, $|Q| = 5$, and for the same number of query data files, $n = 2000$

(c) For different number of data files with the same size of dictionary, $|W| = 4000$, and for the same number of query keywords, $|Q| = 5$

Figure 6: The time of search.



(a) For different parameter t with the same number of retrieved data files, $k = 20$

(b) For different number of retrieved data files with the same parameter t, $t = 30$

Figure 7: The time of reencrypted key blocks decryption.

phase is about 63 M, and in download phase it is about 0.45 M. Although the communication cost in upload is a little big, it is one-time process for uploading index. In a word, even the communication resource-constrained device could play as proxy.

6. Related Work

6.1. Secure Data Sharing with IBE and PRE. The concept of identity-based encryption (IBE) was first introduced by Shamir [25]; then Boneh and Franklin [22] proposed the first secure and practical IBE scheme with the computable bilinear

map. Then Canetti et al. [26] designed the first construction for IBE which was provably secure without the random oracle model. Moving a step forward, in 2004, Boneh and Boyen [27] proposed two efficiency improved IBE schemes under the "selective-ID" model proposed in [26]. The IBE approach provides an efficient cryptographic system where the public key can be any arbitrary string and the secret key is extracted from a trusted party called a private key generator (PKG). Motivated by its significant advantage that owner could release from heavy load of managing certificates, Kackeri [9] utilized IBE in the secure data distribution system that only users with authorized identifications could access the encrypted data.

Another popular secure data sharing method is proxy-based reencryption (PRE), which was first proposed by Blaze et al. [28]. With a proxy server, the PRE primitive can transfer a ciphertext designated for one user to another ciphertext designated for another user without knowing the plaintext content of data. After that, many PRE schemes have been proposed for the improved security or efficiency [29–31]. In 2006, Ateniese et al. [32] then improved the concept of PRE and employed it for remote data storage. In their scheme, owner encrypts his/her files and outsources them to proxy. The proxy can transfer the owner's ciphertext to the ciphertext of the requester if and only if he/she has obtained a reencryption key from the owner. Then, combining technologies of IBE and PRE, Green and Ateniese [33] introduced the concept of identity-based proxy reencryption (IBPRE) scheme and then the IBPRE scheme is further developed in terms of security and functionality enhancement [34, 35]. The proxy in IBPRE schemes also plays a similar role to traditional PRE, but the transfer key is related to the identities of owners and users.

However, when we utilize the method of IBE or PRE for data sharing system, it is necessary for data owner to remain online all the time (such as generating reencryption key for user in PRE schemes) or know the specific information about the data user (such as encrypting data according to the user's identification in IBE schemes).

6.2. Secure Data Sharing with ABE. To address the mentioned problems of IBE and PRE, Goyal et al. [36] firstly proposed Key-Policy Attribute-Based Encryption, where the ciphertexts are labeled with sets of attributes and private keys are associated with access structures that control which ciphertexts a user is able to decrypt. Then, a number of works used ABE to realize fine-grained access control for outsourced data [12, 13, 37]. There is no doubt that the ABE schemes improve the scalability of data sharing system, so that the data owner could encrypt outsourced data for multiple users with one encryption key. Besides, ABE schemes also provide great flexibilities since only attributes of the user set instead of specific identity (or public key) need to be known by owner. But there are still some limits for directly using ABE scheme in our "Paid Data Sharing" model: one is that may be the owner even does not know the user's attributes for one data file; another is that the authority in ABE system suffers from the key escrow problem [12, 16], since the authority can access all the encrypted files.

Besides, Hiremath and Annapurna [38] discussed the framework and security issues at collaboration in multicloud computing environments; inspired by the "proxy as service" model they proposed, we design a distributed data/key storage system architecture in this paper.

7. Conclusion

In this paper, for the first time, we explore the problems in a practical secure data sharing scenario. Specifically, the data owner does not know which data users (even their attributes) will download and decrypt which outsourced data files of him/her, while the owner cannot always remain online and there is no trusted third party in the system. We utilize the secret sharing approach to split and encrypt the decryption key and then outsource it to multiple cloud service providers (CSPs). Such distributed key management not only protects the data confidentiality but also guarantees that if there are not enough CSPs collude with each other, the adversaries cannot successfully forge the bills for data transaction. In order to improve the reliability of such data service, we exploit the erasure-correcting code for distributing file to multiple CSPs. Even if there are data losses occurring in some CSPs, the data redundancies that erasure-correcting code produced can help to recover the complete outsourced data immediately. A semitrusted proxy that offered a secure and efficient keyword-based query function enables the data user to search what he/she needs in outsourced data files before downloading. Experiments on real-world dataset demonstrate the time efficiency of our proposed scheme.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

References

[1] S. Lian and Y. Duan, "Smoothing of the lower-order exact penalty function for inequality constrained optimization," *Journal of Inequalities and Applications*, Paper No. 185, 12 pages, 2016.

[2] J. Zhang, B. Qu, and N. Xiu, "Some projection-like methods for the generalized Nash equilibria," *Computational optimization and applications*, vol. 45, no. 1, pp. 89–109, 2010.

[3] M. Wang, L. Song, and G.-l. Tian, "SCAD-penalized least absolute deviation regression in high-dimensional models," *Communications in Statistics—Theory and Methods*, vol. 44, no. 12, pp. 2452–2472, 2015.

[4] P. Wang and L. Zhao, "Some geometrical properties of convex level sets of minimal graph on 2-dimensional Riemannian manifolds," *Nonlinear Analysis*, vol. 130, pp. 1–17, 2016.

[5] K. Ren, C. Wang, and Q. Wang, "Security challenges for the public cloud," *IEEE Internet Computing*, vol. 16, no. 1, pp. 69–73, 2012.

[6] Z. Kong, S. Yang, F. Wu, S. Peng, L. Zhong, and L. Hanzo, "Iterative Distributed Minimum Total MSE Approach for Secure Communications in MIMO Interference Channels," *IEEE*

Transactions on Information Forensics and Security, vol. 11, no. 3, pp. 594–608, 2016.

[7] S. Kamara and K. Lauter, "Cryptographic cloud storage," in *Financial Cryptography and Data Security*, vol. 6054 of *Lecture Notes in Computer Science*, pp. 136–149, Springer, Berlin, Germany, 2010.

[8] J. Han, W. Susilo, and Y. Mu, "Identity-based data storage in cloud computing," *Future Generation Computer Systems*, vol. 29, no. 3, pp. 673–681, 2013.

[9] R. R. Kackeri, *Identity-based encryption system for secure data distribution*, U.S. Patent No. 7,003,117, 2006.

[10] L. Xu, X. Wu, and X. Zhang, "CL-PRE: A certificateless proxy re-encryption scheme for secure data sharing with public cloud," in *Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security, (ASIACCS '12)*, pp. 87-88, Republic of Korea, May 2012.

[11] Q. Liu, G. Wang, and J. Wu, "Time-based proxy re-encryption scheme for secure data sharing in a cloud environment," *Information Sciences*, vol. 258, pp. 355–370, 2014.

[12] S. Yu, C. Wang, K. Ren, and W. Lou, "Attribute based data sharing with attribute revocation," in *Proceedings of the 5th ACM Symposium on Information, Computer and Communication Security, (ASIACCS '10)*, pp. 261–270, April 2010.

[13] M. Yang, F. Liu, J.-L. Han, and Z.-L. Wang, "An efficient attribute based encryption scheme with revocation for outsourced data sharing control," in *Proceedings of the 1st International Conference on Instrumentation and Measurement, Computer, Communication and Control, IMCCC '11*, pp. 516–520, China, October 2011.

[14] N. Cao, C. Wang, M. Li, K. Ren, and W. Lou, "Privacy-preserving multi-keyword ranked search over encrypted cloud data," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 1, pp. 222–233, 2014.

[15] Z. Xia, X. Wang, X. Sun, and Q. Wang, "A Secure and Dynamic Multi-Keyword Ranked Search Scheme over Encrypted Cloud Data," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 2, pp. 340–352, 2016.

[16] M. Li, S. Yu, Y. Zheng, K. Ren, and W. Lou, "Scalable and secure sharing of personal health records in cloud computing using attribute-based encryption," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 1, pp. 131–143, 2013.

[17] J. S. Plank and Y. Ding, "Note: Correction to the 1997 tutorial on Reed-Solomon coding," Tech. Rep. CS-03-504, University of Tennessee, 2003.

[18] J. Li, Q. Wang, C. Wang, N. Cao, K. Ren, and W. Lou, "Fuzzy keyword search over encrypted data in cloud computing," in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM '10)*, pp. 441–445, IEEE, San Diego, Calif, USA, March 2010.

[19] W. Zhang, S. Xiao, Y. Lin, T. Zhou, and S. Zhou, "Secure ranked multi-keyword search for multiple data owners in cloud computing," in *Proceedings of the 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, DSN '14*, pp. 276–286, USA, June 2014.

[20] D. Cash, P. Grubbs, J. Perry, and T. Ristenpart, "Leakage-abuse attacks against searchable encryption," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, CCS '15*, pp. 668–679, USA, October 2015.

[21] A. Zhou, S. Wang, B. Cheng et al., "Cloud Service Reliability Enhancement via Virtual Machine Placement Optimization," *IEEE Transactions on Services Computing*, vol. 10, no. 6, pp. 902–913, 2017.

[22] D. Boneh and M. Franklin, "Identity-based encryption from the Weil pairing," in *Advances in cryptology—CRYPTO*, vol. 2139 of *Lecture Notes in Computer Science*, pp. 213–229, Springer, Berlin, 2001.

[23] (2016). Request for comments. [Online]. Available: http://www.rfc-editor.org/index.html.

[24] S. Wang, A. Zhou, C.-H. Hsu, X. Xiao, and F. Yang, "Provision of Data-Intensive Services Through Energy-and QoS-Aware Virtual Machine Placement in National Cloud Data Centers," *IEEE Transactions on Emerging Topics in Computing*, vol. 4, no. 2, pp. 290–300, 2016.

[25] A. Shamir, "Identity-based cryptosystems and signature schemes," *Workshop on the Theory and Application of Cryptographic Techniques—CRYPTO*, vol. 84, 1984.

[26] R. Canetti, S. Halevi, and J. Katz, "A forward-secure public-key encryption scheme," in *Advances in cryptology—EUROCRYPT*, vol. 2656 of *Lecture Notes in Computer Science*, pp. 255–271, Springer, Berlin, 2003.

[27] D. Boneh and X. Boyen, "Efficient selective-ID secure identity-based encryption without random oracles," in *Advances in cryptology—EUROCRYPT*, vol. 3027 of *Lecture Notes in Computer Science*, pp. 223–238, Springer, Berlin, 2004.

[28] M. Blaze, G. Bleumer, and M. Strauss, "Divertible protocols and atomic proxy cryptography," in *Advances in cryptology—EUROCRYPT'98 (Espoo)*, vol. 1403 of *Lecture Notes in Computer Science*, pp. 127–144, Springer, Berlin, 1998.

[29] T. Matsuda, R. Nishimaki, and K. Tanaka, "CCA proxy re-encryption without bilinear maps in the standard model," in *Public Key Cryptography—PKC*, vol. 6056 of *Lecture Notes in Comput. Sci.*, pp. 261–278, Springer, Berlin, 2010.

[30] S. S. Chow, J. Weng, Y. Yang, and R. H. Deng, "Efficient unidirectional proxy re-encryption," in *International Conference on Cryptology in Africa*, vol. 6055 of *Lecture Notes in Comput. Sci.*, pp. 316–332, Springer, Berlin, 2010.

[31] J. Weng, Y. Zhao, and G. Hanaoka, "On the security of a bidirectional proxy re-encryption scheme from PKC 2010," in *Public Key Cryptography—PKC*, vol. 6571 of *Lecture Notes in Comput. Sci.*, pp. 284–295, Springer, Heidelberg, 2011.

[32] G. Ateniese, K. Fu, M. Green, and S. Hohenberger, "Improved proxy re-encryption schemes with applications to secure distributed storage," *ACM Transactions on Information and System Security*, vol. 9, no. 1, pp. 1–30, 2006.

[33] M. Green and G. Ateniese, "Identity-Based Proxy Re-encryption," in *International Conference on Applied Cryptography and Network Security*, Lecture Notes in Computer Science, pp. 288–306, Springer-Verlag, 2007.

[34] C. K. Chu and W. G. Tzeng, "Identity-based proxy re-encryption without random oracles," in *International Conference on Information Security*, pp. 189–202, Springer-Verlag, 2007.

[35] L. Wang, L. Wang, M. Mambo, and E. Okamoto, "Identity-Based Proxy Cryptosystems with Revocability and Hierarchical Confidentialities," in *International Conference on Information and Communications Security*, vol. 6476 of *Lecture Notes in*

Computer Science, pp. 383–400, Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[36] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proceedings of the 13th ACM Conference on Computer and Communications Security (CCS '06)*, pp. 89–98, November 2006.

[37] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *Proceedings of the IEEE Symposium on Security and Privacy (SP '07)*, pp. 321–334, May 2007.

[38] M. M. Hiremath and P. P. Annapurna, "Collaboration in multi-cloud computing environments: Framework and security issues," *International Journal of Computer Science and Information Technologies*, vol. 6, no. 3, pp. 2859–2862, 2015.