Wireless Communications and Mobile Computing

Big IoT Data Analytics in Fog Computing

Lead Guest Editor: Xuyun Zhang
Guest Editors: Yongrui Qin, Deepak Puthal, and Xiaobing Wu

Big IoT Data Analytics in Fog Computing

Wireless Communications and Mobile Computing

Big IoT Data Analytics in Fog Computing

Lead Guest Editor: Xuyun Zhang
Guest Editors: Yongrui Qin, Deepak Puthal, and Xiaobing Wu

Editorial Board

Contents

Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 9596141, 10 pages
https://doi.org/10.1155/2018/9596141

Research Article

Energy-Efficient User Association with Congestion Avoidance and Migration Constraint in Green WLANs

Wenjia Wu, Junzhou Luo, Kai Dong, Ming Yang, and Zhen Ling

School of Computer Science and Engineering, Southeast University, Nanjing 211189, China

Correspondence should be addressed to Wenjia Wu; wjwu@seu.edu.cn

Received 12 February 2018; Accepted 13 May 2018; Published 28 June 2018

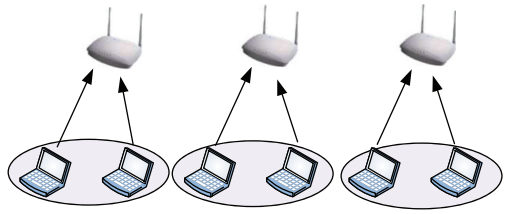
Academic Editor: Xiaobing Wu

Green wireless local area networks (WLANs) have captured the interests of academia and industry recently, because they save energy by scheduling an access point (AP) on/off according to traffic demands. However, it is very challenging to determine user association in a green WLAN while simultaneously considering several other factors, such as avoiding AP congestion and user migration constraints. Here, we study the energy-efficient user association with congestion avoidance and migration constraint (EACM). First, we formulate the EACM problem as an integer linear programming (ILP) model, to minimize APs' overall energy consumption within a time interval while satisfying the following constraints: traffic demand, AP utilization threshold, and maximum number of demand node (DN) migrations allowed. Then, we propose an efficient migration-constrained user reassociation algorithm, consisting of two steps. The first step removes k AP-DN associations to eliminate AP congestion and turn off as many idle APs as possible. The second step reassociates these k DNs according to an energy efficiency strategy. Finally, we perform simulation experiments that validate our algorithm's effectiveness and efficiency.

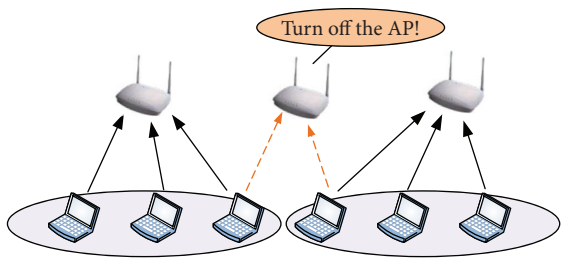
1. Introduction

The IEEE 802.11-based wireless local area network (WLAN) is one of the most popular, widely used wireless access technologies in fog computing [1, 2]. WLANs are especially popular for campuses, enterprise environments, and public hotspots, serving a large number of users while satisfying their continuously growing traffic demands. Network operators usually deploy thousands of access points (APs) to achieve full coverage and provide enough capacity. Although each AP only consumes a little energy, the overall network's energy consumption is significant due to the large-scale deployment. Escalating energy consumption increases greenhouse gas emissions and becomes a threat to environment protection and sustainable development. Moreover, it leads to higher energy costs, which is a critical component of network operational expenditures. Because the APs in a WLAN are overprovisioned with respect to peak traffic demands, but only handle peak traffic for small portions of time, energy is wasted when network traffic is low [3]. Thus, it is unnecessary to keep all resources active, which is why many researchers have pursued energy efficiency in the resource allocation and proposed many green scheduling strategies [4–6], and thus the concept of green WLANs is proposed [7–10]. In green WLANs, APs are scheduled on/off according to time-varying traffic demands, thereby saving energy.

User association is an important issue in WLAN research, and it aims to determine AP selection for each user and optimize APs' resource allocation. Most existing studies focus on performance optimization—such as load balancing, user fairness, and user quality of service (QoS)—and they propose several user association approaches [11–18]. However, little attention has been paid to energy efficiency and the impact of different association solutions on saving energy [19]. Taking the case in Figure 1 as an example, a WLAN with three APs needs to serve six users in the current scenario, and we assume that each AP has the ability to serve three users. When each AP serves two users, respectively, no AP can be turned off. However, when we migrate the users associated with the middle AP to other APs, the middle AP can be turned off. Hence, some of the less-utilized APs can be turned off by adjusting user association, and this saves energy.



(1) Three active APs serve six users.



(2) One AP can be turned off after the adjustment of user association.

Figure 1: An example of energy-efficient user association.

Energy efficiency is a necessity for user association in green WLANs, and finding the right balance is challenging. On one hand, to turn off as many APs as possible, adjusting user association inevitably increases other APs' load, and these other APs could become congested. Therefore, we must find the tradeoff between energy efficiency and congestion avoidance. On the other hand, because of time-varying traffic demands, user association should be adjusted dynamically with several user migrations. These migrations will negatively impact the corresponding users' experience. Hence, we must reduce user migrations in the process of dynamic user association.

In this paper, we investigate the user association problem in green WLANs, to minimize the network's energy consumption while satisfying time-varying traffic demands. Moreover, AP congestion avoidance and user migration constraints are both considered. The main contributions of this paper are as follows:

(i) We investigate the energy-efficient user association with congestion avoidance and migration constraint (EACM) in green WLANs and formulate the EACM problem as an integer linear programming (ILP) model.

(ii) We propose an efficient migration-constrained user reassociation algorithm (MURA), where k AP-DN (DN stands for demand node) associations are removed to eliminate AP congestion and make as many APs as possible idle, and then the k DNs are reassociated according to an energy efficiency strategy.

(iii) We conduct simulation experiments to evaluate the performance of the proposed algorithm, and the results show its efficiency and effectiveness.

The rest of the paper is structured as follows. In Section 2, we discuss the related work. We then present the system model and problem formulation in Section 3. Next, we propose MURA, an efficient algorithm in Section 4. Section 5 evaluates MURA using simulations. Finally, Section 6 concludes the paper.

2. Related Work

User association optimization is of interest to many researchers, because it improves network performance significantly under current transmission technologies.

Over the years, several user association approaches have been proposed. Bejerano et al. [11] propose an efficient algorithm to determine the AP-user associations that achieve load balancing and max-min fair bandwidth allocation, that is, maximizing the minimum user throughput. Li et al. [12] investigate the problem of optimizing AP-user association to achieve proportional fairness in multirate WLANs and present a centralized algorithm to derive the AP-user association. To handle the case of dynamic user membership, a distributed heuristic is proposed to provide an AP selection criterion for newcomers.

Gong et al. [13] study the user association problem for 802.11n WLANs with heterogeneous clients (802.11a/b/g/n) and present solutions whereby each user achieves the throughput proportional to its usable data rate. Yu et al. [14] investigate the problem of optimal user association in wireless mesh networks, considering max-min fairness, proportional fairness, and link interference. The problem is formulated as two-step mixed ILP models, and then two rounding algorithms and their corresponding approximation ratio improvement algorithms are proposed to address the problem.

Wong et al. [15] consider the cost of user migration in the process of user reassociation and propose an efficient approximation algorithm to achieve max-min fairness, while satisfying a certain total user migration cost constraint. Raschellà et al. [16] present a centralized network management framework for user association based on software-defined networking (SDN), where an AP section metric jointly considers the QoS requirements of a user joining the network, the bandwidth efficiency, and the QoS requirements of other users active in the networks.

Chen et al. [17] explore the user association problem in SDN WLANs with some new features, such as centralized association, global network state awareness, seamless handoff, and flow-level association. The problem is formulated as a delay-minimized optimization problem to minimize the interpacket delay of individual flows, and the greedy algorithm and bounded local search algorithm are proposed to solve the problem. Kim et al. [18] consider the link interference and AP load and design a user association mechanism that is based on metrics derived by signal strength variance and a beacon collision rate.

However, all the aforementioned works only take into consideration network performance, which does not translate well when applied in green WLANs. In green WLANs,

there is an energy-saving mechanism that dynamically turns APs on/off to adapt to users' resource demands [7]. Related to green WLANs, there are only a few recent works on user association. These works focus on improving energy efficiency by optimizing user association.

Kumazoe et al. [19] propose a user reassociation scheme where an AP migrates its associated users to another AP and then switches its status to sleep when the AP finds that its utilization is lower than the predefined threshold. Wang et al. [20] consider user association in a heterogeneous network with hybrid energy supplies, where energy saving is subject to the constraints of a user data rate requirement, transmission power budget, and so forth. They formulate the energy cost saving optimization problem and present both centralized and distributed solutions.

Chen et al. [21] propose an AP energy-saving mechanism using SDN, aiming to reduce the amount of idle APs while satisfying QoS requirements. In the mechanism, the user association problem is formulated as an ILP model. Lee et al. [22] present a centralized management mechanism to improve energy efficiency and avoid interference without sacrificing users demands, and jointly optimize the AP on/off scheduling, channel assignment, and user association.

As discussed above, several energy-efficient user association solutions have been proposed in the past. However, to the best of our knowledge, no prior work exists that deals with the issues of AP congestion and user migration in the context of energy-saving optimization in WLANs. Hence, here we study the user association problem in green WLANs, which aims to achieve energy efficiency with AP congestion avoidance and user migration constraints.

3. System Model and Problem Formulation

3.1. System Model. Our IEEE 802.11-based multirate WLAN consists of multiple APs. The set of APs is denoted by A, and we use m to denote their number, i.e., $A = \{a_1, a_2, \ldots, a_m\}$. Because users are on the go and their traffic demands are dynamic in WLANs, we represent the distribution of user traffic demands by discrete points (that is DNs) that are the center of an area aggregating the traffic demands of several users [23]. Thus, AP-user associations can be represented by AP-DN associations. We use U to denote the set of DNs, and the number of DNs is represented by n—that is, $U = \{u_1, u_2, \ldots, u_n\}$. As shown in Figure 2, the AP-DN associations can be modeled as a directed graph $G(A \cup U, E)$, where E is the set of potential association relations between DNs and APs.

Because multimedia content and downloads of mobile applications dominate the traffic demand, downlink traffic is much larger than uplink traffic, and thus we focus on downlink traffic (from the APs to DNs). For a DN $u_j \in U$, we assume that its traffic demand is a constant for a certain time interval T, and the constant is denoted by r_j.

We assume that neighboring APs do not interfere with each other through allocating nonoverlapping channels. For the transmission between $a_i \in A$ and $u_j \in U$, its available transmission rate is denoted by c_{ij}. We exploit the PHY multirate capability and enable each device to select the best transmission rate according to the received signal noise ratio (SNR), which we obtain by

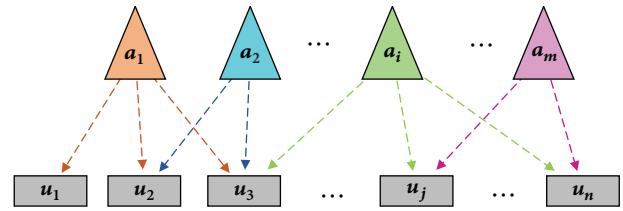


Figure 2: AP-DN association model.

$$SNR_{ij} = p_i - PL\left(d_{ij}\right) - N \tag{1}$$

where $SNR_{i,j}$ represents the SNR of signals that are transmitted from AP a_i and received at DN u_j; p_i is the transmit power of AP a_i; d_{ij} is the distance between AP a_i and DN u_j, and $PL(d_{ij})$ denotes the propagation path loss; N is the background noise power.

To characterize energy consumption of APs correctly, we use a fine-grained power model where we ascribe the power consumed by AP a_i to the following two elements [24, 25]:

(i) The baseline power (denoted by b_i) is a constant, which quantifies power consumption when the AP neither sends nor receives traffic after it is powered on.

(ii) The traffic-related power, a variable part, is generated by the wireless interface and relevant components. Because we only consider downlink traffic in this paper, the traffic-related power pertains to AP transmitting wireless signals, and thus it positively correlates with AP transmit power and AP utilization. AP utilization is defined by the fraction of time during which the AP transmits traffic.

For $a_i \in A$, we define AP power as follows:

$$P_i = b_i + \eta_i p_i \sum_{j \in U(a_i)} \frac{r_j}{c_{ij}} \tag{2}$$

where the power of AP a_i is denoted by P_i, $U(a_i)$ represents the set of DNs that are associated with AP a_i, and η_i is an efficiency factor that accounts for the AP's electrical model.

The notations and their definitions are summarized in Table 1.

3.2. Problem Formulation. In this section, we formulate the energy-efficient user association as an optimization problem based on an ILP model. Specifically, we not only consider energy optimization in the context of user association, but also take AP congestion avoidance and user migration constraints into account. We therefore define the problem as follows.

Definition 1 (EACM). Given the network $G(A \cup U, E)$, the current traffic demands of DNs (i.e.,$\{r_j \mid u_j \in U\}$), and the

Table 1: Notations.

Symbol	Definition
A	The set of access points (APs)
U	The set of demand nodes (DNs)
E	The set of potential association relations
m	The number of APs
n	The number of DNs
a_i	An AP
u_j	Am DN
r_j	The traffic demand of DN u_j
T	Time interval
c_{ij}	The available transmission rate between AP a_i and DN u_j
SNR_{ij}	The received SNR at DN u_j when signal is from AP a_j
P_i	The power of AP a_i
b_i	The baseline power of AP a_i
p_i	The transmit power of AP a_i
η_i	The efficiency factor of AP a_i
d_{ij}	The distance between AP a_i and DN u_j
$PL(d_{ij})$	The propagation path loss
N	The background noise power
$U(a_i)$	The set of DNs that are associated with AP a_i
β_{ij}	The coverage indicator between AP a_i and DN u_j
φ	The threshold of AP utilization
$\alpha(j)$	The previous association indicator
k	The maximum allowed number of migrations

previous AP-DN associations, the problem is to minimize the overall energy consumption of the APs in the current time interval through optimizing the AP-DN associations; while the traffic demands of the DNs are satisfied, the utilization of each AP is limited by a threshold, and DN migrations are also constrained.

To formulate the ILP model, we define the following sets of binary variables:

(i) x_i, which is set to 1 if AP a_i is turned on, or 0 otherwise.

(ii) $y_{i,j}$, which is set to 1 if DN u_j is associated with AP a_i, or 0 otherwise.

For each $a_i \in A$ and $u_j \in U$, we also define a coverage indicator to denote whether DN u_j is in the coverage range of AP a_i; that is,

$$\beta_{i,j} = \begin{cases} 1 & c_{i,j} > 0 \\ 0 & c_{i,j} = 0. \end{cases} \tag{3}$$

In addition, let $\alpha(j)$ be the previous association indicator that represents the AP-DN association in the previous time interval for DN u_j.

The objective is to minimize the overall energy consumption of the APs in the current time interval, as described by

$$\min T \sum_{i \in A} \left(b_i x_i + \eta_i p_i \sum_{j \in U} \frac{r_j y_{ij}}{c_{ij}} \right). \tag{4}$$

The minimization is subject to the following constraints:

$$y_{ij} \le \beta_{ij} x_i \quad \forall i \in A,\ j \in U \tag{5}$$

$$\sum_{i \in A} y_{ij} = 1 \quad \forall j \in U \tag{6}$$

$$\sum_{j \in U_i} \frac{r_j y_{ij}}{c_{ij}} \le \varphi x_i \quad \forall i \in A \tag{7}$$

$$\sum_{j \in U} y_{\alpha(j) j} \ge n - k \tag{8}$$

$$x_i \in \{0, 1\} \quad \forall i \in A \tag{9}$$

$$y_{ij} \in \{0, 1\} \quad \forall i \in A,\ j \in U. \tag{10}$$

In the aforementioned formulation, the constraint (5) states that no DN is associated with powered-off APs and an AP does not provide services for DNs beyond its coverage range. The constraint (6) imposes that each DN must be associated with an AP only. The constraint (7) ensures that each AP can satisfy the traffic demands of its associated DNs, and its AP utilization is limited by the coefficient φ. The constraint (8) denotes the allowed maximum number of migrations that cannot be exceeded. Finally, the constraints (9) and (10) define the binary of the variables.

The EACM problem is NP-hard, because it includes as a special case the set-covering problem, known to be NP-hard [26]. Although the schemes that solve the ILP model directly can find the optimal solution or at least bound it, they are impractical for a relatively large-scale scenario, because of computational complexity and memory limitations [27]. Thus, the better alternative here is to design a heuristic algorithm to solve the EACM problem efficiently.

4. MURA

In this section, we propose MURA, our efficient two-step algorithm to solve the EACM problem. The basic idea is to migrate at most k DNs from their original APs to neighboring APs, while considering energy saving and congestion avoidance. We solve this in two steps. In Step 1, we select k DNs that are associated with heavily loaded or idle APs and remove them from the current associations. In Step 2, we reassociate the removed DNs according to the energy efficiency strategy.

4.1. Step 1: DN Removal. In this step, we need to determine which k DNs to remove from the current associations. As described in Algorithm 1, we first initialize the set of APs that are turned on (A_on), the utilization of each AP (P_Util), the number of associated DNs for each AP (AP_UNum), the set of current AP-DN associations (UA_fxd), the set

Input: $A, U, E, UA\_pre\ k, \varphi, \{r_j \mid u_j \in U\}$, $\{c_{ij} \mid a_i \in A, u_j \in U\}$
Output: UA_fxd, U_rmv
(1) Initialize A_on // Set of APs that are turned on
(2) Initialize AP_Util // AP utilization
(3) Initialize AP_UNum // Number of associated DNs for each AP
(4) $UA\_fxd \leftarrow UA\_pre$ // Set of fixed AP-DN associations
(5) $U\_rmv \leftarrow \emptyset$
(6) $rm\_cnt \leftarrow k$ // Limitation of DN removal
(7) **while** *true* **do**
(8) $a_q \leftarrow \arg\max_{a_i \in A} AP\_Util(a_i)$ // Find the AP with the maximum utilization
(9) **if** $AP\_Util(a_q) > \varphi$ **then**
(10) **if** $rm\_cnt = 0$ **then**
(11) **return** *null*, *null* // Failure of AP congestion avoidance
(12) **else**
(13) $U\_a \leftarrow \{u \mid (a_q, u_j) \in UA\_fxd, \forall u_j \in U\}$
(14) Select a proper DN u_p from U_a
(15) Remove (a_q, u_p) from UA_fxd
(16) $U\_rmv \leftarrow U\_rmv \cup \{u_p\}$
(17) Update $AP\_Util(a_q)$, $AP\_UNum(a_q)$
(18) $rm\_cnt = rm\_cnt - 1$
(19) **else**
(20) **if** $rm\_cnt = 0$ **then return** UA_fxd, U_rmv
(21)
(22) **else** break
(23)
(24) **while** *true* **do**
(25) $a\_q \leftarrow \arg\min_{a_i \in A\_on} AP\_UNum(a_i)$
(26) $U\_a \leftarrow \{u_j \mid (a_q, u_j) \in UA\_fxd, \forall u_j \in U\}$
(27) **foreach** $u_j \in U\_a$ **do**
(28) Remove (a_q, u_j) from UA_fxd
(29) $rm\_cnt = rm\_cnt - 1$
(30) **if** $rm\_cnt = 0$ **then return** UA_fxd, U_rmv
(31)
(32) $A\_on = A\_on \setminus \{a_q\}$ // Turn off as many APs as possible

Algorithm 1: DN removal.

of removed DNs (U_rmv), and the required number of DN removals (rm_cnt). Then, we remove DNs from the current associations, successively considering AP congestion avoidance and energy saving. Specifically, we divide the process of DN removal into two parts. In the first part (lines (7) to (23)), we iteratively remove DNs to address the issue of AP congestion. In each iteration, we find the AP with the maximum utilization and select a DN migrating from it if its utilization exceeds the threshold φ. The DN selection strategy is to make AP utilization lower than the threshold or nearer to the threshold. In the second part (lines (24) to (32)), we remove DNs so as to turn off as many APs as possible for energy saving. In each iteration, we find the AP with the minimum number of associated DNs, remove these DNs, and

Input: $A, U, E, U\_rmv, UA\_fxd, \varphi, \{r_j \mid u_j \in U\}$, $\{c_{ij} \mid a_i \in A, u_j \in U\}$, $\{b_i, p_i, \eta_i \mid a_i \in A\}$
Output: UA
(1) Initialize A_on // Set of APs that are turned on
(2) Initialize AP_Util // AP utilization
(3) $UA \leftarrow UA\_fxd$ // Set of AP-DN associations
(4) Sort U_rmv in descending order by traffic demand
(5) **while** $U\_rmv \neq \emptyset$ **do**
(6) obtain DN u_p from U_rmv in sequence
(7) $a\_weight(i) \leftarrow \infty\ \forall a_i \in A$
(8) **if** $AP\_Util(a_i) + r_p/c_{ip} \leq \varphi$ **then**
(9) **if** $a_i \in A\_on$ **then**
(10) $a\_weight(i) \leftarrow \eta_i p_i r_p / c_{ip}$ // increment of energy consumption
(11) **else**
(12) $a\_weight(i) \leftarrow b_i + \eta_i p_i r_p / c_{ip}$
(13) $a_q \leftarrow \arg\min_{a_i \in A} a\_weight(i)$
(14) **if** $a\_weight(q) \neq \infty$ **then**
(15) **if** $a_q \notin A\_on$ **then**
(16) $A\_on \leftarrow A\_on \cup \{a_q\}$
(17) $UA \leftarrow UA \cup \{(a_q, u_p)\}$
(18) $U\_rmv \leftarrow U\_rmv \setminus \{u_p\}$
(19) Update $AP\_Util(a_q)$
(20) **else**
(21) **return** *null* // Reassociation failed
(22) **return** UA

Algorithm 2: DN reassociation.

turn off the APs. The step stops when the total number of removed DNs reaches k and returns UA_fxd and U_rmv.

4.2. Step 2: DN Reassociation. As DNs in set U_rmv are removed from the current associations, we need to reassociate them in this step. As described in Algorithm 2, first we initialize the set of APs turned on (A_on), the utilization of each AP (AP_Util), and the set of current AP-DN associations (UA). Then, we determine the new reassociations of these DNs in sequence and prioritize the DN with greater traffic demand. For each DN, we choose an AP with the lowest increment of network energy consumption and assign the new association. The increment of network energy consumption is the additional energy consumption caused by the AP-DN association. If the AP is not active, it should be turned on first, and its baseline power should be involved in the increment of network energy consumption. After each reassociation, the set U_rmv and AP_Util are updated.

4.3. Time Complexity. In the following, we analyze the time complexity of the MURA algorithm. This includes the two steps for DN removal and DN reassociation.

In the first step, the algorithm is initialized in $O(mn)$, and DN removal iterates at most k times. For each iteration (lines (8) to (23) or lines (25) to (32)), the time complexity is $O(m+$

$n + |U\_a|$). Because $m < n$ and $|U\_a| < n$, the time complexity of this step is $O((m + k)n)$.

In the second step, again the initialization completes within $O(mn)$, and the sorting of U_rmv can be accomplished in $O(k \lg k)$. DN reassociation iterates at most k times. For each iteration (lines (5) to (20)), the time complexity is $O(m)$. Because $k < n$, the time complexity of this step is $O(mn + k \lg k)$.

Therefore, as discussed, the time complexity of the MURA algorithm is $O((m + k)n)$.

5. Simulation Evaluation

In this section, we evaluate the performance of the proposed MURA algorithm via simulations and compare its solutions with the solutions of the ILP-based association scheme and the solutions of the received signal strength indication- (RSSI-) based association scheme. First, we validate the MURA algorithm's effectiveness in terms of energy consumption and execution time. Then, we evaluate how the performance of the MURA algorithm is affected by the AP utilization threshold, number of allowed migrations, AP types, and traffic patterns.

5.1. Performance Metrics. We are interested primarily in energy consumption and execution time as performance metrics. Here, *energy consumption* is the overall energy consumption of all APs in a time interval. By *execution time*, we mean the algorithm's execution time within a time interval. In addition, we compare MURA to the following schemes:

(i) ILP-based association scheme: in this scheme, our EACM model is solved by an ILP solver, such as Gurobi.

(ii) RSSI-based association scheme: in this scheme, each DN is associated with the AP that offers the best signal strength.

5.2. Simulation Setup and Parameters. In the simulation, we consider small-scale scenarios, medium-scale scenarios, and large-scale scenarios, respectively. As shown in Table 2, the network field is divided by grids, and APs are deployed at the center of grids to provide full coverage to the field. In addition, the same number of DNs are randomly placed in each grid.

To characterize the indoor environment, we define the path loss calculation as follows [28]:

$$PL\left(d_{ij}\right) = 40 + 10 \times 3.3 \lg d_{ij}. \quad (11)$$

After the transmit power of AP is set to be 20 dBm (i.e., 0.1 W) and the background noise level is set to −93 dBm, we can compute the SNR of DNs. In this paper, we consider an IEEE 802.11n WLAN with 40 MHz channel, and we list the corresponding link rates in Table 3 [29, 30].

We take the traffic pattern shown in Table 4, where the time duration is three hours, and assign a probability $prob_t$ of DNs requesting demands in each interval t [31]. Moreover, we use two different traffic demand profiles:

(i) Standard mode, where each DN's traffic demands randomly are generated in intervals of 1 to 10 Mbps.

(ii) Busy mode, where each DN's traffic demand varies between 8 and 10 Mbps.

For each setting, we execute these schemes 20 times with different scenarios and use the average results as the final results. We conduct the simulation experiments on a PC equipped with an Intel Core i7 3.40 GHz processor, 8 GB RAM, Microsoft Windows 7, and MATLAB environment, where the proposed ILP model is solved by Gurobi.

5.3. Verification of Energy Efficiency. We compare the solutions obtained from the MURA algorithm with those obtained from the ILP-based association scheme and the RSSI-based association scheme. For the AP power model, we set the baseline power (b_i) to be 9 W and the efficiency factor (η_i) to be 30. For the traffic pattern, we adopt the standard mode. We set the AP utilization threshold (φ) to be 0.8 and the maximum allowed number of migrations (k) to be 30 percent of the DN amount. We perform the simulations for small-scale, medium-scale, and large-scale scenarios, respectively.

Table 2: Network scenarios.

Network size	Number of APs	Number of DNs
Small ($100 \times 100m^2$)	4	20
Medium ($250 \times 250m^2$)	25	125
Large ($1000 \times 1000m^2$)	400	2000

Table 3: SNR and link rates.

SNR range(dB)	Link rate (Mbps)
[5, 8)	15
[8, 12)	30
[12, 14)	45
[14, 18)	60
[18, 21)	90
[21, 23)	120
[23, 28)	135
[28, ∞)	150

Table 4: Traffic pattern.

Index t	Starting	Ending	Duration(h)	$prob_t$
1	0	3	3	0.35
2	3	6	3	0.1
3	6	9	3	0.45
4	9	12	3	1
5	12	15	3	0.7
6	15	18	3	0.85
7	18	21	3	0.6
8	21	24	3	0.5

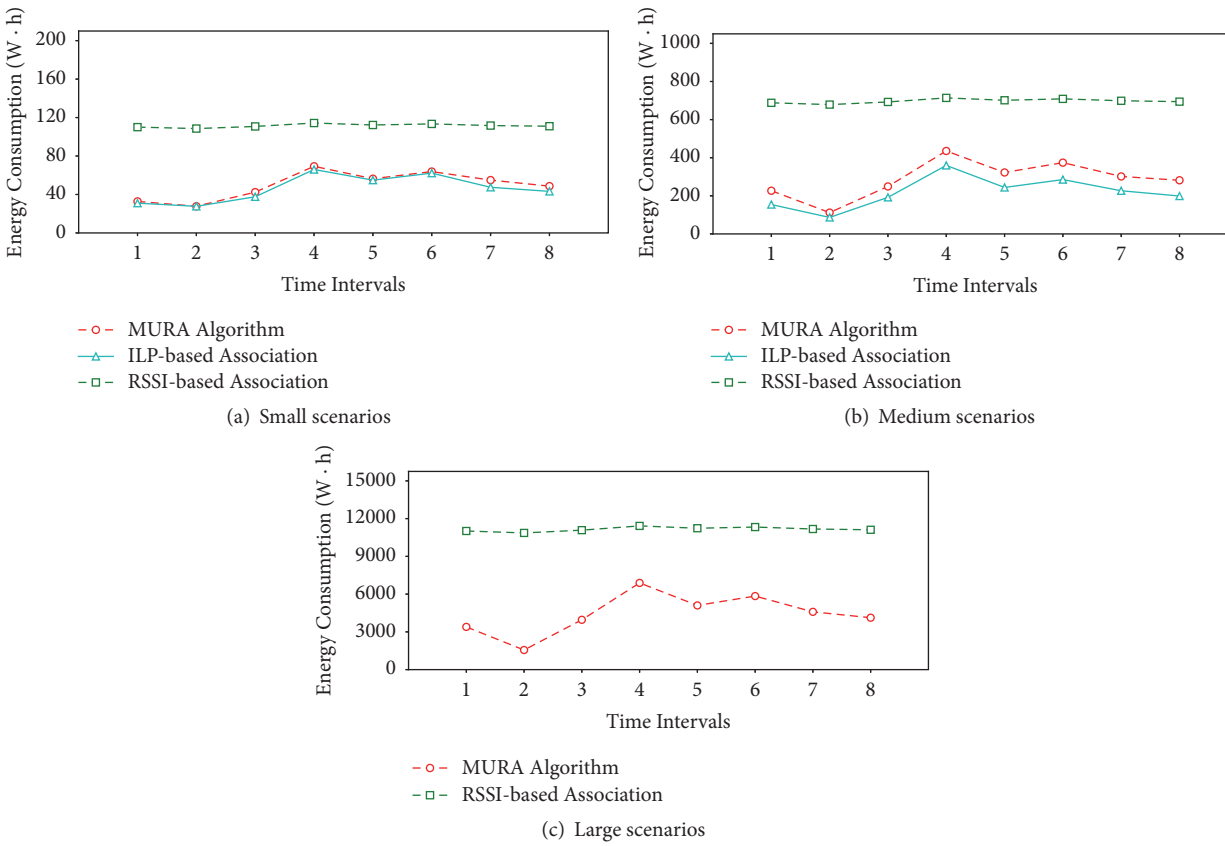


Figure 3: Validation of energy efficiency.

Table 5: Execution Time.

Time intervals	Execution time (Sec.)								
	Small scenarios			Medium scenarios			Large scenarios		
	MURA	ILP-based	RSSI-based	MURA	ILP-based	RSSI-based	MURA	ILP-based	RSSI-based
1	0.0013	0.0086	0.0005	0.0057	0.4928	0.0017	0.2718	-	0.0176
2	0.0004	0.0068	0.0002	0.0015	0.0235	0.0009	0.0524	-	0.0127
3	0.0008	0.0099	0.0003	0.0051	0.7347	0.0013	0.3151	-	0.0193
4	0.0008	0.0124	0.0003	0.0052	0.5227	0.0016	0.4825	-	0.0306
5	0.0007	0.0120	0.0004	0.0042	3.8941	0.0014	0.2137	-	0.0239
6	0.0008	0.0137	0.0003	0.0051	6.8822	0.0015	0.4613	-	0.0276
7	0.0007	0.0128	0.0003	0.0043	2.6668	0.0013	0.2814	-	0.0219
8	0.0008	0.0122	0.0003	0.0045	1.0751	0.0012	0.3111	-	0.0202

Figure 3 shows the performance of the MURA algorithm, the ILP-based association scheme, and the RSSI-based association scheme. From this figure, we can see that the MURA algorithm achieves significant energy savings comparing to the RSSI-based association scheme. Also, there is only a small gap in performance between our solution and the optimal solutions obtained by the ILP-based association scheme, which validates the near optimality of our algorithm. When the network scenarios become large, the ILP model cannot be solved by Gurobi directly, but our algorithm provides solutions within an acceptable time, as shown in Table 5. Moreover, the curves of our algorithm in Figure 3 show that an interval's energy consumption is proportional to the traffic demand, and energy consumption greatly diminishes when traffic demand is low.

5.4. Varying the Maximum Allowed Number of Migrations. Here, we evaluate the performance of the MURA algorithm, while varying the maximum allowed number of migrations (the parameter k). We perform simulations on large-scale

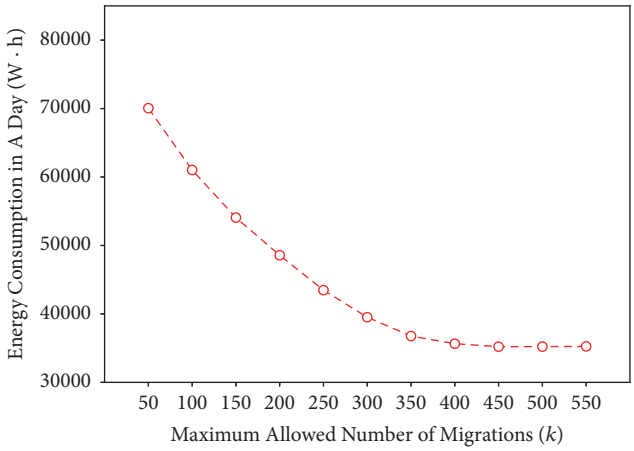


Figure 4: Energy consumption varying parameter k.

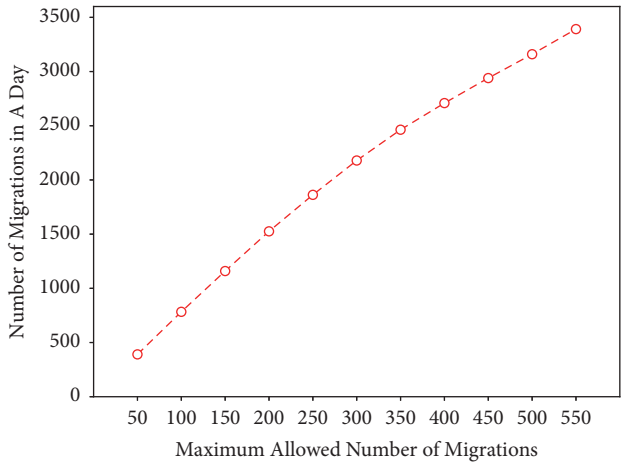


Figure 5: DN migrations varying parameter k.

scenarios and set the AP utilization threshold (φ) to be 0.8. For the AP power model, we again set the baseline power (b_i) to be 9 W and the efficiency factor (η_i) to be 30. For the traffic pattern, we adopt the standard mode. We set the maximum allowed number of migrations (k) to be 50, 100, ..., and 550.

Figures 4 and 5 show the results. From the two figures, we can see that energy consumption decreases as the maximum allowed number of migrations relaxes, because we can further optimize energy consumption through more DN migrations. We also find that the decrement of energy consumption is extremely limited once the parameter k has reached 350. This substantiates why we set 350 as the maximum number of migrations allowed.

5.5. Varying the AP Utilization Threshold. Next, we evaluate the MURA algorithm's performance while varying the AP utilization threshold (the parameter φ). In performing simulations on large-scale scenarios, we set the maximum allowed number of migrations (k) to be 350. For the AP power model, we set the baseline power (b_i) to be 9 W and the efficiency factor (η_i) to be 30. For the traffic pattern, we adopt the standard mode. We set the AP utilization threshold to be 0.7, 0.75, ..., and 1.0.

Figures 6 and 7 show the results. We can see that energy consumption decreases as the AP utilization threshold relaxes, because each AP is able to serve more DN migrations and thus we can turn off more APs. Hence, φ is an adjustable parameter that can provide a tradeoff between energy efficiency and congestion avoidance.

Table 6: Parameters of the AP power model.

Type ID	Baseline power (b_i)	Efficiency factor (η_i)
1	9	30
2	7	50
3	6	60
4	3	90

Table 7: Energy consumption affected by traffic pattern.

Traffic pattern	Energy consumption in a day (W · h)	
	MURA algorithm	RSSI-based association
Standard	36752	89237
Busy	48934	91501

5.6. Impacts of AP Types. Now, we evaluate the MURA algorithm's performance while varying AP types. We perform simulations on large-scale scenarios, setting the AP utilization threshold to be 0.8 and the maximum allowed number of migrations (k) to be 350. For the traffic pattern, we adopt the standard mode. Table 6 shows parameter settings of the AP power model [25].

As shown in Figure 8, no matter which APs we use, our algorithm's solutions perform better on energy consumption than the RSSI-based association scheme does. In addition, type 1 APs show greater energy savings, because any baseline power that can be eliminated by turning off APs plays a decisive role in AP power.

5.7. Impacts of Traffic Pattern. Here, we evaluate the performance of the MURA algorithm under different traffic patterns, such as standard mode and busy mode. We perform simulations on large-scale scenarios, set the AP utilization threshold to be 0.8, and set the maximum allowed number of migrations (k) to be 350. For the AP power model, we set the baseline power (b_i) to be 9 W and the efficiency factor (η_i) to be 30.

As shown in Table 7, no matter which traffic pattern we adopt, the solutions of our algorithm perform better on energy consumption than that of the RSSI-based association scheme. In addition, we find that when a traffic pattern switches from the busy mode to the standard mode, the decrement of our algorithm's energy consumption is more than the RSSI-based association scheme's, because some APs will turn off while executing our algorithm.

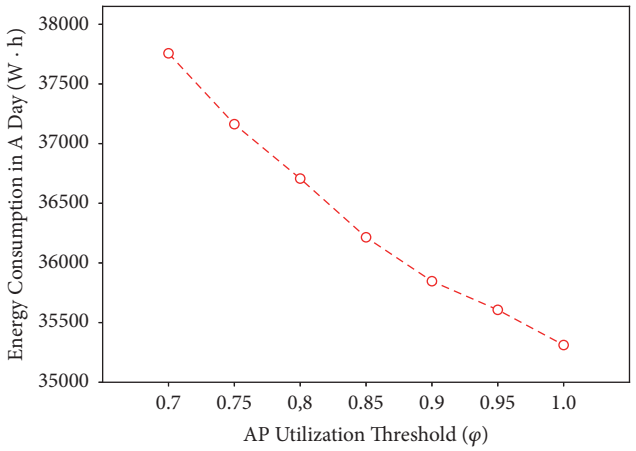


Figure 6: Energy consumption varying parameter φ.

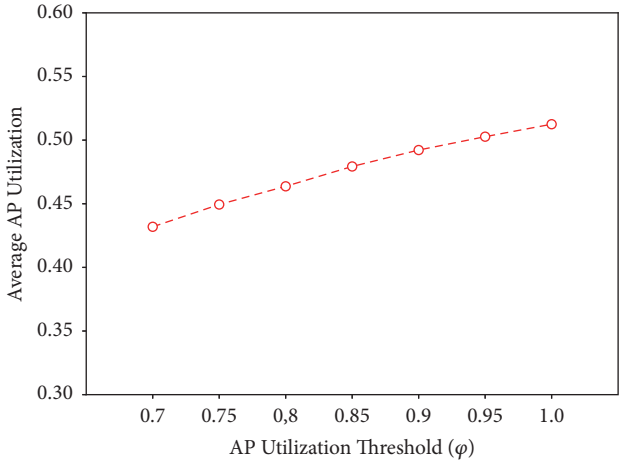


Figure 7: Average AP utilization varying parameter φ.

6. Conclusion

We investigated the energy-efficient user association in green WLANs while considering AP congestion avoidance and user migration constraints. First, we formulated the EACM problem as an ILP model, to minimize APs' overall energy consumption in a current time interval under the constraints of traffic demand, the AP utilization threshold, and the maximum number of migrations allowed. Then, we proposed MURA, a two-step algorithm that efficiently solves the problem. Finally, we conducted simulation experiments to evaluate the performance of our proposed algorithm. The results demonstrate that MURA effectively saves energy—beyond that, the algorithm can obtain the tradeoff between energy efficiency, congestion avoidance, and migration cost.

In the future, we plan to measure traffic patterns in real-world scenarios through crowdsourcing and propose a data-driven user association algorithm that applies well to a real network.

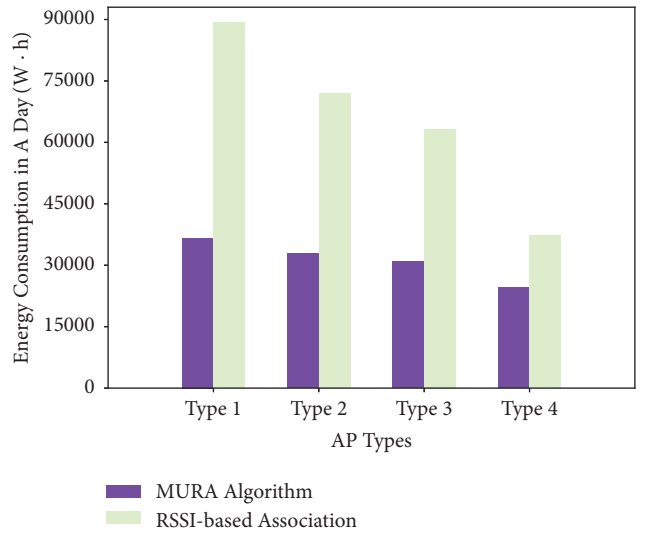


Figure 8: Energy consumption varying AP types.

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 work was supported by National Key R&D Program of China 2017YFB1003000, National Natural Science Foundation of China under Grants no. 61632008, no. 61572130, no. 61502100, no. 61602111, no. 61532013, and no. 61320106007, Jiangsu Provincial Natural Science Foundation of China under Grants no. BK20150637 and no. BK20150628, Jiangsu Provincial Key Laboratory of Network and Information Security under Grant no. BM2003201, and Key Laboratory of Computer Network and Information Integration of Ministry of Education of China under Grant no. 93K-9.

References

[1] 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.

[2] B. Bellalta, L. Bononi, R. Bruno, and A. Kassler, "Next generation IEEE 802.11 Wireless Local Area Networks: Current status, future directions and open challenges," *Computer Communications*, vol. 75, pp. 1–25, 2016.

[3] M. Afanasyev, T. Chen, G. M. Voelker, and A. C. Snoeren, "Usage patterns in an urban WiFi network," *IEEE/ACM Transactions on Networking*, vol. 18, no. 5, pp. 1359–1372, 2010.

[4] Ł. Budzisz, F. Ganji, G. Rizzo et al., "Dynamic resource provisioning for energy efficiency in wireless access networks:

A survey and an outlook," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 4, pp. 2259–2285, 2014.

[5] X. Xu, W. Dou, X. Zhang, and J. Chen, "EnReal: An Energy-Aware Resource Allocation Method for Scientific Workflow Executions in Cloud Environment," *IEEE Transactions on Cloud Computing*, vol. 4, no. 2, pp. 166–179, 2016.

[6] 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.

[7] A. P. Jardosh, K. Papagiannaki, E. M. Belding, K. C. Almeroth, G. Iannaccone, and B. Vinnakota, "Green WLANs: on-demand WLAN infrastructures," *Mobile Networks and Applications*, vol. 14, no. 6, pp. 798–814, 2009.

[8] P. Serrano, A. De La Oliva, P. Patras, V. Mancuso, and A. Banchs, "Greening wireless communications: Status and future directions," *Computer Communications*, vol. 35, no. 14, pp. 1651–1661, 2012.

[9] F. Ganji, Ł. Budzisz, F. G. Debele et al., "Greening campus WLANs: Energy-relevant usage and mobility patterns," *Computer Networks*, vol. 78, pp. 164–181, 2015.

[10] V. Sivaraman, J. Matthews, C. Russell, S. T. Ali, and A. Vishwanath, "Greening residential Wi-Fi networks under centralized control," *IEEE Transactions on Mobile Computing*, vol. 14, no. 3, pp. 552–564, 2015.

[11] Y. Bejerano, S. Han, and L. E. Li, "Fairness and load balancing in wireless LANs using association control," in *Proceedings of the 10th Annual International Conference on Mobile Computing and Networking*, pp. 315–329, ACM, Philadelphia, PA, USA, September 2004.

[12] W. Li, S. Wang, Y. Cui et al., "AP association for proportional fairness in multirate WLANs," *IEEE/ACM Transactions on Networking*, vol. 22, no. 1, pp. 191–202, 2014.

[13] D. Gong and Y. Yang, "On-line AP association algorithms for 802.11n WLANs with heterogeneous clients," *Institute of Electrical and Electronics Engineers. Transactions on Computers*, vol. 63, no. 11, pp. 2772–2786, 2014.

[14] J. Yu and W.-C. Wong, "Optimal association in wireless mesh networks," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 5, pp. 2084–2096, 2015.

[15] W. Wong, A. Thakur, and S. G. Chan, "An approximation algorithm for AP association under user migration cost constraint," in *Proceedings of the IEEE INFOCOM 2016 - IEEE Conference on Computer Communications*, pp. 1–9, San Francisco, CA, USA, April 2016.

[16] A. Raschella, F. Bouhafs, M. Seyedebrahimi, M. Mackay, and Q. Shi, "Quality of service oriented access point selection framework for large Wi-Fi networks," *IEEE Transactions on Network and Service Management*, vol. 14, no. 2, pp. 441–455, 2017.

[17] J. Chen, B. Liu, H. Zhou, Q. Yu, L. Gui, and X. S. Shen, "QoS-Driven Efficient Client Association in High-Density Software-Defined WLAN," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 8, pp. 7372–7383, 2017.

[18] H. Kim, W. Lee, M. Bae, and H. Kim, "Wi-Fi Seeker: A link and Load Aware AP Selection Algorithm," *IEEE Transactions on Mobile Computing*, vol. 16, no. 8, pp. 2366–2378, 2017.

[19] K. Kumazoe, D. Nobayashi, Y. Fukuda, T. Ikenaga, and K. Abe, "Multiple station aggregation procedure for radio-on-demand WLANs," in *Proceedings of the 2012 7th International Conference on Broadband, Wireless Computing, Communication and Applications, BWCCA 2012*, pp. 156–161, Victoria, Canada, November 2012.

[20] B. Wang, Q. Kong, and L. T. Yang, "Context-aware user association for energy cost saving in a green heterogeneous network with hybrid energy supplies," *Mobile Networks and Applications*, vol. 20, no. 6, pp. 802–816, 2015.

[21] Y. Chen, Y. Shen, and L. Wang, "Achieving energy saving with QoS guarantee for WLAN using SDN," in *Proceedings of the ICC 2016 - 2016 IEEE International Conference on Communications*, pp. 1–7, Kuala Lumpur, Malaysia, May 2016.

[22] K. Lee, Y. Kim, S. Kim, J. Shin, S. Shin, and S. Chong, "Just-in-time WLANs: On-demand interference-managed WLAN infrastructures," in *Proceedings of the IEEE INFOCOM 2016 - IEEE Conference on Computer Communications*, pp. 1–9, San Francisco, CA, USA, April 2016.

[23] K. Tutschku, "Demand-based radio network planning of cellular mobile communication systems," in *Proceedings of the IEEE INFOCOM'98 Conference on Computer Communications Seventeenth Annual Joint Conference of the IEEE Computer and Communications Societies Gateway to the 21st Century*, pp. 1054–1061, San Francisco, CA, USA, 1998.

[24] A. Garcia-Saavedra, P. Serrano, A. Banchs, and G. Bianchi, "Energy consumption anatomy of 802.11 devices and its implication on modeling and design," in *Proceedings of the 8th ACM International Conference on Emerging Networking EXperiments and Technologies, CoNEXT 2012*, pp. 169–180, fra, December 2012.

[25] R. G. Garroppo, G. Nencioni, G. Procissi, and L. Tavanti, "The impact of the access point power model on the energy-efficient management of infrastructured wireless LANs," *Computer Networks*, vol. 94, pp. 99–111, 2016.

[26] V. Chvatal, "A greedy heuristic for the set-covering problem," *Mathematics of Operations Research*, vol. 4, no. 3, pp. 233–235, 1979.

[27] M. Rim, A. Mujumdar, R. Jain, and R. DeLeone, "Optimal and Heuristic Algorithms for Solving the Binding Problem," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 2, no. 2, pp. 211–225, 1994.

[28] Y. Bejerano and S.-J. Han, "Cell breathing techniques for load balancing in wireless LANs," *IEEE Transactions on Mobile Computing*, vol. 8, no. 6, pp. 735–749, 2009.

[29] MCS index - 802.11n and 802.11ac,2015, https://www.wlanpros.com/resources/mcs-index-802-11ac-vht-chart/.

[30] MCS value achieved by clients at various signal to noise ratio levels (SNR), 2015, https://www.wlanpros.com/resources/revolution-wifi-mcs-to-snr-single-page/.

[31] A. Capone, F. Malandra, and B. Sansò, "Energy savings in Wireless Mesh Networks in a time-variable context," *Mobile Networks and Applications*, vol. 17, no. 2, pp. 298–311, 2012.

Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 8196906, 15 pages
https://doi.org/10.1155/2018/8196906

Research Article

Multiactivation Pooling Method in Convolutional Neural Networks for Image Recognition

Qi Zhao, Shuchang Lyu, Boxue Zhang, and Wenquan Feng

School of Electronics and Information Engineering, Beihang University, Beijing 100191, China

Correspondence should be addressed to Qi Zhao; zhaoqi@buaa.edu.cn

Received 21 February 2018; Accepted 8 May 2018; Published 26 June 2018

Academic Editor: Xuyun Zhang

Convolutional neural networks (CNNs) are becoming more and more popular today. CNNs now have become a popular feature extractor applying to image processing, big data processing, fog computing, etc. CNNs usually consist of several basic units like convolutional unit, pooling unit, activation unit, and so on. In CNNs, conventional pooling methods refer to 2×2 max-pooling and average-pooling, which are applied after the convolutional or ReLU layers. In this paper, we propose a Multiactivation Pooling (MAP) Method to make the CNNs more accurate on classification tasks without increasing depth and trainable parameters. We add more convolutional layers before one pooling layer and expand the pooling region to 4×4, 8×8, 16×16, and even larger. When doing large-scale subsampling, we pick top-k activation, sum up them, and constrain them by a hyperparameter σ. We pick VGG, ALL-CNN, and DenseNets as our baseline models and evaluate our proposed MAP method on benchmark datasets: CIFAR-10, CIFAR-100, SVHN, and ImageNet. The classification results are competitive.

1. Introduction

Convolutional neural networks (CNNs) have excellent performance on image classification and many other visual tasks [1–10] in recent years since AlexNet [11] achieved great success in ImageNet Challenge. With the rapid development of hardware capacity, wider and deeper networks can be trained smoothly. Nowadays, increasing the depth of networks is the main trend of improving networks' overall performance. The first proposed convolutional neural network, LeNet5 [12], has 5 layers. AlexNet contains 8 layers which consist of 5 convolutional layers and 3 fully connected layers. VGG [13] networks are designed even deeper. The deepest number of layers reaches 19, while GoogLeNet [14] achieves 22. Residual Networks (ResNets) [16, 17] and Dense Convolutional Networks (DenseNets) [15] which have been proposed in the last two years start to use shortcuts structure to allow the networks to surpass 100, even 1000-layer barrier.

When networks are deep, gradient vanishing is a serious problem during training process. ResNets [16, 17] and DenseNets [15] explicitly take advantage of the shortcuts structure between each two blocks. Highway Networks [18] integrate nonlinear transform and transform gates to create shortcuts implicitly. FractalNets [19, 20] apply drop paths, which can achieve the similar effect as ResNets. All of the methods can effectively solve the gradient vanishing.

Research [21] shows that classification accuracy will degrade rapidly when applying backpropagation [22] to deeper plain networks. Normalization method [23], to some degree, can help ease the problem. Despite lacking the potential to become deeper, plain networks still have many advantages, especially when dealing with some embedded vision tasks. The ALL Convolution Net (ALL-CNN) [24] uses only convolutional layers with small amount of parameters instead of alternating convolutional and max-pooling layers, which has outstanding performance. Based on plain network structure, many small networks perform better on limited resource. Similarity Networks (SimNets) [25] use the similarity operator and global MEX pooling method to improve the capacity of small-size networks as much as possible. SqueezeNet [26] reduces the channels of 3×3 filters and partially substitutes them by 1×1 in order to simplify the networks without hurting network capability. MobileNets [27] use depth-wise separable convolutions to build lightweight networks and work well on embedded devices. Moreover, some notable researches such as ShuffleNet [28] and SEP-Nets [29] have already

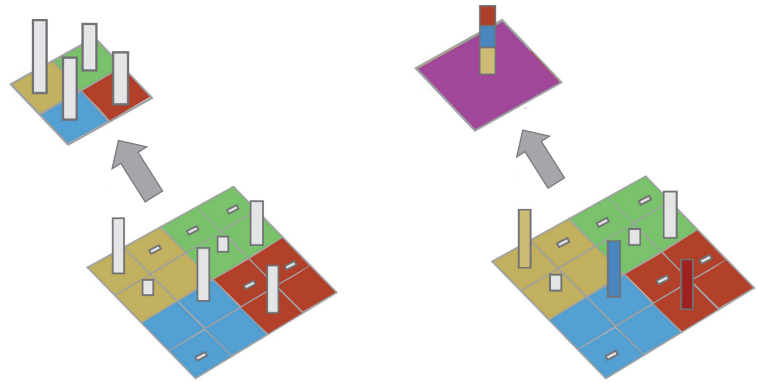


Figure 1: Comparison between 2×2 max-pooling (left) [36] and 4×4 top-3 Multiactivation Pooling (right) where σ is set to 0.5.

applied residual structure on mobile device and can reduce computational complexity effectively. With concise structure and less parameters, many researches [10] apply different kinds of CNNs as part of their model to process the big data, which improves the whole performance.

In this paper, we propose a new pooling method that picks top-k activation in every pooling region (Figure 1) to make sure that the maximum information can pass through subsampling gates. To enable this method suitable for plain networks, we do some necessary adjustment on networks architecture. **First,** we use small 3×3 receptive fields (stride = 1) throughout the whole net and add more convolutional layers before certain pooling layer. Nonlinear transform unit: ReLU [30] is followed after each convolutional layer. As input feature maps pass through more convolutional and ReLU layers, output feature maps become sparse. Therefore, expanding the pooling region and picking the max activation surely have effect on reducing noise even if some information might be ignored. **Second,** max-pooling method shows its strong performance in sparse features representation, but more key features will certainly be dropped when pooling regions are larger. To further improve the representation capability, we pick top-k activation, sum them up, and constrain the summation by a constant σ, which ranges from zero to one. If σ is set equivalent to $1/k$, the output value of pooling layer is the average of the picked top-k activation. Generally, we make σ a little bigger than $1/k$ to prevent activation from weakening too much when there are only few features active.

In plain networks, such as VGG and ALL-CNN, our MAP method is competitive in achieving higher accuracy on classification tasks with depth and trainable parameters not increased. When reducing the number of layers, the cost of graphics memories or internal memories will decrease a lot during training process. Therefore, networks with MAP method have huge advantage against both traditional networks and deep networks with shortcuts structure. Figure 2 shows the plain networks used different pooling strategies. In Networks with shortcuts structure, like DenseNets, we proposed plain structure (Figure 3) with MAP method into the Transition layers to extract features. Although a little more trainable parameters are added, the classification results improve.

We evaluate MAP method on 4 notable benchmarks datasets (CIFAR-10, CIFAR-100 [31], SVHN, and ImageNet). We mainly choose VGG [13] as our baseline models and it turns out that our enhanced models outperform them. We also apply our method to ALL-CNN [24] and DenseNets [15] and also get better results.

2. Related Works

In visual task, most of the networks insert pooling layers between convolutional layers for features abstraction and dimension reduction. The idea of pooling originates in Hubel and Wiesel's seminars [32] on complex cells in the visual cortex. In 1982, Fukuhshima and Miyake used feature pooling in neocognitron [33]. Pooling method has been widely used since the concept of convolutional neural networks structure was first put forward. LeNet-5 [12] has two convolutional layers and a pooling layer closely follows each of them. At that time, max-pooling and average-pooling both performed well in LeNet-5. Although the development of CNN slowed down for a period, the researches of pooling method never stop. In many famous traditional algorithms like SIFT [34] and HOG [35], pooling method plays an important role.

Two most popular pooling methods, max-pooling and average-pooling, have already been deeply researched in theory [37]. When choosing pooling strategies, there is always a trade-off between preserving more information which may include noise and decreasing more noise as well as useful information. Through research, it is found that max-pooling is more suitable for sparse coding than average-pooling [38].

Recently, CNNs have achieved great success on various visual tasks. Pooling layers have already become a fixed part in almost every network. Moreover, many new pooling methods are proposed one after another, which inspire us a lot. Different from conventional deterministic pooling operations, Stochastic Pooling [39] method picks activation randomly from each pooling region according to multinomial distribution. This notable research highly improves the generalization of networks. When applying Dropout to max-pooling layers [40] on training datasets, a new pooling method called scaled max-pooling [41] is designed to be applied on testing datasets. This idea of adopting different pooling methods on training and testing datasets is novel

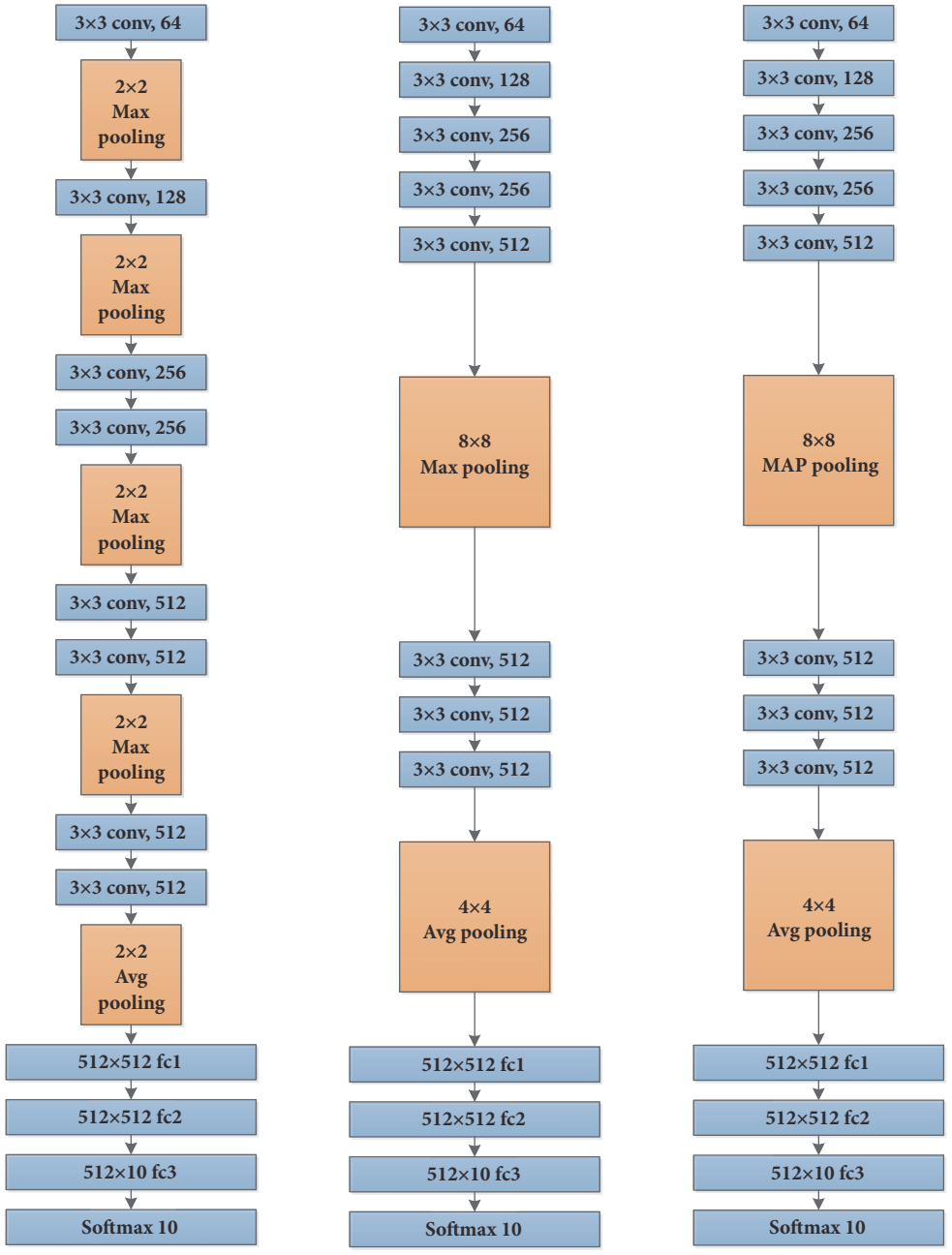


Figure 2: Plain networks for CIFAR-10. **Left:** applying VGG-11 [13] model, which is slightly different from the model designed for ImageNet. **Middle:** applying large-scale pooling (using 8×8 max-pooling to extract lower level features and 4×4 average-pooling for high-level features [36]) in VGG-11 model without increasing its depth and parameters. **Right:** applying MAP method in the network.

and effective. Combined with Dropout, probability weighted pooling [39] method also performs well. Another creative method, Spatial Pyramid Pooling [42] method, uses different size of pooling windows before the fully connected layers to solve the problem of feature loss caused by arbitrary input image size. This method has huge advantage on extracting different active features, which is similar to our idea. In addition, the idea of expanding pooling regions has already been implemented in some networks such as Value Iteration Network (VIN) [43]. VIN uses the large-scale channel-wise max-pooling method to improve the generalization of networks. The idea of mixing max-pooling and average-pooling method [44] has been proven effective. It is similar to our proposed method.

At the bottom of networks, almost every high-level feature in feature maps is useful. Global average-pooling (GAP) [45] method enables all of them to contribute to the final representation. Global max-pooling [46], which focuses on the most active feature in every feature map, is a universal method in object detection tasks. These two existing global pooling methods share a key characteristic: they both concentrate on large-scale pooling size.

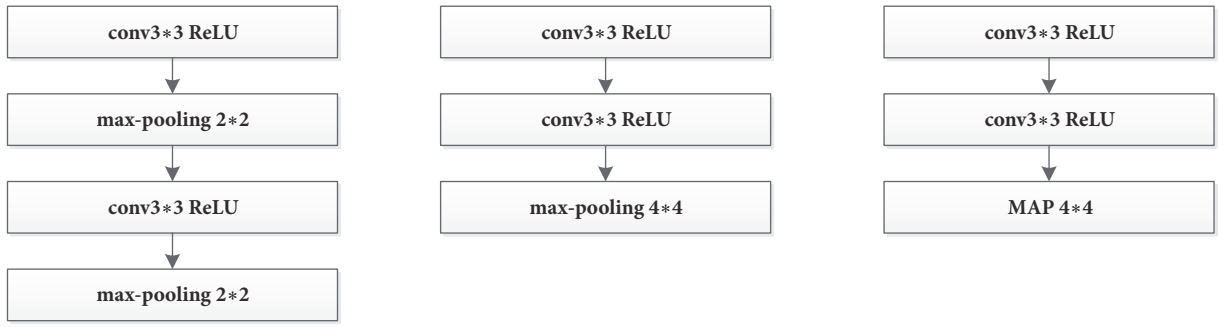


Figure 3: Plain structures used in Transition layers of DenseNets to better extract features. **Left:** using 2×2 max-pooling and downsample twice. **Middle:** using 4×4 max-pooling. **Right:** using 4×4 MAP method.

3. Multiactivation Pooling

3.1. Conventional Pooling Methods. Formally, we denote a single input feature map of a pooling layer as an assemblage $\mathbf{X}$. Before entering the pooling gate, $\mathbf{X}$ can be regarded as a combination of several small local regions [39], $\mathbf{X_0}, \ldots, \mathbf{X_n}$, where n is constrained by both the size of input feature maps and the size of pooling regions.

Consider an arbitrary local region $\mathbf{X_i}$ where $\boldsymbol{i}$ is an index between 0 and n. We denote the size of pooling region as M and each element as x. Here, we define

$$\mathbf{X_i} = (x_1, x_2, \ldots, x_{M\times M}) \tag{1}$$

The conventional pooling methods, max-pooling and average-pooling, make the use of different strategies dealing with the elements in each pooling region. We denote the output after pooling as act. Equation (2) describes max-pooling:

$$act_i = \max_{1\le j\le M\times M} \left(x_j\right) \tag{2}$$

Equation (3) describes average-pooling:

$$act_i = \frac{\mathbf{1}}{M\times M} \sum_{j=1}^{M\times M} x_j \tag{3}$$

Max-pooling method only picks the most active feature in a pooling region. On the contrary, average-pooling method takes all of the features into consideration. Thus, max-pooling method detects more texture information, while average-pooling method preserves more background information.

3.2. Multiactivation Pooling Method. In most cases, 2×2 pooling method without overlapping is frequently used. This means M in (1) is always set to 2. Large-scale pooling means M is set bigger such as 4, 8, and 16, which depends on the input image size. Figure 2 (**middle**) shows the 8×8 max-pooling and 4×4 average-pooling in VGG-11 architecture.

Based on the large pooling region, we propose Multi-activation Pooling method which allows top-k activation to pass through the pooling gate, where k indicates the total number of the picked activation. If $k = M\times M$, it means every activation will contribute to the final output; if $k = 1$, only the max activation can pass through the gate. To avoid mixing too much useless features, k is set small and constrained by the size of pooling region; e.g., k is set to 4, while pooling region is 8×8.

In an arbitrary pooling region $\mathbf{X_i}$, as defined in (1), we denote the lth-picked activation as act_l, where $l \in [1, k]$. Here we have

$$act_l = \max\left(\mathbf{X_i} \ominus \sum_{j=1}^{l-1} act_j\right) \tag{4}$$

where the symbol $\ominus$ means removing elements from an assemblage. The summation symbol in (4) indicates a small set of elements, which contains top1~ $(l-1)$ activation but not adding up all the activation numerically.

After picking out the top-k activation, we do not simply add them together as an output or just compute the average of them. We propose a hyperparameter σ as a constraint factor to multiply the sum of the top-k activation.

The final output refers to

$$\text{output} = \sigma * \sum_{j=1}^{k} act_j \tag{5}$$

Here, the summation symbol means sum operation, where $\sigma \in (0, 1)$. Particularly, if $\sigma = 1/k$, the output is the average value.

The constraint factor, σ, is used to adjust the output. When few features remain active after ReLU gate, i.e., few positive values can move forward, if σ is set too small, positive values will be heavily weakened. On the contrary, when more features remain active, if σ is set more close to 1, output values will be big and easy to distort. In practice, the value of σ is influenced by the value of k. We carry out a lot of experiments and finally get a group of proper combination of σ and k shown in Table 2.

3.3. Network Architectures. We investigate two different plain network architectures in popularity: VGG and ALL-CNN series. We mainly do research on VGG-11 and VGG-13 architectures shown in Table 1. We also pick one model from ALL-CNN series as our baseline model, which is shown in

Table 1: Baseline architectures. Models A and E are two baseline architectures [13] with slightly different number of filters in some layers. Others are architectures with large-scale pooling layers or MAP layers. Conv3 indicates that the filter kernel sizes are 3×3. The ReLU gate is not shown for simplicity. The structures between small-size datasets (CIFAR, SVHN) and large-scale datasets (ImageNet) are different in input layers and fully connected layers. Parameters on the right of the slash belong to large-scale datasets.

Networks architectures based on VGG							
VGG-11					VGG-13		
A	B	C	D	E	F	G	H
Small-size datasets: Input (32×32 RGB image)/Large-scale datasets: Input (224×224 RGB image)							
conv3-64	conv3-64	conv3-64	conv3-64	conv3-64	conv3-64	conv3-64	conv3-64
2×2 maxpool	conv3-128	conv3-128	conv3-128	conv3-64	conv3-64	conv3-64	conv3-64
	conv3-128	conv3-128	conv3-128	**2×2 maxpool**	conv3-128	conv3-128	conv3-128
	4×4 maxpool/MAP	conv3-128	conv3-128		conv3-128	conv3-128	conv3-128
conv3-128		conv3-256	conv3-256	conv3-128	conv3-256	conv3-256	conv3-256
2×2 maxpool		**8×8 maxpool/MAP**	conv3-256	conv3-128	**4×4 maxpool/MAP**	conv3-256	conv3-256
			16×16 maxpool/MAP	**2×2 maxpool**		conv3-256	conv3-256
						8×8 maxpool/MAP	conv3-512
conv3-256				conv3-256			**16×16 maxpool/MAP**
conv3-256				conv3-256			
2×2 maxpool				**2×2 maxpool**			
	conv3-128				conv3-256		
	conv3-256				conv3-256		
	conv3-256				conv3-512		
conv3-512	**4×4 maxpool/MAP**			conv3-512	**4×4 maxpool/MAP**		
conv3-512				conv3-512			
2×2 maxpool		conv3-256		**2×2 maxpool**		conv3-512	
		conv3-512				conv3-512	
		conv3-512				conv3-512	
conv3-512	conv3-512	**4×4 avgpool**	conv3-512	conv3-512	conv3-512	**4×4 avgpool**	conv3-512
conv3-512	conv3-512		conv3-512	conv3-512	conv3-512		conv3-512
2×2 avgpool	**2×2 avgpool**		**2×2 avgpool**	**2×2 avgpool**	**2×2 avgpool**		**2×2 avgpool**
fc1-512×512/25088×4096							
fc2-512×512/4096×4096							
fc3-512×10/4096×1000							
Softmax-10/1000							

Table 2: The optimal combinations of σ and k under different pooling size.

pooling size	4×4	8×8	16×16
(k, σ)	(3, 0.5)	(4, 0.5)	(8, 0.25)

Table 5. For network with shortcuts structure, we pick one model from DenseNets series as our baseline model, which is shown in Table 7.

VGG. The series of VGG architectures first adopt small size (3×3) receptive field and use large number of filters (512) with much more trainable parameters in each convolutional layers. VGG architectures also put more convolutional layers together before pooling layers, which better extract features.

ALL-CNN. ALL-CNN first uses convolutional layers to subsample the feature maps. Compared to VGG, ALL-CNN are designed with less trainable parameters.

DenseNets. DenseNets are typical deep convolutional neural networks with shortcuts structure. The architecture alternatively uses Dense Blocks and Transition layers to extract features [15]. In every Transition layer, structures such as "BN-ReLU-conv1-avgpool (2×2)" are adopted to subsample the features.

3.4. Implementation. On VGG nets, all convolutional layers have equal kernel size 3×3 together with the same padding strategy, zero-padded by one pixel. On ALL-CNN, 1×1 convolutional layers are also included. When applying large-scale pooling method or MAP to certain architectures, total number of layers and trainable parameters are kept the same as baseline models. Before the fully connected layers and softmax classifier, average-pooling method is adopted in every model. When dealing with two key hyperparameters, σ and k, we use different combinations according to different pooling size. Table 2 shows the optimal combinations in experiment.

4. Experimental Results

4.1. Datasets

CIFAR. The two benchmark datasets, CIFAR-10 and CIFAR-100 [31], are consisted of tiny RGB images with 32×32 pixels. Both of the two datasets contain 50000 training images and 10000 testing images. Before training, we adopt a standard data normalization method [23] using the channel means and standard deviations. For training data augmentation, we randomly apply horizontal flip and crop on 4 pixels padded images. Finally, we evaluate the classification result by the test error rate.

SVHN. The Street View House Number datasets is a real-world image dataset obtained from house numbers in Google Street View images, which contains 10 classes, one for each digit. The dataset contains 73257 digits for training, 26032 digits for testing, and 531131 additional, somewhat less difficult samples, to use as extra training data. The images in SVHN are in two formats: original images and MNIST-like images with 32×32 pixels, and we choose the latter in our experiments.

ImageNet. The ILSVRC 2012 classification dataset [47, 48] consists of 1.2 million images for training and 50,000 for validation, 1000 classes in total. We adopt the same data augmentation scheme for training images as in [16, 17] and apply a single-crop or 10-crop with size 224×224 at test. Our experiment is only based on a few models due to the limit of computing speed because of the lack of GPU device.

4.2. Training Details. For every network, we use stochastic gradient descent (SGD) with momentum of 0.9 and weight decay of 0.0005 in backpropagation process. The initial learning rate is set to 0.05. The batch size is set to 128 and the images in every batch are shuffled every epoch. We do not use BN [23] and LRN [11] in every layer of our networks in order to keep the sparsity of feature maps. In fully connected layers, 50% dropout rate is adopted. For small-size datasets (CIFAR and SVHN), the total number of epochs is set to 150 and learning rate will be divided by 10 every 45 epochs.

For ImageNet, we train models for 100 epochs with the batch size of 128. The learning rate is set to 0.1 initially and is lowered by 10 times at epoch 40 and 80. We only experiment this large-scale datasets on a few model of VGG-11 series (Table 1). Due to different baseline models, some training details will be slightly adjusted.

4.3. Classification on VGG. To evaluate the efficiency of our MAP method, we train the networks shown in Table 1 on VGG architectures. We evaluate top-1 error rates and show the result in Table 3.

Large-Scale Max-Pooling Method. When the depth of layers and the number of parameters keep unchanged, piling up more convolutional layers together with large-scale pooling area extract features better. Based on this theory, we first reconstruct the two VGG architectures and then replace the original 2×2 pooling layers with larger-scale pooling layers. As is shown in Table 1, model A and model E are two baseline models. Model B and model F use 4×4 pooling layers. Model C and Model G use 8×8 pooling layers, while model D and model H apply even larger pooling size 16×16. Increasing the size of pooling layers leads the number of pooling layers to decrease. The result in second row of Table 3 shows that large-scale max-pooling method decreases error rate.

MAP Method. Table 3 shows that large-scale max-pooling method helps improve the classification accuracy. Then, we keep using the large pooling size, but instead of using the max-pooling method, we replace it with MAP method. The results from the last row of Table 3 (both **top** and **bottom**) are the most noticeable. First, with MAP method, the lowest error rate based on VGG-11 architecture is 6.94%, dropping around 20% compared to the original error rate of baseline model.

Table 3: Error rates in the test dataset of CIFAR-10. **Top:** error rates based on VGG-11 model and its extension models. **Bottom:** error rates based on VGG-13 model and its extension models.

		VGG-11 error rate (%)	
Baseline model(model A)		8.72	
With large-scale pooling method	4×4 maxpool (model B)	8×8 maxpool (model C)	16×16 maxpool (model D)
	8.24	7.84	8.19
With MAP method	4×4 MAP (model B)	8×8 MAP (model C)	16×16 MAP (model D)
	8.07	**6.94**	7.76
		VGG-13 error rate (%)	
Baseline model(model E)		7.50	
With large-scale pooling method	4×4 maxpool (model F)	8×8 maxpool (model G)	16×16 maxpool (model H)
	7.08	7.22	7.35
With MAP method	4×4 MAP (model F)	8×8 MAP (model G)	16×16 MAP (model H)
	6.96	**6.85**	7.07

Table 4: Error rates in the test dataset of CIFAR-100, SVHN, and ImageNet on VGG models.

Models	CIFAR-100 (%)	SVHN (%)	ImageNet top-5 (%)
Baseline model(A)	30.96	3.04	10.86
With 4×4 large-scale pooling method(B)	30.37	2.87	-
With 4×4 MAP method(B)	30.04	2.44	-
With 8×8 large-scale pooling method(C)	29.45	2.36	10.54
With 8×8 MAP method(C)	**28.37**	**2.03**	**10.19**
Baseline model(E)	29.05	2.27	-
With 4×4 large-scale pooling method(F)	28.62	2.14	-
With 4×4 MAP method(F)	27.92	1.98	-
With 8×8 large-scale pooling method(G)	28.03	2.05	-
With 8×8 MAP method(G)	**27.4**	**1.89**	-

We get similar result when the baseline model is VGG-13. The error rate also drops when integrating the MAP method into models. Second, if we analyze the result by column, it is clear that models with MAP layers perform better than models with max-pooling layers. We also try average-pooling method. Most of the results are even poorer than the results of baseline model.

Noticeably, the relationship between pooling size and error rate is not monotonic. Through experiment, we find that blindly increasing the pooling size is of no use.

On CIFAR-100 datasets, models with MAP method work much better than baseline models. For the reason that 8×8 MAP method is the strongest (Table 3), we pick 2 models with this method as comparison. Results are shown in Table 4. When integrating MAP method in VGG-11, the error rate decreases around 2.5%. In VGG-13, the error rate comes to 1.5%. With MAP method, model C performs even better than model E that has two more layers and more trainable parameters. On SVHN datasets, model with 8×8 MAP method is also the strongest as Table 3 shows. The classification results prove that MAP method with large pooling kernel size can still work well, while the number of classes are larger and the training images of each class are fewer.

4.4. Classification on ALL-CNN. Through changing the stride from 1 to 2, convolutional layers are able to subsample as well as pooling layers. Based on ALL-CNN, we first rearranged the convolutional layers and then integrated large-scale max-pooling and MAP method into the model (Table 5). We only experimented on CIFAR-10 datasets and got exciting classification results in Table 6. It shows that the architecture with 8×8 MAP method lowers the error rate to 7.11%, which outperforms the result (8.07%) of our baseline model and even a little better than the result in their paper (7.25%).

4.5. Classification on DenseNets. To integrate our MAP method into DenseNets, we change the original structure into "BN-ReLU-[conv3-ReLU] ×3-MAP (4×4)". As is shown in Table 7, DenseNet-40_MAP is the model with MAP method Based on DenseNet-40 with L = 40, k = 12, $\theta = 1$ [15].

Table 5: The baseline ALL-CNN architecture [24] and our modified conv-maxpool/MAP architectures.

conv-maxpool/MAP		All-CNN
Input (32×32 RGB image)		
conv3-96	conv3-96	conv3-96
conv3-96	conv3-96	conv3-96
conv3-96	conv3-96	
maxpool/MAP 4×4	conv3-192	conv3-96 stride=2
conv3-192	conv3-192	conv3-192
conv3-192		conv3-192
maxpool/MAP 4×4	**maxpool/MAP 8×8**	conv3-192 stride=2
conv3-192	conv3-192	conv3-192
conv3-192	conv3-192	conv1-192
		conv1-10
avgpool 2×2	avgpool 4×4	avgpool 6×6
fc1 192×192	fc1 192×192	Softmax-10
fc2 192×10	fc2 192×10	
Softmax-10	Softmax-10	

Table 6: Error rates of the baseline model ALL-CNN and its derived models on CIFAR-10 datasets.

Method	Error rate (%)	Params
ALL-CNN [24]	7.25	1.35M
ALL-CNN_ours	8.17	1.35M
conv-maxpool 4×4	8.79	1.35M
conv-maxpool 8×8	7.91	1.35M
conv-MAP 4×4	8.03	1.35M
conv-MAP 8×8	**7.11**	1.35M

Table 7: The baseline DenseNet-40 architecture [15] and our DenseNet-40_MAP architectures. Two models are both designed for CIFAR-10 datasets.

Layers	Densenet-40	Densenet-40_MAP
Convolution	conv3 stride 1 padding 1	
Dense Block(1)	[BN-ReLU-conv3] ×12	
Transition Layer(1)	[BN-ReLU] conv1 avgpool 2×2	[BN-ReLU] **[conv3-ReLU] ×3** **MAP 4×4**
Dense Block(2)	[BN-ReLU-conv3] × 12	
Transition Layer(2)	[BN-ReLU] conv1 avgpool 2×2	[BN-ReLU] **[conv3-ReLU] ×3** **MAP 4×4**
Dense Block(3)	[BN-ReLU-conv3] × 12	
Classification Layer	avgpool 8×8 10D fully-connected, softmax	avgpool 2×2

We used the code implemented on Pytorch at https://github.com/liuzhuang13/DenseNet. When training the two models, the initial learning rate is set to 0.1 and is divided by 10 at 50% and 75% of total 200 epochs [15]. The batch size is 64. The classification results are shown in Table 8. The model with plain structure (Figure 3, **right**) achieves lower error rate

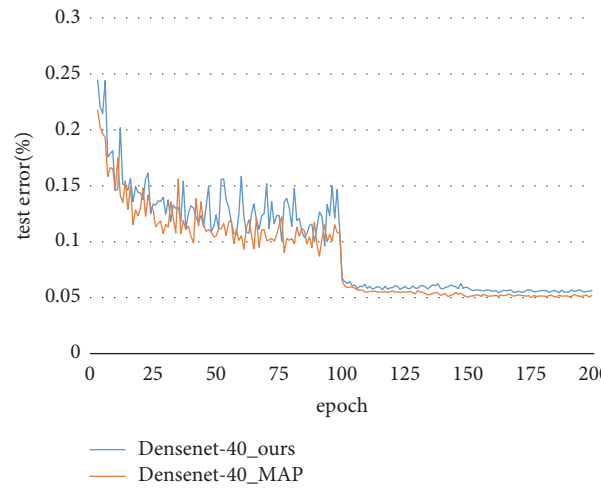


Figure 4: Testing error rate curves of DenseNet-40 and DenseNet-40_MAP on CIFAR-10.

on every small-size datasets (4.99% for CIFAR-10, 24.03% for CIFAR-100, 1.68% for SVHN) than baseline model. CIFAR-10 testing error rate curves show the strong performance of DenseNet-40_MAP (Figure 4) more clearly.

In plain networks, the basic structure like "conv3-ReLU-MAP" (Figure 3) is proved very effective on extracting features. In DenseNets, we integrate the structure into Transition layers to deal with output feature maps from Dense Blocks and classification results are encouraging.

4.6. Classification on ImageNet. We only used this large-scale datasets on model A and model C in our experiment. The results are shown in Table 4. Compared to VGG-11 baseline model, VGG-11 model with 8×8 MAP method lowers the top-5 error rate from 10.86% to 10.19%. Although limited results are listed, our results on large-scale datasets are still encouraging.

Table 8: Error rates in the test dataset of CIFAR and SVHN. The first row shows the error rate of the baseline model DenseNet-40 from paper [15]. The second row shows the error rate of our implement on the baseline model DenseNet-40. The third row shows the error rate of our DenseNet-40_MAP model in Table 7.

Method	CIFAR-10 (%)	CIFAR-100 (%)	SVHN (%)	Params
Densenet-40 [15]	5.24	24.42	1.79	1.05M
Densenet-40_ours	5.43	24.91	1.92	1.05M
Densenet-40_MAP	**4.99**	**24.03**	**1.68**	4.32M

Table 9: The four architectures are designed with fewer convolutional layers for CIFAR-10 datasets. Model A0 uses 2×2 max-pooling layer and 2×2 average-pooling layer. Model B0 adopts 8×8 MAP method. Model C0 uses 1×1 convolutional layer to replace the 3×3 convolutional layer right after MAP layer. Model D0 reduces one fully layer based on model C0. ReLU gates after each convolutional layer are not shown for simplicity.

VGG-9			
A0	B0	C0	D0
Input (32×32 RGB image)			
conv3-64 maxpool 2×2	conv3-64 conv3-128	conv3-64 conv3-128	conv3-64 conv3-128
conv3-128 maxpool 2×2	conv3-128 conv3-256	conv3-128 conv3-256	conv3-128 conv3-256
conv3-128 maxpool 2×2	**MAP 8×8**	**MAP 8×8**	**MAP 8×8**
conv3-256 maxpool 2×2	conv3-256 conv3-512	**conv1-256** conv3-256	**conv1-256** conv3-256
conv3-256 conv3-512 avgpool 2×2	avgpool 4×4	avgpool 4×4	avgpool 4×4
fc1 512×512		fc1 256×256	fc1 256×256
fc2 512×512		fc2 256×256	fc2 256×10
fc3 512×10		fc3 256×10	-
Softmax-10			

4.7. Classification on Compact Models

Model Complexity Analysis. We evaluate the model complexity mainly in several perspectives. First, we compare the number of parameters among VGG models. From Table 10, it can be seen that the two series model B-D and model F-H both have fewer trainable parameters than their baseline models. Compared to baseline model A, model B-D series cut trainable parameters from 9.8M to 5.4M. To achieve that, we simply reduce the number of filters in some convolutional layers. It is obvious that models with large-scale max-pooling method or MAP method show strong performance even when the trainable parameters are fewer. Second, each of the three models such as model B-D has the same number of trainable parameters and depth. MAP method is better than large-scale max-pooling method under the same condition.

All of our test models are simply composed of only three kinds of different layers: convolutional layers, pooling layers, and fully connected layers without adding any additional layers like BN [23] and LRN [11]. In addition, all of our searched networks are not that deep. It saves a lot of memory during training and it is easy to implement.

Compact Model Classification. Based on models with same number of trainable parameters and depth, we have proven our proposed method more effective. For further research, we keep reducing the number of layers. Based on model A, we drop two more convolutional layers and name the model series VGG-9 which is shown in Table 9. By comparing the classification results of model A, A0, and B on CIFAR-10 datasets (Figure 5), we conclude that the test error rate of model B0 that has MAP layers achieves 8.59%, even lower than the result of model A which has more layers and trainable parameters.

Table 10: The number of trainable parameters and depth in different VGG models are shown in Table 1. "Conv-D" indicates the number of convolutional layers and "FC-D" indicates the number of fully connected layers.

Models	Params	Depth	Conv-D	FC-D
A	9.8M	11	8	3
B-D	5.4M	11	8	3
E	9.9M	13	10	3
F-H	8.2M	13	10	3
A0	2.9M	9	6	3
B0	2.9M	9	6	3
C0	1.3M	9	6	3
C0+shortcuts	1.3M	9	6	3
D0	1.2M	8	6	2

Compared to model A, model B0 cuts down nearly 70% of the trainable parameters from 9.8M to 2.9M. However, 2.9M is still a large number. Therefore, more compact models with smaller amount of parameters are proposed in Table 10. Inspired by bottleneck structure [15], we choose to use 1×1 filters instead of 3×3 on the first convolutional layer locating after MAP layer in model C0. Compared to model B0, model C0 achieves better result (Figure 5) with more than 50% decline of trainable parameter (Table 10). Based on this encouraging result, we continue reducing one fully connected

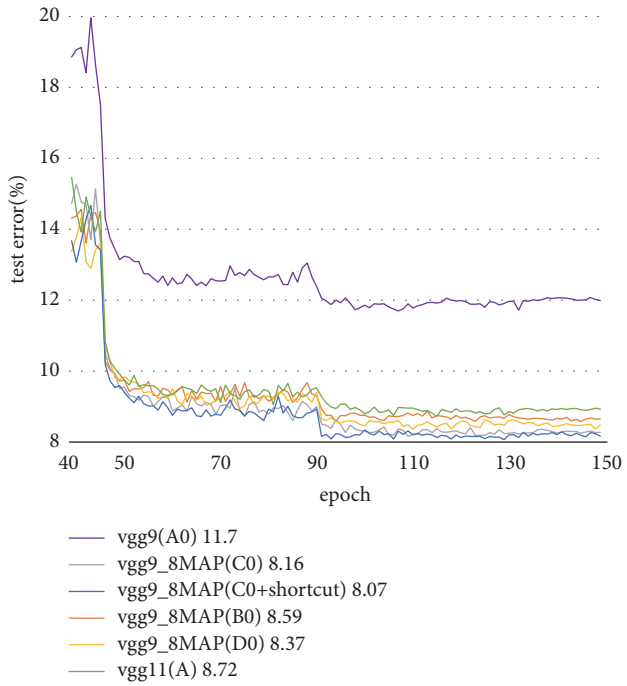


FIGURE 5: Testing error rate curves on CIFAR-10 of model A0 (vgg9), model B0 (vgg9_8MAP), model A (vgg11), model C0, model D0, and model C0 with shortcut structure. Values after each model name represent the lowest error rate (%).

layer and achieve close result to model C0. Both model C0 and D0 perform better than baseline model A with only 12% of the trainable parameters.

However, using the MAP method without increasing the number of parameters and layers cannot avoid increasing the computational complexity during training process, because more filters are forced to slide on bigger features maps.

5. Discussion

5.1. MAP Method Analysis

Sparsity Analysis. In CNNs, smaller receptive field size is proved more effective in most cases. Researches also show that piling up more convolutional layers extract features better. If basic blocks like "conv3-ReLU" are put together, the networks will be able to do multilevel template matching where negative values are set to zero by ReLU. Output feature maps are very sparse and activation left is quite important.

Figure 6 shows the output feature maps in different convolutional layers. Model A-C adopts different pooling size and the arrangement of the convolutional layers is different either. From left to right, the feature maps become more and more sparse. Theoretically, average-pooling is not suitable in sparse feature maps because useful activation will be heavily diluted. From top to bottom, we find that activation distributes densely in small local parts. So simply using max-pooling method may cause the loss of some good activation. Our MAP method finds a way to balance feature lost and feature dilution, which performs very well.

Visualization with Deconvolution. Deconvolution can be thought as a model that uses the same components (filtering, pooling) but in reverse. Here, deconvolution is used on our pretrained CNN model C and we use this method to map multiactivation back to the input pixel space, showing what input pattern eventually cause given activation in the feature maps. [36].

In this paper, we use basic structure (Figure 3 right) in our CNN models to thinning the feature maps. Every feature map in different convolutional layers can be regarded as a kind of representation of the original image. In other words, every feature map reflects characteristics of an original image. Through deconvolution, the reflection maps are visualized. In Figure 7, the deconvolution results on CIFAR-10 are shown. We use pretrain model C (Table 1), which contains an 8×8 pooling layers. There are 5 convolutional layers piling up before the large kernel-size-pooling layers. The channel numbers of each layer are 64, 128, 128, 128, and 256. As is shown in Figure 7, features maps become more and more sparse from the first convolutional layer to the fifth convolutional layer. Sparse feature maps often reflect the remarkable local characteristics of an image. In this way, our models can focus on different local parts specifically owned by different class of images, which represent raw images better.

5.2. Model Structure Analysis.

On plain networks and networks with shortcuts structure, MAP layers are always combined with plain structure shown in Figure 3. Through the analysis of feature maps sparsity and deconvolution, we believe that the plain structure can provide good activation and MAP method can make good use of this activation. In addition, the size of feature maps after 8×8 MAP layer becomes small (4×4 on CIFAR-10; 28×28 on ImageNet). We think that each activated feature (positive value in feature maps) on small feature maps is relatively independent. Therefore, trying to find the relationship between activated features in a 3×3 area may be unnecessary. Through experiment, we prove that 1×1 convolution can better represent the features picking out from MAP layer. At the same time, the trainable parameters are reduced a lot.

Since the MAP method can provide such good activation, we find a good way to take better advantage of this activation. Through mixing the plain structure and shortcuts structure, the classification results improve again! In Figure 8, the shortcuts structure is used to sum the output from MAP layer and the output from the 1×1 convolutional layer. We apply this structure to model C0 (Table 9) and name the new model "C0+shortcuts". All the parameters are keeping the same. The lowest error rate decreases from 8.16% to 8.07%. Noticeably, in Figure 5, the error rate curve of the new model lies beneath all the other curves, indicating that the overall capability of the new model is superior when fully trained.

5.3. Comparison between MAP and Mixing Pooling.

Mixing pooling method [44] mainly contributes to doing a trade-off between max-pooling and average-pooling. Both our method and theirs are based on a hypothesis that only using max-pooling method or average-pooling method cannot

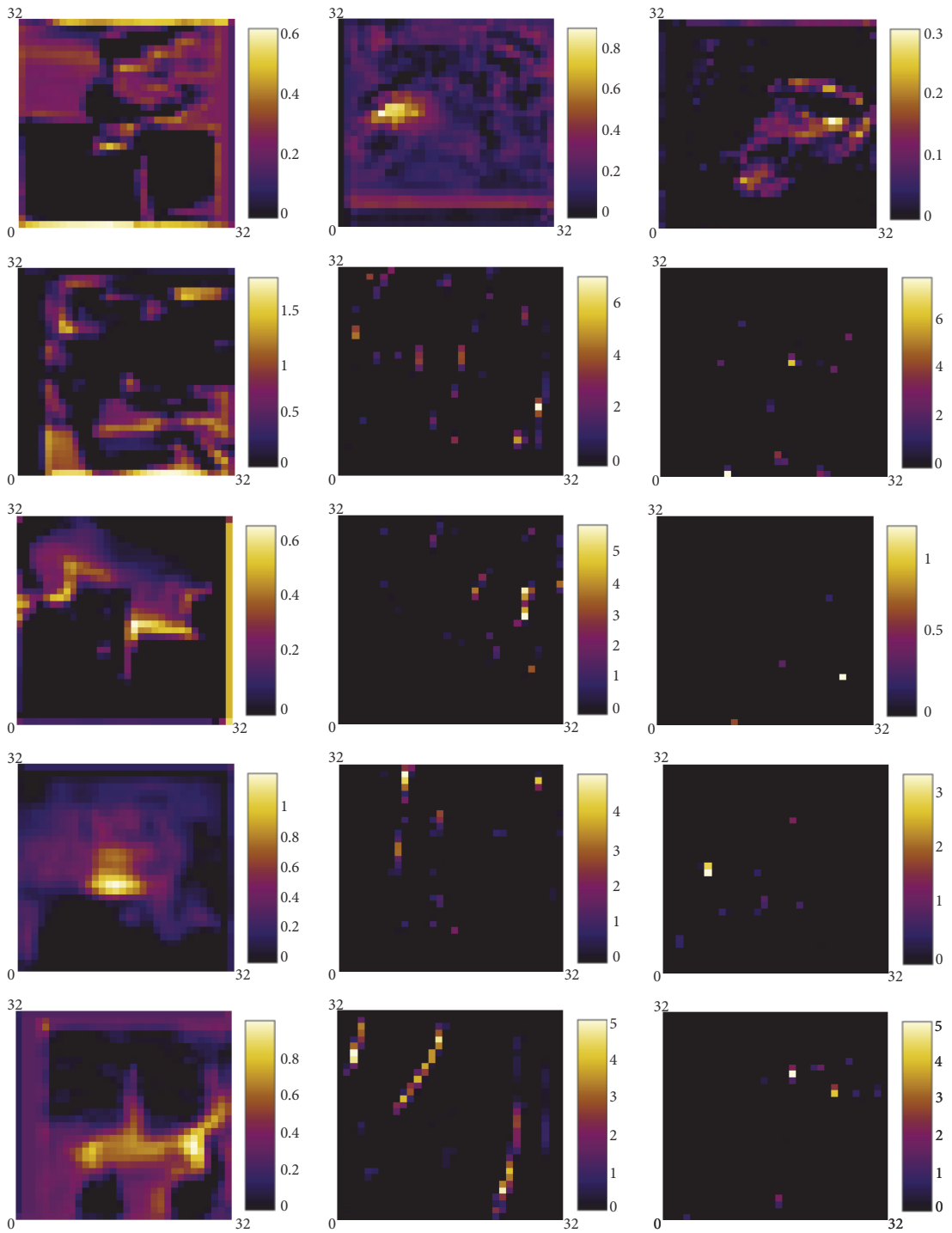


FIGURE 6: Each image indicates a certain convolutional layer's output feature map. Each column consists of 5 feature maps, which are extracted during training process of 3 different VGG-11 models. From top to bottom, the feature maps are extracted after training 1, 30, 60, 90, and 120 epochs, respectively. From left to right, the output feature maps are randomly picked from different convolutional layers, which locate right before the first pooling layer in each model (**left column:** the first convolutional layers in model A; **middle column:** the third convolutional layer in model B; and **right column:** the fifth convolutional layer in model C). Therefore, the kernel size of all maps is 32×32. The colour bar on the right of each map indicates the pixel intensity; i.e., the brighter the pixel colour, the bigger the pixel value. In addition, black corresponds to value 0.

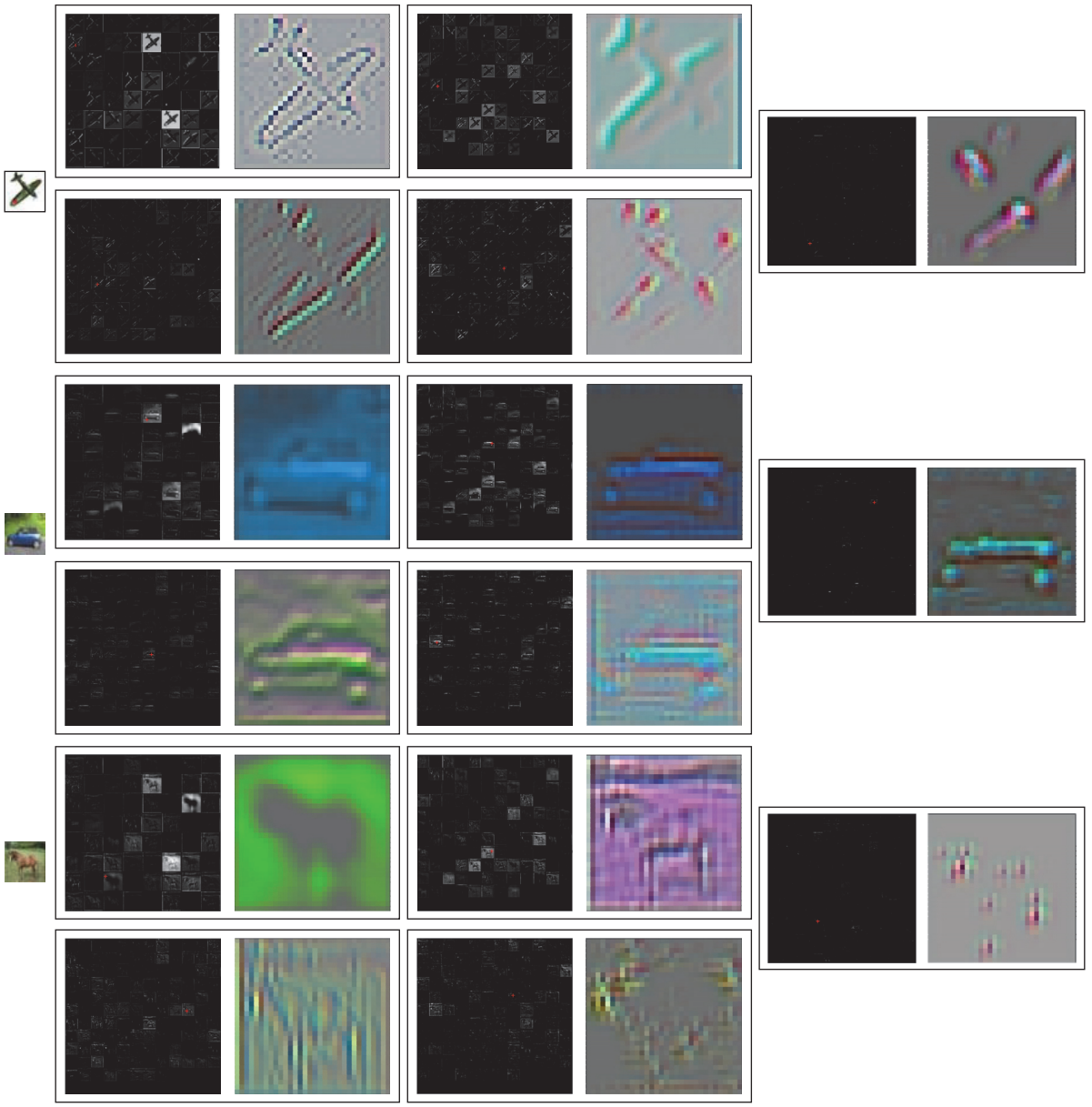


Figure 7: The left most three images are raw images from CIFAR-10 datasets. The first image is from class 0 (plane). The second image is from class 1 (car) and the third image is from class 7 (horse). Each raw image has 5 pairs of images on the right. The left image in each pair contains all channels' output feature maps from its corresponding convolutional layer. The right image in each pair is the reflection map of one feature map casually selected from all channels. The upper-left corner pairs are extracted from the first convolutional layer. The upper-right corner, lower-left, lower right, and the very right are extracted from the second, third, fourth, and fifth convolutional layer, respectively. In feature maps, black corresponds to 0 pixel value.

produce optimal results. However, MAP and mix pooling are quite different in the following aspects.

First, the novelty of mixing pooling method is bringing learning into the pooling operation. Two main expressions are

$$output = \alpha * \max(\boldsymbol{x}) + (1 - \alpha) * avg(\boldsymbol{x}) \tag{6}$$

$$output = \sigma\left(\mathbf{w}^{\mathrm{T}}\mathbf{x}\right) * \max(\boldsymbol{x}) + \left(1 - \sigma\left(\boldsymbol{w}^{T}\boldsymbol{x}\right)\right) * avg(\boldsymbol{x}) \tag{7}$$

where $\boldsymbol{x}$ denotes the values in the region being pooled; $\alpha \in [0, 1]$ is a scalar specifying a certain combination of max and average; $\boldsymbol{w}$ denotes the values in a "gating mask". $\sigma(\boldsymbol{w}^{T}\boldsymbol{x}) = 1/(1 + \exp(-\boldsymbol{w}^{T}\boldsymbol{x}))\epsilon[0, 1]$ is a sigmoid function. α and $\boldsymbol{w}$ are trainable.

Obviously, mixing pooling method combines max-pooling and average-pooling and constrains them with trainable factors. However, our MAP method only chooses top-k activation in a pooling region, which means most of the features are ignored.

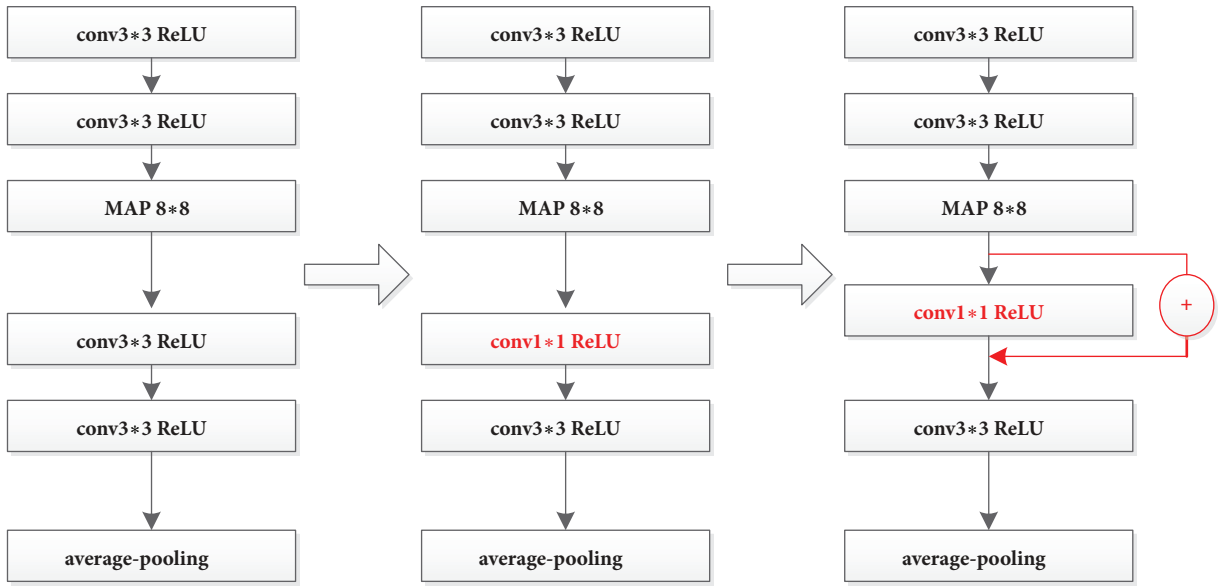


Figure 8: Plain structure and plain structure mixing with shortcuts structure.

Table 11: Different CIFAR-10 classification results with different combinations of k and σ based on VGG-11. The left two column results are based on the modelusing 4×4 MAP method. The right two column results are based on the model using 8×8 MAP method.

(k, σ)	result (%)	(k, σ)	result (%)
(1, 1)	8.24	(1,1)	7.84
(3, 0.5)	**8.07**	(4, 0.5)	**6.94**
(3, 0.75)	8.21	(4, 0.75)	7.14
(3, 0.33)	8.16	(4, 0.25)	7.07
(4, 0.5)	8.25	(6, 0.5)	7.15
(4, 0.25)	8.31	(6, 0.17)	7.33
(6, 0.5)	8.40	(8, 0.5)	7.57
(8, 0.5)	8.51	(8, 0.125)	7.63

Second, our MAP method is working on large pooling kernel size and takes good use of the sparsity of feature maps, while mixing pooling method does not take feature maps pattern and pooling kernel size into consideration.

Third, MAP method does not need more layers and trainable parameters. It can even achieve higher performance with fewer trainable parameters. On the contrary, mixing pooling method introduces trainable parameters into pooling layers to help improve model capacity.

5.4. Hyperparameter Analysis. In Table 2, the optimal combinations of σ and k under different pooling size are listed. Instead of arbitrarily choosing these two hyperparameters, we picked the best combination of σ and k through a great amount of experiments. The CIFAR-10 classification results with different hyperparameters based on VGG-11 are shown in Table 11. The first row of Table 11 corresponds to the max-pooling result and the fourth row corresponds to the average-pooling result. On large sparse feature maps, only choosing one or two features ignores quite big number of active features. On the contrary, choosing too much features adds a lot of useless closed features, which leads to feature dilution.

6. Conclusion

In this paper, we propose a new pooling method, which refers to Multiactivation Pooling (MAP) method. It provides a new subsampling idea to better pick features. We apply our method on convolutional neural networks. In our experiment, MAP method can highly improve the classification accuracy in CNNs. Actually, the plain structure "conv-ReLU-MAP" is a better feature-extractor in CNNs. Based on this structure, we also prove the feasibility of applying MAP method on DenseNets. Moreover, we carry out additional experiments such as sparsity analysis and deconvolution to illustrate that the plain structure can provide "good" activation.

We hope to study further and try to explore the potential of the basic layers. MAP method still has huge space for development. It may perform better by carefully fine-tuning hyperparameters, σ and k. In the future, we plan to study further on MAP method and test it on different datasets and different networks.

Data Availability

All the datasets in this paper are available on the Internet. Previously reported CIFAR (CIFAR-10 and CIFAR-100) data were used to support this study and are available at http://www.cs.toronto.edu/~kriz/cifar.html. Previously reported SVHN data were used to support this study and are available at http://ufldl.stanford.edu/housenumbers/. The ImageNet (ILSVRC2012) data used to support the findings of this study were supplied by http://image-net.org/download-images

under license and so cannot be made freely available. Signing up at http://image-net.org/download-images should be made for data availability.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

All the authors are supported by the National Natural Science Foundation of China (Grant 61772052).

References

[1] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, 2017.

[2] L. Zhao and K. Jia, "Multiscale CNNs for Brain Tumor Segmentation and Diagnosis," *Computational and Mathematical Methods in Medicine*, vol. 2016, Article ID 8356294, 2016.

[3] Y. Xu, G. Yu, Y. Wang, X. Wu, and Y. Ma, "Car detection from low-altitude UAV imagery with the faster R-CNN," *Journal of Advanced Transportation*, vol. 2017, Article ID 2823617, 11 pages, 2017.

[4] J. Shotton, J. Winn, C. Rother, and A. Criminisi, "TextonBoost for image understanding: multi-class object recognition and segmentation by jointly modeling texture, layout, and context," *International Journal of Computer Vision*, vol. 81, no. 1, pp. 2–23, 2009.

[5] P. Sermanet, D. Eigen, X. Zhang et al., *Overfeat: Integrated recognition, localization and detection using convolutional networks*, 2013, arXiv:1312.6229.

[6] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016*, pp. 779–788, July 2016.

[7] T. N. Sainath, B. Kingsbury, G. Saon et al., "Deep Convolutional Neural Networks for Large-scale Speech Tasks," *Neural Networks*, vol. 64, pp. 39–48, 2015.

[8] E. Shelhamer, J. Long, and T. Darrell, "Fully Convolutional Networks for Semantic Segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 4, pp. 640–651, 2017.

[9] E. Ribeiro, A. Uhl, G. Wimmer, and M. Häfner, "Exploring deep learning and transfer learning for colonic polyp classification," *Computational and Mathematical Methods in Medicine*, vol. 2016, Article ID 6584725, 16 pages, 2016.

[10] Z. Lv, H. Song, P. Basanta-Val, A. Steed, and M. Jo, "Next-Generation Big Data Analytics: State of the Art, Challenges, and Future Research Topics," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 4, pp. 1891–1899, 2017.

[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proceedings of the 26th Annual Conference on Neural Information Processing Systems (NIPS '12)*, pp. 1097–1105, Lake Tahoe, Nev, USA, December 2012.

[12] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2323, 1998.

[13] K. Simonyan and A. Zisserman, *Very deep convolutional networks for large-scale image recognition*, 2014, https://arxiv.org/abs/1409.1556.

[14] C. Szegedy, W. Liu, Y. Jia et al., "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '15)*, pp. 1–9, Boston, Mass, USA, June 2015.

[15] G. Huang, Z. Liu, L. v. Maaten, and K. Q. Weinberger, "Densely Connected Convolutional Networks," in *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2261–2269, Honolulu, HI, July 2017.

[16] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016*, pp. 770–778, July 2016.

[17] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *Proceedings of the European Conference on Computer Vision*, pp. 630–645, 2016.

[18] R. K. Srivastava, K. Greff, and J. Schmidhuber, "Training very deep networks," in *Proceedings of the 29th Annual Conference on Neural Information Processing Systems, NIPS 2015*, pp. 2377–2385, can, December 2015.

[19] G. Larsson, M. Maire, and G. Shakhnarovich, *Fractalnet: Ultra-deep neural networks without residuals*, 2016, https://arxiv.org/abs/1605.07648.

[20] Y. Yang, H. Li, Y. Han, and H. Gu, "High resolution remote sensing image segmentation based on graph theory and fractal net evolution approach," in *Proceedings of the 2015 ISPRS International Workshop on Image and Data Fusion, IWIDF 2015*, pp. 197–201, usa, July 2015.

[21] H. Lei, T. Han, F. Zhou et al., "A deeply supervised residual network for HEp-2 cell classification via cross-modal transfer learning," *Pattern Recognition*, vol. 79, pp. 290–302, 2018.

[22] Y. LeCun, B. Boser, J. S. Denker et al., "Backpropagation applied to handwritten zip code recognition," *Neural Computation*, vol. 1, no. 4, pp. 541–551, 1989.

[23] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proceedings of the 32nd International Conference on Machine Learning (ICML '15)*, pp. 448–456, July 2015.

[24] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, *Striving for simplicity: The all convolutional net*, 2014.

[25] N. Cohen, O. Sharir, and A. Shashua, "Deep SimNets," in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016*, pp. 4782–4791, usa, July 2016.

[26] F. N. Iandola, M. W. Moskewicz, K. Ashraf et al., *Squeezenet: Alexnet-level accuracy with 50x fewer parameters and< 0.5 MB model size*, 2016, arXiv:1602.07360.

[27] A. G. Howard, M. Zhu, B. Chen et al., *Mobilenets: Efficient convolutional neural networks for mobile vision applications*, 2017, arXiv:1704.04861.

[28] X. Zhang, X. Zhou, M. Lin, and J. Sun, *ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices*, 2017, arXiv:1707.01083.

[29] Z. Li, X. Wang, X. Lv, and T. Yang, *SEP-Nets: Small and Effective Pattern Networks*, 2017, 1706.03912.

[30] V. Nair and G. E. Hinton, "Rectified linear units improve Restricted Boltzmann machines," in *Proceedings of the 27th International Conference on Machine Learning (ICML '10)*, pp. 807–814, Haifa, Israel, June 2010.

[31] A. Krizhevsky, "Learning multiple layers of features from tiny images," Tech. Rep., 2009.

[32] D. H. Hubel and T. N. Wiesel, "Receptive fields, binocular interaction, and functional architecture in the cat's visual cortex," *The Journal of Physiology*, vol. 160, pp. 106–154, 1962.

[33] K. Fukushima and S. Miyake, "Neocognitron: A new algorithm for pattern recognition tolerant of deformations and shifts in position," *Pattern Recognition*, vol. 15, no. 6, pp. 455–469, 1982.

[34] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[35] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '05)*, vol. 1, pp. 886–893, June 2005.

[36] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Computer Vision—ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6–12, 2014, Proceedings, Part I*, vol. 8689 of *Lecture Notes in Computer Science*, pp. 818–833, Springer, 2014.

[37] Y.-L. Boureau, F. Bach, Y. LeCun, and J. Ponce, "Learning mid-level features for recognition," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '10)*, pp. 2559–2566, San Francisco, Calif, USA, June 2010.

[38] J. Yang, K. Yu, Y. Gong, and T. Huang, "Linear spatial pyramid matching using sparse coding for image classification," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '09)*, pp. 1794–1801, June 2009.

[39] M. D. Zeiler and R. Fergus, *Stochastic pooling for regularization of deep convolutional neural networks*, 2013, arXiv:1301.3557.

[40] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[41] H. Wu and X. Gu, "Towards dropout training for convolutional neural networks," *Neural Networks*, vol. 71, pp. 1–10, 2015.

[42] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 9, pp. 1904–1916, 2015.

[43] A. Tamar, Y. Wu, G. Thomas, S. Levine, and P. Abbeel, "Value iteration networks," in *Proceedings of the 30th Annual Conference on Neural Information Processing Systems, NIPS 2016*, pp. 2154–2162, esp, December 2016.

[44] C. Y. Lee, P. W. Gallagher, and Z. Tu, "Generalizing pooling functions in convolutional neural networks: Mixed, gated, and tree," in *Artificial Intelligence and Statistics*, pp. 464–472, 2016.

[45] M. Lin, Q. Chen, and S. Yan, *Network in network*, 2013, arXiv:1312.4400.

[46] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Learning and transferring mid-level image representations using convolutional neural networks," in *Proceedings of the 27th IEEE Conference on Computer Vision and Pattern Recognition (CVPR '14)*, pp. 1717–1724, IEEE, Columbus, Ohio, USA, June 2014.

[47] J. Deng, W. Dong, and R. Socher, "ImageNet: a large-scale hierarchical image database," in *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 248–255, Miami, Fla, USA, June 2009.

[48] O. Russakovsky, J. Deng, H. Su et al., "ImageNet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 3794175, 13 pages
https://doi.org/10.1155/2018/3794175

Research Article

Processing Optimization of Typed Resources with Synchronized Storage and Computation Adaptation in Fog Computing

Zhengyang Song,[1] Yucong Duan,[1] Shixiang Wan,[2] Xiaobing Sun,[3] Quan Zou,[2] Honghao Gao,[4,5] and Donghai Zhu[1]

[1] *State Key Laboratory of Marine Resource Utilization in the South China Sea, College of Information Science and Technology, Hainan University, Haikou, China*
[2] *School of Computer Science and Engineering, Tianjin University, Tianjin, China*
[3] *School of Information Engineering, Yangzhou University, Yangzhou, China*
[4] *Computing Center, Shanghai University, Shanghai, China*
[5] *Shanghai Key Laboratory of Intelligent Manufacturing and Robotics, Shanghai, China*

Correspondence should be addressed to Yucong Duan; duanyucong@hotmail.com

Received 27 January 2018; Accepted 16 April 2018; Published 30 May 2018

Academic Editor: Xuyun Zhang

Wide application of the Internet of Things (IoT) system has been increasingly demanding more hardware facilities for processing various resources including data, information, and knowledge. With the rapid growth of generated resource quantity, it is difficult to adapt to this situation by using traditional cloud computing models. Fog computing enables storage and computing services to perform at the edge of the network to extend cloud computing. However, there are some problems such as restricted computation, limited storage, and expensive network bandwidth in Fog computing applications. It is a challenge to balance the distribution of network resources. We propose a processing optimization mechanism of typed resources with synchronized storage and computation adaptation in Fog computing. In this mechanism, we process typed resources in a wireless-network-based three-tier architecture consisting of Data Graph, Information Graph, and Knowledge Graph. The proposed mechanism aims to minimize processing cost over network, computation, and storage while maximizing the performance of processing in a business value driven manner. Simulation results show that the proposed approach improves the ratio of performance over user investment. Meanwhile, conversions between resource types deliver support for dynamically allocating network resources.

1. Introduction

With the extensive development of IoT applications, it is necessary to promote storage, computing, and communication capability of IoT devices, such as efficient storage, semantic integration, and parallel processing. Due to powerful computing capability compared with edge devices, cloud computing puts all the computing tasks on the cloud to process data efficiently and provides highly vitalized storage and computing services on top of massively parallel distributed systems [1–3]. However, with the growing quantity of data generated at the edge, the speed is becoming the bottleneck for the cloud-based computing paradigm [4]. Also, future wireless access networks will face the challenge of transporting the ever-increasing volume of data centric traffic with ever tighter timing and quality of service (QoS) requirements [5]. In order to reduce the communication bandwidth needed between edge devices and the central data centre, nascent technologies and applications for mobile computing and IoT are driving computing towards dispersion [6]. Conventional methods, which are based on bandwidth allocation scheme, improve effectively bandwidth utilization efficiency [7, 8]. Some methods [9–11] achieve real-time data analysis, but they have not considered user investment [12, 13], processing cost [14], and the efficient utilization of IoT resources with reduced risks [15, 16] of security and robustness [17] issues.

Fog computing stresses the proximity to terminal users and client objectives, dense geographical distribution and local resource pools, latency reduction, and backbone bandwidth savings to achieve better quality of service (QoS) [18].

Meanwhile, the computing paradigm pushes data analytic and knowledge generation power away from centralized nodes to the logical extremes of a network near the source of the data. Therefore, a large number of heterogeneous, complexity, and hierarchy resources can be processed effectively in fog computing systems. However, storage and computation power in fog computing systems are not balanced. We argue that it is a challenge to provide better load balancing of resources in the Internet of Things. From the perspective of resource owners, obtaining cost-effective resources processing services remains the primary concern for adoption of fog computing services. In [19], the authors not only studied the tradeoff between network performance improvement and the communication overhead of transmitting network information but also proposed to optimize resource allocation in wireless networks.

Fog devices have increasingly generated a large amount of resources including data, information, and knowledge. In [20], Chaim illustrated the concepts of defining data, information, and knowledge. We use Knowledge Graph to solve the problem of extracting relationships from sources of knowledge [21, 22]. Knowledge Graph has become a powerful tool for representing knowledge in the form of labelled digraphs and gives textual information semantics. Knowledge base contains a set of concepts, examples, and relationships [23]. In [24], Duan et al. clarified the architecture of Knowledge Graph in terms of data, information, knowledge, and wisdom. In [25], the authors proposed to designate the form of Knowledge Graph as four basic forms: Data Graph, Information Graph, Knowledge Graph, and Wisdom Graph. In [26], the authors proposed to answer five Ws questions through constructing the architecture composed of Data Graph, Information Graph, Knowledge Graph, and Wisdom Graph. The development of software service system can be divided into data sharing, information transmission, and knowledge creation stages in terms of data, information, and knowledge [27]. In fog computing systems, the computing resources for providing device services are highly limited such as storage capacity, computing power, and network bandwidth. The demand for huge search space and computation power has become a big challenge. We propose a mechanism to improve the ratio of performance over investment when providing resources processing services. The proposed approach minimizes processing cost over network, computation, and storage while maximizing the performance of processing in a business value driven manner. Multitypes users' investment and corresponding benefit rate decide final resource combination cases, which makes the system have a relatively stable state. Furthermore, we design several simulations to verify the feasibility and evaluate the performance of the proposed approach. Currently we did not consider negative situations of which improper arrangement of resources might lead to penalties [28] instead of rewards.

The rest of this paper is organized as follows. In Section 2, we define typed resources and explain the proposed system model. Section 3 provides a running example. Section 4 presents practice of storage and computing collaborative adaptation towards typed IoT resources. Section 5 reports the simulation and summarizes the simulation results. Section 6 reports related work. Finally, we draw conclusions and outline aspects of future work in Section 7.

2. Definitions and System Model

In this section, we propose the definition of typed resources. Furthermore, we design the processing architecture of typed resources in the Internet of Things. In order to satisfy the transmission demands of nodes in a given wireless network, we construct the system model through restricting network resources.

2.1. Definitions of Typed Resources. Wide application of the Internet of Things has been acquiring a huge amount of typed resources. We classify resources into three types: data, information, and knowledge [27]. Data is not specified for stakeholder or machine. Data is the collection of discrete elements and concepts. Information is not specified for stakeholder or machine. Information is conveyed through conceptual mapping and combination of data and is used for interaction. Knowledge is used to reason and predict unknown resources such as data, information, and knowledge. Table 1 provides an explanation of typed resources of data, information, and knowledge. Definitions of resource elements and graphs are as follows.

Definition 1 (resource element (RE_{DIK})). We define resource element as

$$RE_{DIK} \coloneqq \langle D_{DIK}, I_{DIK}, K_{DIK} \rangle . \quad (1)$$

D_{DIK}, I_{DIK}, and K_{DIK} represent data, information, and knowledge in DIK hierarchy, respectively, where DIK is an abbreviation of D_{DIK}, I_{DIK}, and K_{DIK}.

Definition 2 (graphs). In our previous work [25], we specified Knowledge Graph in a progressive manner as four basic forms: Data Graph, Information Graph, Knowledge Graph, and Wisdom Graph. In this work, we propose to specify the existing concept of Knowledge Graph in three layers. We define graphs as

$$Graph_{DIK} \coloneqq \langle (DG_{DIK}), (IG_{DIK}), (KG_{DIK}) \rangle . \quad (2)$$

DG_{DIK}, IG_{DIK}, and KG_{DIK} represent Data Graph, Information Graph, and Knowledge Graph, respectively. DG_{DIK} is used to model the temporal and spatial features of D_{DIK}. IG_{DIK} is a combination of related D_{DIK}. IG_{DIK} expresses the interaction and transformation of I_{DIK} between entities in the form of a directed graph. KG_{DIK} is a collection of statistical rules summarized from known resources.

In order to optimize resources processing, it is important to convert the types of resources in the Internet of Things. We give the explanations of the type conversion mechanism as follows.

(a) For conversion from D_{DIK} to I_{DIK}, in the absence of context, D_{DIK} is meaningless. We convert D_{DIK} to I_{DIK} through reorganizing D_{DIK}. The new collection of D_{DIK} corresponds to a different class or concept, which is I_{DIK}.

Table 1: An explanation of three types of resources.

	D_{DIK}	I_{DIK}	K_{DIK}
Semantic load	Not specified for stakeholder/machine	Specified for stakeholder/machine	Abstracting unknown resources
Form	Discrete elements	Related elements	Probabilistic or categorization
Usage	Identification of existence after conceptualization	Communication	Reasoning and predicting
Graph	DG_{DIK}	IG_{DIK}	KG_{DIK}

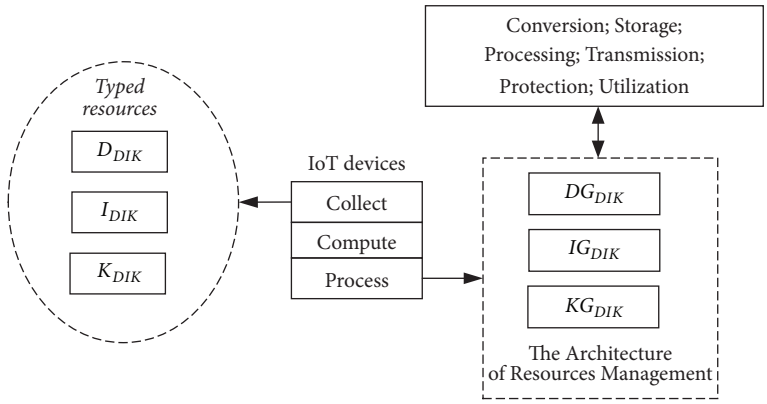


Figure 1: The processing architecture of typed resources in IoT.

(b) For conversion from D_{DIK} to K_{DIK}, D_{DIK} inherits semantic relationships from standard mode and is effectively integrated and reused by other applications. In the conversion process from D_{DIK} to K_{DIK}, we use knowledge validation technology to eliminate redundancy and inconsistency of D_{DIK} through linking D_{DIK} sources and semantic constraints. Then, we identify the most reliable D_{DIK} to form K_{DIK}.

(c) For conversion from I_{DIK} to K_{DIK}, I_{DIK} is used to express the interaction and collaboration between entities. Through the classifying and abstracting interactive records or behaviour records related to the dynamic behaviour of entities, we obtain K_{DIK} in the form of statistical rules. We infer K_{DIK} from known resources and collect necessary I_{DIK} in the process of inference through appropriate research techniques such as experiments and surveys.

(d) Regarding conversion from I_{DIK} to D_{DIK}, we know that conversion from I_{DIK} to D_{DIK} is the transition from concept set to resource instances. I_{DIK} expresses the dynamic interaction and collaboration between entities; we obtain D_{DIK} through observing an object at a certain time in a static state.

(e) For conversion from K_{DIK} to D_{DIK}, according to knowledge reasoning, we establish relevant examples for extracted collection of K_{DIK}. Relationships between K_{DIK} nodes are associated with instances in the form of attributes.

(f) For conversion from K_{DIK} to I_{DIK}, the schema-less feature of KG_{DIK} makes it possible to link and utilize a richer knowledge base to help users make decisions, from K_{DIK} retrieval to I_{DIK} creation.

2.2. Processing Architecture of Typed Resources in the Internet of Things. Generally, we obtain I_{DIK} through mining D_{DIK} and K_{DIK} through deducting I_{DIK} and intelligence from K_{DIK}. Therefore, D_{DIK}, I_{DIK}, K_{DIK}, and wisdom are the layers of a gradual relationship. The Knowledge Graph is divided into four levels: DG_{DIK}, IG_{DIK}, KG_{DIK}, and Wisdom Graph. Figure 1 shows the processing architecture of typed resources in IoT on the basis of DG_{DIK}, IG_{DIK}, and KG_{DIK}. In this architecture, graph types are allowed to convert between each other. We provide the definitions of DG_{DIK}, IG_{DIK}, and KG_{DIK} as follows.

Definition 3 (DG_{DIK}). We define DG_{DIK} as

$$DG_{DIK} \coloneqq \text{collection}\{\text{array, list, stack, queue, tree, graph}\}. \tag{3}$$

DG_{DIK} is a collection of discrete elements expressed in the form of various data structures including arrays, lists, stacks, trees, and graphs. DG_{DIK} featuring a static state is more about expressing a structural relationship. DG_{DIK} is used to model the temporal and spatial features of DG_{DIK}.

Definition 4 (IG_{DIK}). We define IG_{DIK} as

$$IG_{DIK} \coloneqq \text{combination}\{\text{related } D_{DIK}\}. \tag{4}$$

IG_{DIK} is a combination of related D_{DIK}. IG_{DIK} expresses the interaction and transformation of I_{DIK} between entities in the form of a directed graph. IG_{DIK} records the interaction between entities including both direct interaction and indirect interaction. Also, IG_{DIK} can be expressed using multiple tuples.

Definition 5 (KG_{DIK}). We define KG_{DIK} as

$$KG_{DIK} \coloneqq \text{collection}\{\text{Statistic Rules}\}. \tag{5}$$

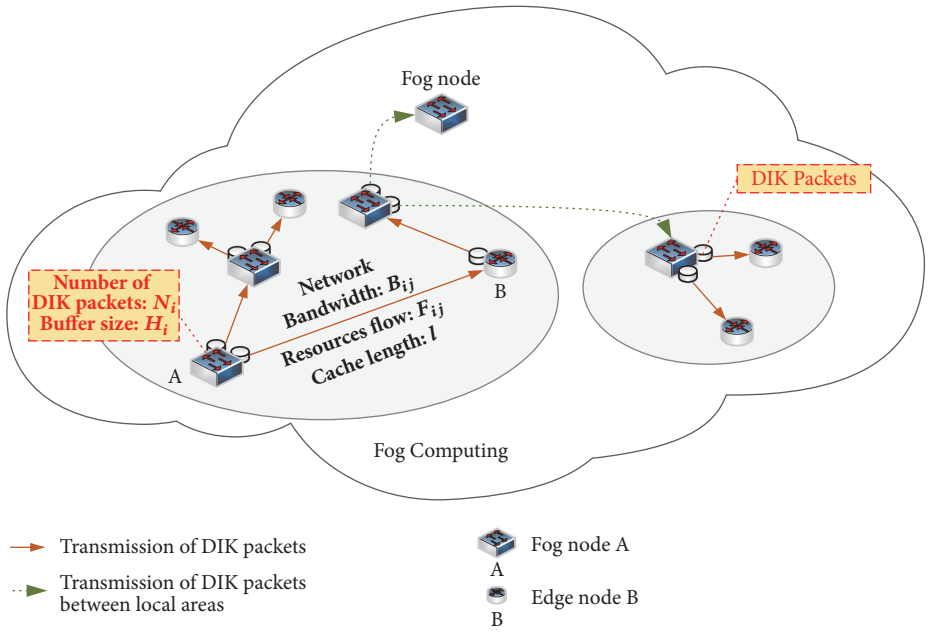


Figure 2: The transmission optimization model of typed resources in IoT (IoT_TOM).

KG_{DIK} is of free schema and expresses rich semantic relationships, which is conductive to have a completing mapping towards user requirements described through natural language. KG_{DIK} is a collection of statistical rules summarized from known resources. KG_{DIK} further improves and perfects the semantic relations between the entities on the basis of DG_{DIK} and IG_{DIK} and then forms a semantic network connected by a large number of interactive relationships.

2.3. System Model. We propose a transmission optimization model of typed resources in the Internet of Things, which is called IoT_TOM. Figure 2 shows a network composed of multiple wireless nodes. In this given network, A-B is a communication link between fog nodes A and B. We denote buffer size of each node as H_i. We denote the number of DIK packets to be forwarded as N_i. A DIK packet is a unit of resource made into a single package in forms of a resource combination in our proposed IoT_TOM. We denote the average length of DIK packets as l. We denote bandwidth between A and B as B_{AB}. We denote resource flow capacity on this link as F_{AB}. In practice, the distribution of network resources is not balanced in fog computing applications. The proposed mechanism of converting resource types achieves global optimization to deal with this problem. We use resources forwarding-waiting equilibrium and bandwidth utilization equilibrium to evaluate the performance of the proposed mechanism.

2.3.1. Bandwidth Utilization Equilibrium. Given a wireless network, we consider utilizing bandwidth efficiently. It is necessary to dynamically allocate bandwidth. Therefore, we consider transmitting DIK resource packets between IoT nodes, which aims to obtain enough storage and computing resources through consuming bandwidth. BE_{buse} denotes bandwidth utilization equilibrium. BE_{buse} is related to the network parameters such as bandwidth between IoT nodes and resource flow. Let IR_{ban} denote the bandwidth idle rate; then BE_{buse} is the variance of IR_{ban}. IR_{ban} and BE_{buse} can be illustrated as

$$IR_{ban} = \frac{B_{ij} - F_{ij} * l}{B_{ij}}, \quad \mathrm{i,j} = 1, 2, \ldots, \mathrm{n} \tag{6}$$

$$BE_{buse} = E\left(IR_{ban}{}^2\right) - \left(\left[E\left(IR_{ban}\right)\right]^2\right) \tag{7}$$

In (6), B_{ij} denotes bandwidth between IoT nodes. F_{ij} represents resource flow on the link.

2.3.2. Forwarding-Waiting Equilibrium. Resources forwarding-waiting time reflects the performance of the proposed IoT_TOM, which includes two parameters, forwarding-waiting rate (FR_{wait}) and forwarding-waiting equilibrium (BE_{wequ}). It is important to reduce the traffic load and congestion in the network in the absence of packet loss to balance resource flow. Let $N_i * l$ be the size of resources to be forwarded at node i; then BE_{wequ} is a variance of FR_{wait}. BE_{wequ} and FR_{wait} can be illustrated as

$$FR_{\mathbf{wait}} = \frac{N_i * l}{H_i}, \quad \mathrm{i} = 1, 2, \ldots, \mathrm{n} \tag{8}$$

$$BE_{\mathbf{wequ}} = E\left(FR_{\mathbf{wait}}{}^2\right) - \left(\left[E\left(FR_{wait}\right)\right]^2\right) \tag{9}$$

The cooperation between forwarding-waiting equilibrium and bandwidth utilization equilibrium enables users to utilize IoT resources more efficiently. We denote the objective function for defining utilization of typed resources in the Internet of Things as F, which can be illustrated as

$$\mathrm{F} = \alpha BE_{buse} + \beta BE_{\mathbf{wequ}}, \quad \alpha + \beta = 1 \tag{10}$$

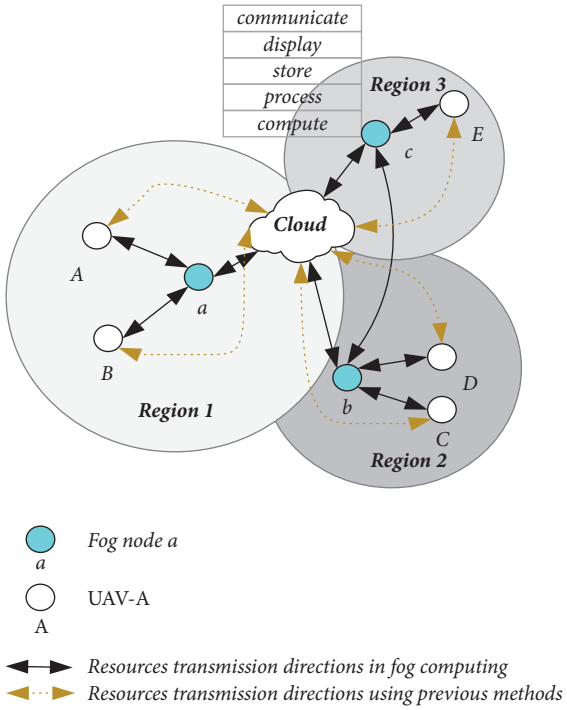


Figure 3: A UAVs courier service example in fog computing.

In (10), α and β, which can be obtained through data training, are parameters that contribute to BE_{buse} and BE_{wequ}, respectively. The smaller F is, the better resource flow capacity distribution performs.

3. Running Example to Provide UAVs Courier Service

Today's unmanned aerial vehicles (UAVs) courier service is a typical IoT application in fog computing. However, it is difficult to address bandwidth bottlenecks and management complexity of a UAV dispatch centre. For example, Figure 3 shows the geographical location distribution of UAVs providing courier services over the air. Dotted directional lines represent resources transmission directions using conventional methods, which may require expensive satellite navigation. The black directional lines display resource transmission directions of providing courier service using a fog computing system.

In an ideal situation, IoT resources without redundancy and inconsistency are transmitted in low user investments. The actual situation may not follow the ideal situation. For example, D_{DIK} resources of (*longitude* and *latitude*) featuring redundancy and inconsistency make the scale of D_{DIK} collection large. Therefore, if storage capacity at node UAV-A is not enough, we need to consume corresponding bandwidth to transmit these D_{DIK} resources to other nodes, which aims to obtain storage resources. We consider converting the resource type to I_{DIK} or K_{DIK} when transmitting these D_{DIK} resources, which aims to change the scale of initial D_{DIK} collection. Table 2 shows partial D_{DIK} resources generated by *UAV-A*. And Table 3 shows partial I_{DIK} resources generated by *UAV-B*.

Table 2: Partial D_{DIK} resources generated by *UAV-A*.

UAV	Unit D_{DIK} (8 bit)	Time range	Times	Space
UAV-A	(*longitude*, *latitude*)	2018.1.1, 9:00–12:00	5000	39.1 Kbit

For *UAV-A*, according to D_{DIK} resources generated from time t_1 (2018.1.1, 9:00) to t_2 (2018.1.1, 9:20), we sum up that two relationships between {*longitude, latitude*} and *time* are $longitude = k_1 * time + b_1$ and $latitude = k_2 * time + b_2$. In fact, we sum up 400 linear relationships between {*longitude, latitude*} and *time* roughly. Conversion from D_{DIK} to I_{DIK} is illustrated as

$$R\left(D_{DIK}\left[longitude, latitude\right]\right) \longrightarrow I_{DIK}\left[k_1 * time + b_1, k_2 * time + b_2\right] \tag{11}$$

We store these I_{DIK} resources converted from D_{DIK} resources in form of I_{DIK} $\{k_1 * time + b_1, k_2 * time + b_2), (k_1, b_1, k_2, b_2, time (t_1, t_2))$, which requires storage space of 32 bits. Then total storage space to store these I_{DIK} resources is $32 * 400 = 12.5$ Kbit.

Table 3 shows I_{DIK} resources generated by *UAV-B*. Through counting and summarizing these rules, we obtain 100 K_{DIK} resources roughly. Conversion from I_{DIK} to K_{DIK} is illustrated as

$$R\left(I_{DIK}\left[Path1, Path2\right]\right) \longrightarrow K_{DIK}\left[rain, down, UAV\text{-}B\right] \tag{12}$$

We store these K_{DIK} resources in form of K_{DIK} (*rain, down, UAV-B*), which requires storage space of 4 bits. Then total storage space to store these K_{DIK} resources is $4 * 250 = 0.97$ Kbit.

Hence, these conversions between resource types reduce the scale of collected resources. In our proposed resource type conversion mechanism, there are at most $27 * 27$ kinds of resource combinations in each node over the network. As shown in Figure 4, we take one resource combination case {(D, D, I), (DG, IG, IG)} as an example. Conversion from {D, D, I} into {I, I, K} aims to obtain storage resources through consuming computing capability. In general, we keep the system in a relatively stable state with storage and computing collaborative adaptation.

Because of complex courier tasks and geographic environment conditions of UAVs, resources distribution is unbalanced. Global optimization of the UAVs wireless network can solve this problem to a large extent. We design a mechanism that minimizes processing cost over network, computation, and storage while maximizing the performance of processing in a business value driven manner, which is illustrated in Section 4. Figure 5 displays resources allocation results in region 1 of the wireless network by using the proposed mechanism. Initial resource combinations on *UAV-A* and *UAV-B* are as follows:

Combination (UAV-A) = {(D_{DIK}, D_{DIK}, I_{DIK}), (DG_{DIK}, IG_{DIK}, IG_{DIK})}. And the resource scale on *UAV-A* is 12 Kbit, *bandwidth* ($A \rightarrow a$) is 6 Kpbs, and *buffer size (A)* is 2 Kbit.

Table 3: Partial I_{DIK} resources generated by *UAV-B*.

UAV	Unit I_{DIK} (8 bit)	Time range	Times	Space
UAV-B	*Path 1 (sky, has, rain)* *Path 2 (rain, damage, UAV-B)*	2018.1.1, 9:00–12:00	800	6.3 Kbit

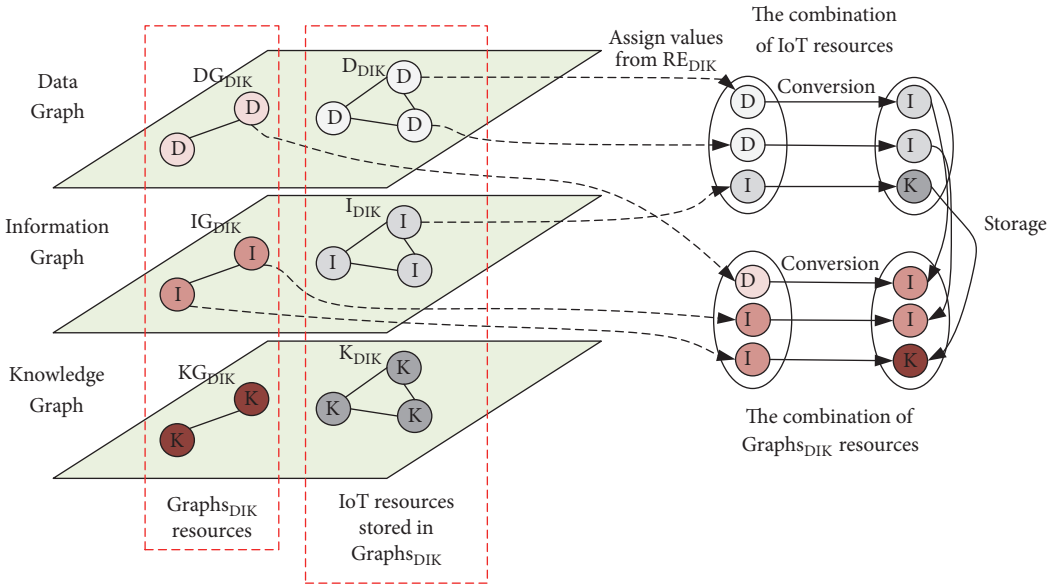


Figure 4: Storing resources according to the resource combination {(D, D, I), (DG, IG, IG)} which is one of the 27∗27 resource combinations.

Combination (UAV-B) = {(D_{DIK}, D_{DIK}, D_{DIK}), (KG_{DIK}, IG_{DIK}, IG_{DIK})}. And the resource scale on *UAV-A* is 26 Kbit, *bandwidth* ($B \rightarrow a$) is 6 Kpbs, and *buffer size (B)* is 2 Kbit.

Total storage spaces at node *UAV-B* and node *UAV-B* are 5 Kbit and 10 Kbit, respectively. Thus, we convert resource types and transmit these resources to node a, which is illustrated as follows:

Combination (UAV-A)' = {(D_{DIK}, I_{DIK}, K_{DIK}), (DG_{DIK}, IG_{DIK}, KG_{DIK})}. As shown in Figure 5(a), the resource scale on *UAV-A* is 24 Kbit and *bandwidth* ($A \rightarrow a$) = 4 Kpbs and *buffer size (A)* = 3 Kbit.

Combination (UAV-B)' = {(I_{DIK}, I_{DIK}, K_{DIK}), (DG_{DIK}, KG_{DIK}, IG_{DIK})}. As shown in Figure 5(b), the resource scale on *UAV-B* is 23 Kbit and *bandwidth* ($B \rightarrow a$) = 3 Kpbs and *buffer size (B)* = 2 Kbit.

However, there are two problems in implementing this idea. First, storage and computation capabilities of IoT devices in fog computing systems are different. Thus, it is necessary to obtain storage capacity by transferring typed resources. Second, both the performance of processing resources and user investments should be considered to improve user investment benefit.

4. Storage and Computing Collaborative Adaptation towards IoT Resources

In fog computing systems, there may be insufficient storage and computation capability at some nodes. We consider transmitting IoT resources to other nodes, aiming to obtain storage and computation resources through consuming bandwidth. Given a wireless network, some nodes consume storage capacity to obtain enough computing resources. Some nodes consume computing capability to obtain enough storage resources. In general, conversion between resource types provides a support for balancing the distribution of network resources in the Internet of Things.

In this section, we describe optimizing processing mechanism of typed resources with collaborative storage and computation adaptation. We define IoT resources and resources on $\text{Grpah}_{\text{DIK}}$ as follows.

Definition 6 (IoT resources). We define IoT resources as a tuple IR = < IRT, IRS >. IRT is the type set of IoT resources represented by a triad < irt_{D}, irt_{I}, irt_{K} >. IRS is the scale of different kinds of IoT resources represented by a triad < irs_{D}, irs_{I}, irs_{K} >. Each irs denotes the scale of resource in the form of irt.

Definition 7 (resources on $\text{Grpah}_{\text{DIK}}$). We define resources on $\text{Grpah}_{\text{DIK}}$ as a tuple RoG = <RGT, RGS>. RGT is the type set of resources on $\text{Grpah}_{\text{DIK}}$ represented by a triad < rgt_{D}, rgt_{I}, rgt_{K} >. RGS is the scale of different kinds of resources on $\text{Grpah}_{\text{DIK}}$ represented by a triad < rgs_{D}, rgs_{I}, rgs_{K} >. Each rgs denotes the scale of resources in the form of rgt.

4.1. Computation of Resource Type Conversion Cost. Considering the disadvantages of conventional resources processing methods, we propose to convert both types of resources in

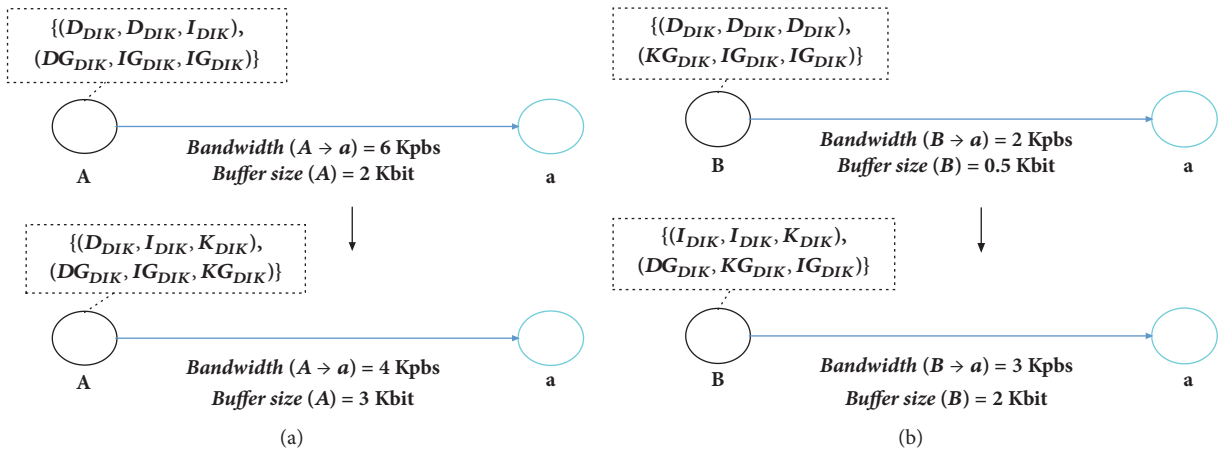


Figure 5: Resource allocation results over region 1 of the UAVs wireless network through using the proposed mechanism.

Table 4: Atomic conversion cost per unit resource in *IR*.

	D_{DIK}	I_{DIK}	K_{DIK}
D_{DIK}	$costCUnitIR_{D\to D}$	$costCUnitIR_{D\to I}$	$costCUnitIR_{D\to K}$
I_{DIK}	$costCUnitIR_{I\to D}$	$costCUnitIR_{I\to I}$	$costCUnitIR_{I\to K}$
K_{DIK}	$costCUnitIR_{K\to D}$	$costCUnitIR_{K\to I}$	$costCUnitIR_{K\to K}$

Table 5: Atomic conversion cost per unit resource in *ROG*.

	D_{DIK}	I_{DIK}	K_{DIK}
D_{DIK}	$costCUnitRG_{D\to D}$	$costCUnitRG_{D\to I}$	$costCUnitRG_{D\to K}$
I_{DIK}	$costCUnitRG_{I\to D}$	$costCUnitRG_{I\to I}$	$costCUnitRG_{I\to K}$
K_{DIK}	$costCUnitRG_{K\to D}$	$costCUnitRG_{K\to I}$	$costCUnitRG_{K\to K}$

IR and RoG. First, we would like to convert the types of resources in IR, which needs to invest corresponding cost. Specifically, we assign values from RE_{DIK} to each element of the type set IRT. Then we use these values to form the combination case IRT' = < irt_D', irt_I', irt_K' >, where irt_D', irt_I', and irt_K' belong to {D_{DIK}, I_{DIK}, K_{DIK}}. As shown in Table 4, we denote $costCUnitIR_{i\to j}$ as the atomic conversion cost per unit resource in IR. Conversion cost of resource types from IR to IRT' can be illustrated as

$$C_{IR} = \sum costCUnitIR_{i\to j} * irs_i, \quad i, j \in \{D, I, K\} \tag{13}$$

Second, we convert the types of resources in RoG. Specifically, we assign values from RE_{DIK} to each resource of the type set RGT of RG. Then we use these values to form the combination case RGT'= < rgt_D, rgt_I, rgt_K >, where rgt_D, rgt_I, and rgt_K belong to {D_{DIK}, I_{DIK}, K_{DIK}}. As shown in Table 5, we denote $costCUnitRG_{i\to j}$ as the atomic type conversion cost per unit resource in RoG. Conversion cost of resource types from RGT' to RoG can be illustrated as

$$C_{RG} = \sum costCUnitRG_{i\to j} * rgs_i, \quad i, j \in \{D, I, K\} \tag{14}$$

4.2. Computation of Cost of Processing IR in RoG. The proposed processing architecture of typed resources is adopted to process IoT resources. Cost of processing *IR* in *RoG* is related to the resource scale. As shown in Table 6, we denote $costPUnitIR_{i\to j}$ as the atomic cost of processing unit D_{DIK}, I_{DIK}, or K_{DIK} resource in corresponding graph resources. Then cost of processing *IR* in *RoG* can be illustrated as

$$C_{pro} = \sum \left(rgs_j + costPUnitIR_{i\to j} * rgs_i'\right) * irs, \quad i, j \in \{D, I, K\} \tag{15}$$

We convert the resource types in hopes of reducing the scale of resources when transferring IR in fog computing systems, aiming to balance resources load in a given wireless network. We assume that bandwidth of some nodes is enough in a period of time. We transmit IR to obtain storage and computation resources of other nodes through consuming bandwidth, aiming to satisfy users' general requirements. As shown in Table 7, let C_{transD}, C_{transI}, and C_{transK} indicate the atomic cost of transmitting unit D_{DIK}, I_{DIK}, or K_{DIK}, respectively. Cost of transmitting IR can be illustrated as

$$C_{trans} = \sum C_{transi} * irs_i, \quad i \in \{D, I, K\} \tag{16}$$

Table 6: Atomic type conversion cost of RE_{DIK}.

	D_{DIK}	I_{DIK}	K_{DIK}
D_{DIK}	$costPUnitIR_{D\to D}$	$costPUnitIR_{D\to I}$	$costPUnitIR_{D\to K}$
I_{DIK}	$costPUnitIR_{I\to D}$	$costPUnitIR_{I\to I}$	$costPUnitIR_{I\to K}$
K_{DIK}	$costPUnitIR_{K\to D}$	$costPUnitIR_{K\to I}$	$costPUnitIR_{K\to K}$

Table 7: Atomic cost of transmitting unit D_{DIK}, I_{DIK}, or K_{DIK} resource.

	D_{DIK}	I_{DIK}	K_{DIK}
C_{transi}	C_{transD}	C_{transI}	C_{transK}

4.3. Calculation of Users' Investment Benefit. We optimize the performance of processing resources through converting the type of resources and rational allocations of infrastructures in Internet of Things. According to previous subsections, we calculate the total cost of processing IR in fog computing

Table 8: Simulation setup.

No.	Input parameters	Setup
1	Number of sensor nodes	10
2	Number of fog nodes	4
3	Area of the wireless network field	500 m ∗ 400 m
4	Frequency band	2.4 GHz
5	Storage capability of each node (bit)	{A, B, C, D, E, F} → {128M, 8M, 256M, 8M, 16M, 4M}
6	Data width of CPU of each node	{A, B, C, D, E, F} → {8 bits, 16 bits, 16 bits, 8 bits, 32 bits, 8bits}
7	Energy model	Battery
8	Sensors	{A, B, C, D, E, F} → {Temperature, Humidity, GPS, Temperature, Photosensitive, GPS} sensors
9	Simulation time	10 minutes

applications by using our proposed mechanism. Total_Cost denotes the total cost of this resources processing service, which can be illustrated as

$$\text{Total\_Cost} = C_{IR} + C_{RG} + C_{pro} + C_{trans} \tag{17}$$

Let IV_{user} denote inquired user investment; then IV_{user} corresponding to each resource combination can be illustrated as

$$\text{IV}_{user} = \phi * \text{Total\_Cost} \tag{18}$$

where ϕ represents the atomic investment that can be obtained through data training. Different investment programs correspond to different benefit ratios (R_e). R_e is used to measure the ratio of performance over investment, which is illustrated as

$$R_e = \frac{C_{trans} + C_{pro}}{\text{IV}_{user}} \tag{19}$$

Then we compare R_e and IV_{user} of each program with R_{e0} and IV_{user0} to determine whether the condition "$R_e > R_{e0}$ & IV_{user}<IV_{user0}" is satisfied. Let R_{e0} be equal to the current R_e when R_e is greater than R_{e0}. Algorithm 1 describes the specific process of investment driven resource processing approach.

Input: *IR, RoG,* IV_{user0}, R_{e0}
Output: The minimum R_{e0}.
  For: each IRT **Do**
    Assign value from RE_{DIK};
    Compute C_{IR};
    Compute C_{RG}; Compute C_{pro};
    Compute C_{trans}; Compute *Total_Cost*;
    Compute IV_{user}; Compute R_e;
    If ($R_e < R_{e0}$ & $IV_{useri} < IV_{user0}$)
      $R_{e0} = R_e$;
  Return R_{e0}

Algorithm 1: Calculating R_e of each resource combination at all fog nodes.

5. Experiment

The goal of our experiments is to evaluate the proposed processing optimization approach towards typed resources. We first established a small wireless network as our simulation environment with discrete illustration [29]. We observed and analysed user investment benefit (R_e) of six nodes in this system. Then, we measured some necessary network indicators to illustrate effects of using the proposed mechanism to optimize resources processing. Finally, we compared our proposed IoT_TOM with a conventional mechanism.

5.1. Experiment Setup. As shown in Figure 6, we first established a small real wireless network with fourteen nodes. These nodes consisted of different kinds of Arduino MCUs. Six nodes of them were equipped with ESP8266 WI-FI modules and some sensors for collecting environment resources. As shown in Table 8, we set up some necessary input parameters that include senor nodes and energy model [30]. We deployed these nodes in four subareas in an area of 500 m ∗ 400 m.

5.1.1. Experiment Procedure. We aimed to illustrate the feasibility of our approach through providing users resources processing services. The proposed mechanism allowed each node to finish storage, computation, and transmission tasks. We assumed that a user demanded to observe environmental resources of this real wireless network anywhere. For convenience, we assigned values to some necessary parameters in the equations, such as $costCUnitIR_{\text{i}\to\text{j}}$, $costPUnitIR_{i\to j}$, and $costCUnitRG_{i\to j}$. We evaluated the performance of the proposed mechanism in terms of investment benefit and F value. The running time is ten minutes. In our experiment, we used some rules to process collected data into three resource collections of D_{DIK}, I_{DIK}, and K_{DIK}. For example, photosensitive sensors collected D_{DIK} resources of light intensity, which were {2, 4, 8, 16. . . 1024}. We processed these D_{DIK} resources into I_{DIK} resources in forms of 2^{x} (x= [1, 10]).

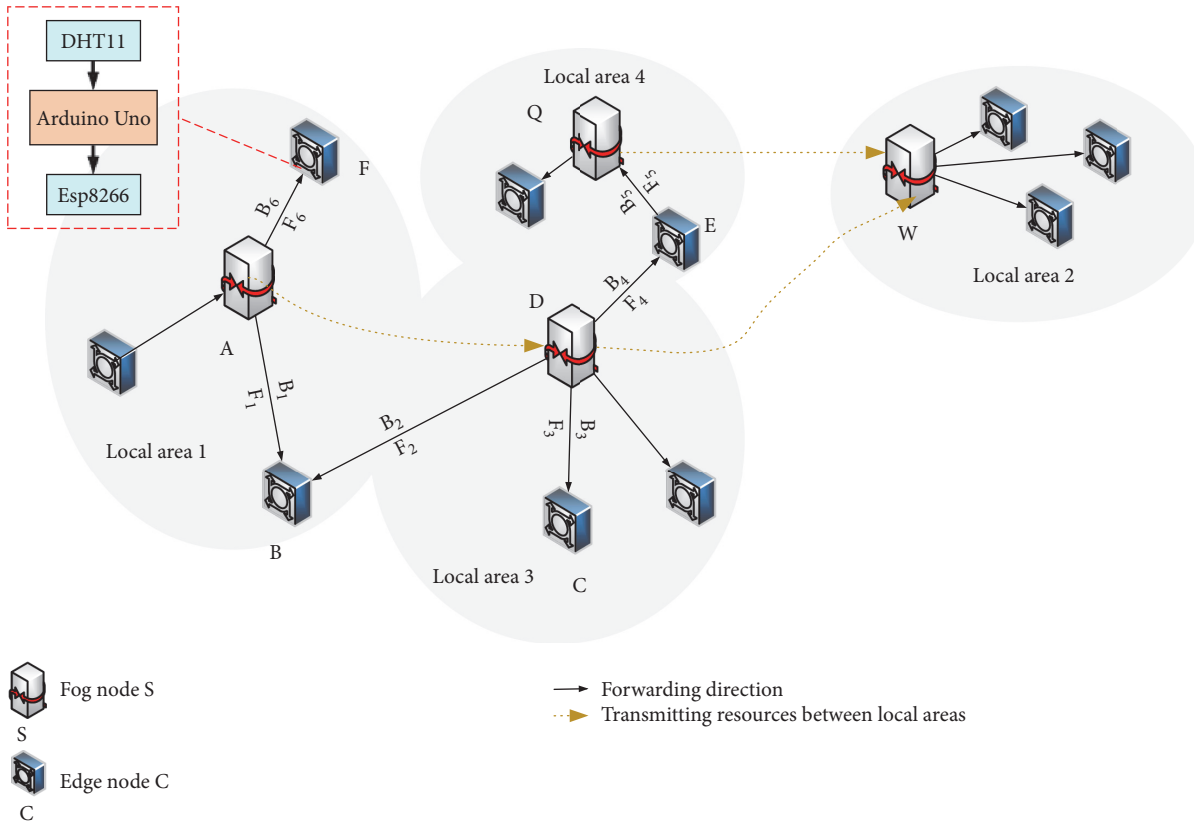


Figure 6: The established simulation environment.

Table 9 shows an example for classifying typed resources in a practical environment.

5.1.2. Statistics Analysis. For convenience, we only considered six resource combinations to illustrate the feasibility of the proposed mechanism. We estimated IV_{user} and R_e after calculating C_{IR}, C_{RG}, C_{trans}, and C_{pro}. As shown in Figure 7, points surrounded by the red box on the R_e curves indicate our recommended resource combinations. In these resource combinations, users are able to acquire resource services of maximum processing performance with minimum investment. In our experiments, the proposed mechanism dynamically recommended different resource combinations for different network environments to meet user demands better. Figure 7 shows that No. 6 of node A, No. 4 of node B, No. 5 of node C, No. 2 of node D, No. 4 of node E, and No. 2 of node F are final recommended resource combinations.

Furthermore, we measured some necessary parameters of the proposed IoT_TOM including F_{ij}, l, and N_i in order to observe the state of this wireless network. As shown in Table 10, FR_{wait} of link A-F is greater than that of other nodes. Storage capacity of node F is 4 Mbit, which is insufficient to store 7 Mbit IoT resources. Therefore node F consumed more bandwidths to obtain enough storage resources compared with other nodes. On the whole, some nodes consumed computing capability to obtain enough storage resources. Some nodes consumed storage capacity to obtain computing resources. Some nodes consumed bandwidth to obtain storage resources. In order to evaluate the performance of IoT_TOM, we estimated BE_{buse} and BE_{wait} which are 0.36 and 0.28, respectively. F value reflects resource flow capacity distribution. In this experiment, we calculated F value according to (10). The calculated F value is 0.34, which means that the proposed mechanism effectively allocated network resources through converting resource types. Table 10 shows that IR_{ban} and FR_{wait} of most nodes are smaller than 0.5, which is not the best result in terms of network parameters intuitively. In order to illustrate advantages of our proposed approach, in next subsection, we compared our proposed mechanism with a conventional mechanism under the same resources scale.

5.2. Comparison with Other Methods. We compared the proposed mechanism with a conventional bandwidth-allocation-based mechanism. We used the performance of traditional bandwidth-allocation-based mechanism as the baseline for comparison. We modelled this simulation on the basis of NSGA-II which is a modified version of NSGA (Nondominated Sorting Genetic Algorithm) released in 1994 used to optimize multiobjectives [31]. The objective function is illustrated as

$$f = F + \sum w_{D_i} + w_{I_i} + w_{K_i}, \quad i \in 1, 2, 3, 4, 5, 6 \tag{20}$$

Table 9: An example for classifying typed resources in a practical environment.

D_{DIK}	I_{DIK}	K_{DIK}
Light intensity {2, 4, 8, 16 . . . 1024}	2^x(x = [1,10])	Dim light (<2048)
Temperature {25, 25, 25,. . ., 25}	25 (n = 100 s)	Constant temperature (100 s)

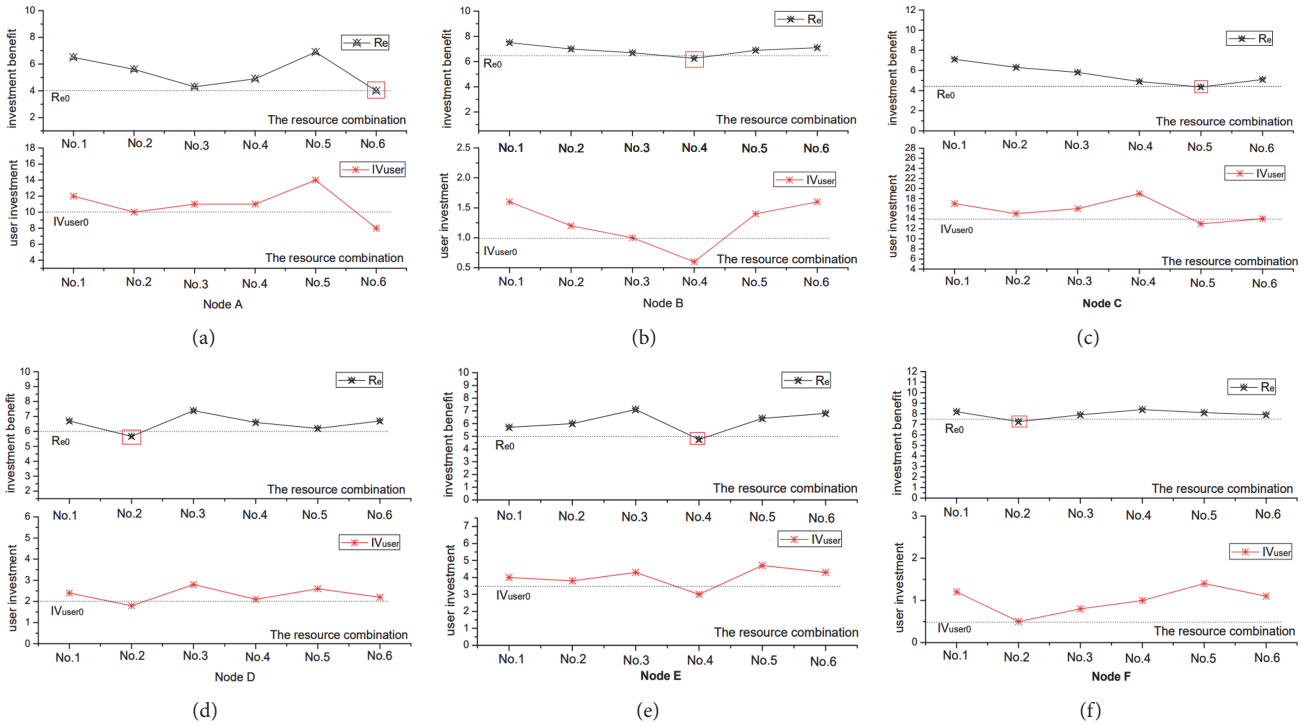


Figure 7: Estimating IV_{user} and R_e values of considered six nodes.

In (20), w_{D_i} is a parameter that contributes to the scale of irs_D. And six restrictive functions are as follows:

$$\sum_{i=1}^{6} w_{D_i} = 1 \tag{21}$$

$$\sum_{i=1}^{6} w_{I_i} = 1 \tag{22}$$

$$\sum_{i=1}^{6} w_{K_i} = 1 \tag{23}$$

$$10 \le \sum_{i=1}^{6} w_{D_i} * irs_D + \sum_{i=1}^{6} w_{K_i} * irs_I + \sum_{i=1}^{6} w_{K_i} * irs_K \le 30 \tag{24}$$

$$\alpha + \beta = 1 \tag{25}$$

$$irs_D + irs_I + irs_K = 30 \tag{26}$$

We calculated user investment benefits of these two mechanisms. Figure 8 shows the R_e values per node at corresponding investment to illustrate benefit difference between these two methods. In the simulation, we assigned values to parameters in the equations, such as α and β. But the actual value of each parameter should be obtained through data learning. According to previous experiment results, we find that investment benefit of our proposed method is smaller compared to conventional method under the same investment. The results show that the proposed method performs better when providing resources processing services.

Table 10: Some network parameters of six links on the wireless network.

Links	A-B	B-C	C-D	D-E	D-F	A-F
B_{ij}	2.5	0.8	3.2	0.9	0.7	1.2
H_i	1.4	3.5	1.1	1.0	0.4	2.3
F_{ij}	1.0	5.0	6.0	2.0	4.0	7.0
l	1.0	2.0	3.0	1.0	2.0	3.0
N	2.0	4.0	1.0	3.0	2.0	1.0
IR_{ban}	0.36	0.46	0.40	0.58	0.51	0.32
FR_{wait}	0.47	0.19	0.31	0.28	0.08	0.61

In order to reflect the performance of processing resources in this fog computing application compared with the conventional method, we analysed F value of a given wireless network. Figure 9 shows F value of two mechanisms

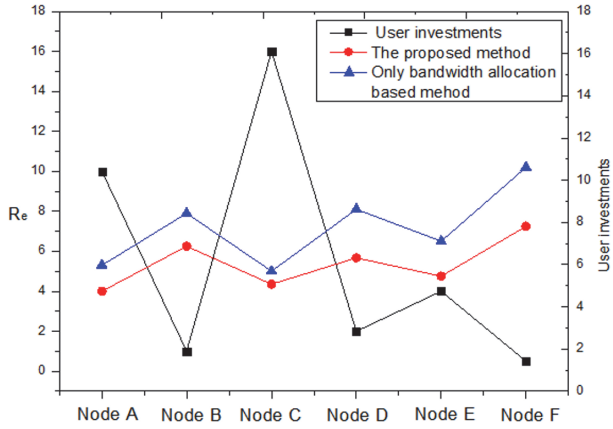


Figure 8: R_e values generated by different methods corresponding to user investment.

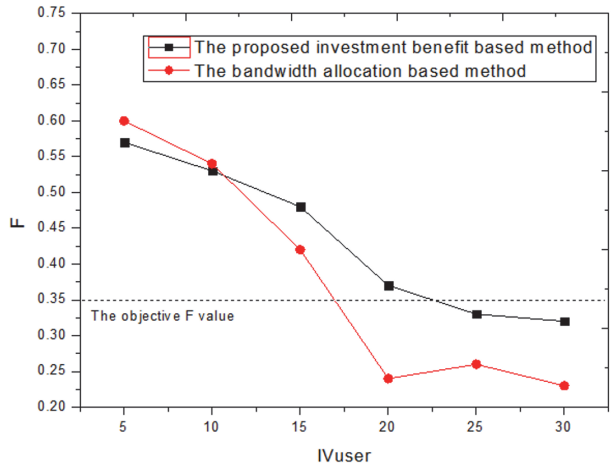


Figure 9: F value of two mechanisms under different investments.

under different investments. The simulation results indicate that the bandwidth-allocation-based method demands fourteen units cost to achieve objective F value. F value of the proposed method is smaller compared with the conventional method.

We used a social network to illustrate the impact of network size on performance of IoT_TOM. The network contains 1133 nodes and 10903 edges, which is from Datatang [32]. We used a sample set of 862 nodes and 1932 edges including five given areas. This simulation was conducted on a Lenovo ThinkPad E431 desktop with a 2.50 GHz Intel core i5 CPU and 8G RAM, running the *pycharm* Edit on Windows 10 operating system. As shown in Figure 10, (6, 12) represents that the number of nodes is 6 and the number of edges is 12. The results show that the bigger the scale of a given wireless network is, the better the distribution of network resources performs in a certain range.

6. Related Work

The dynamic reconstruction of computation and storage resources not only improves the utilization of resources but

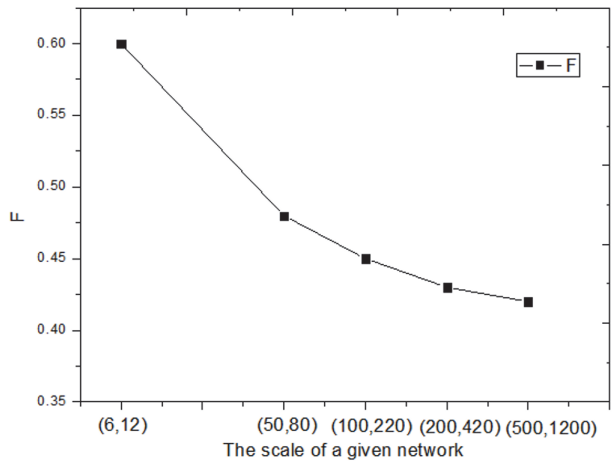


Figure 10: F values of different network scales.

also simplifies management. Some of the workloads that use common resource computing and storage technologies can handle the current cloud system to avoid saturated clouds [33]. However, the cloud computing paradigm is not suitable to solve the problem of unbalanced distributions of network resources in the Internet of Things. We argue that it is necessary to consume bandwidth to transmit resources between IoT nodes, aiming to obtain storage and computation resources of other nodes in order to satisfy user demands. In [34], the authors proposed a distributed multilevel storage (DMLS) model to solve the problems of restricted computation, limited storage, and unstable network. A wireless-based Software Defined Mobile Edge Computing (SDMEC) provided support for autoscaling network storage resource based on the network demand [35]. It is important to dynamically allocate bandwidth in fog computing application. In [36], the authors proposed a bandwidth allocation scheme based on collectable information. An integrated resource allocation method shared the limited bandwidth among multiple wireless body area networks (WBANs) [37]. In [38], the authors studied the tradeoff between the latency and reliability in task offloading to mobile edge computing (MEC). We argue that it is important to achieve global optimization over a wireless network in forms of converting the resource types. The bandwidths of a given wireless network are expensive in fog computing, which makes it a challenge to reduce latency of accessing resources. In [39], the authors proposed an efficient software defined data transmission scheme (ESD-DTS) to maximize data transmission throughput and consistently maintain a flow's latency. T. Taleb et al.'s proposed scheme enforces an autonomic creation of MEC services to allow anywhere-anytime data access with optimum Quality of Experience (QoE) and reduced latency [40]. A hierarchical model composing data, information, knowledge, and wisdom is usually represented in the form of a pyramid [41]. We divide resources in the Internet of Things into data, information, and knowledge. We consider converting resource types to improve user investment benefit in a business value driven manner. We proposed a three-tier resource processing

architecture based on Data Graph, Information Graph, and Knowledge Graph, aiming at optimizing storage, transfer, and processing typed resources.

7. Conclusions and Future Work

Physical services in the Internet of Things are heterogeneous, large-scale, and resource-constrained. Fog computing makes IoT devices more intelligent and saves communication bandwidth. But distributions of network resources in fog computing applications are not balanced. We proposed to convert resource types to deal with this problem. Based on different resource combinations that feature a better user investment benefit, conversions between resource types change the resource scale. In order to satisfy users' general requirements, we consider transmitting typed resources to obtain enough storage and computing resources of other nodes through consuming bandwidth. On the whole, optimized processing of typed resources with storage and computing collaborative adaptation effectively improves the ratio of performance over investment. Meanwhile, processing resources in our proposed three-tier architecture of Data Graph, Information Graph, and Knowledge Graph optimizes spatial and temporal efficiency. The proposed mechanism achieves global optimization over a wireless network to provide cost-effective resource processing services. According to simulation results, users invest minimum costs to obtain maximum performances of processing IoT resources. Generally, our proposed mechanism keeps most fog computing systems in a stable state, which is important for improving the user experience.

In the future, we will continue our study of resources management to improve performance of processing in a business value driven manner and conduct further experiments. We will also explore machine learning approach to optimize processing of resources including data, information, and knowledge.

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 paper is supported by Hainan Key Development Program (no. ZDYF2017128), NSFC (no. 61662021, no. 61363007, and no. 61502294), and the IIOT Innovation and Development Special Foundation of Shanghai (no. 2017-GYHLW-01037).

References

[1] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging IT platforms: vision, hype, and reality for delivering computing as the 5th utility," *Future Generation Computer Systems*, vol. 25, no. 6, pp. 599–616, 2009.

[2] M. H. U. Rehman, V. Chang, A. Batool, and T. Y. Wah, "Big data reduction framework for value creation in sustainable enterprises," *International Journal of Information Management*, vol. 36, no. 6, pp. 917–928, 2016.

[3] M. H. Ur Rehman, P. P. Jayaraman, S. Ur Rehman Malik, A. Ur Rehman Khan, and M. M. Gaber, "RedEdge: A novel architecture for big data processing in mobile edge computing environments," *Journal of Sensor and Actuator Networks*, vol. 6, no. 3, article no. 17, 2017.

[4] 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.

[5] I. Tomkos, L. Kazovsky, and K.-I. Kitayama, "Next-generation optical access networks: Dynamic bandwidth allocation, resource use optimization, and QoS improvements," *IEEE Network*, vol. 26, no. 2, pp. 4–6, 2012.

[6] M. Satyanarayanan, "The emergence of edge computing," *The Computer Journal*, vol. 50, no. 1, pp. 30–39, 2017.

[7] Z. Ma, Q. Zhao, and J. Huang, "Optimizing bandwidth allocation for heterogeneous traffic in IoT," *Peer-to-Peer Networking and Applications*, vol. 10, no. 3, pp. 610–621, 2017.

[8] Y. Ito, H. Koga, and K. Iida, "A bandwidth reallocation scheme to improve fairness and link utilization in data center networks," in *Proceedings of the 13th IEEE International Conference on Pervasive Computing and Communication Workshops, PerCom Workshops 2016*, pp. 1–4, March 2016.

[9] S. K. Sharma and X. Wang, "Live Data Analytics with Collaborative Edge and Cloud Processing in Wireless IoT Networks," *IEEE Access*, vol. 5, pp. 4621–4635, 2017.

[10] K. Staniec and M. Habrych, "Telecommunication platforms for transmitting sensor data over communication networks—state of the art and challenges," *Sensors*, vol. 16, no. 7, article no. 1113, 2016.

[11] X. Wu, S. Zhang, and A. Ozgur, "STAC: Simultaneous Transmitting and Air Computing in Wireless Data Center Networks," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 12, pp. 4024–4034, 2016.

[12] C. Yin and C. Wang, "The perturbed compound Poisson risk process with investment and debit interest," *Methodology and Computing in Applied Probability*, vol. 12, no. 3, pp. 391–413, 2010.

[13] C. Yin and Y. Wen, "An extension of Paulsen-Gjessing's risk model with stochastic return on investments," *Insurance: Mathematics and Economics*, vol. 52, no. 3, pp. 469–476, 2013.

[14] P. Li, M. Zhou, and C. Yin, "Optimal reinsurance with both proportional and fixed costs," *Statistics & Probability Letters*, vol. 106, pp. 134–141, 2015.

[15] D. Hua and L. Zaiming, "Optimal reinsurance with both proportional and fixed costs," *Applied Mathematics-A Journal of Chinese Universities*, vol. 27, no. 2, pp. 150–158, 2012.

[16] Y. Wang and C. Yin, "Approximation for the ruin probabilities in a discrete time risk model with dependent risks," *Statistics & Probability Letters*, vol. 80, no. 17-18, pp. 1335–1342, 2010.

[17] W. Sun and L. Peng, "Observer-based robust adaptive control for uncertain stochastic Hamiltonian systems with state and input delays," *Lithuanian Association of Nonlinear Analysts. Nonlinear Analysis: Modelling and Control*, vol. 19, no. 4, pp. 626–645, 2014.

[18] A. Brogi and S. Forti, "QoS-aware deployment of IoT applications through the fog," *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1185–1192, 2017.

[19] J. Hong and V. O. Li, "Optimal resource allocation for transmitting network information and data in wireless networks," in *Proceedings of the IEEE International Conference on Communications, ICC '10*, pp. 1–5, 2010.

[20] C. Zins, "Conceptual approaches for defining data, information, and knowledge," *Journal of the Association for Information Science and Technology*, vol. 58, no. 4, pp. 479–493, 2007.

[21] Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu, "Learning entity and relation embeddings for knowledge graph completion," in *Proceedings of the in Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, pp. 2181–2187, Austin , Texas, USA, 2015.

[22] J. Pujara, H. Miao, L. Getoor, and W. Cohen, "Knowledge Graph Identification," in *Proceedings of the 12th International Semantic Web Conference, The Semantic Web - ISWC '13*, vol. 8218, pp. 542–557, 2013.

[23] O. Deshpande, D. S. Lamba, M. Tourn et al., "Building, maintaining, and using knowledge bases: A report from the trenches," in *Proceedings of the 2013 ACM SIGMOD Conference on Management of Data, SIGMOD '13*, pp. 1209–1220, 2013.

[24] Y. Duan, L. Shao, G. Hu, Z. Zhou, Q. Zou, and Z. Lin, "Specifying architecture of knowledge graph with data graph, information graph, knowledge graph and wisdom graph," in *Proceedings of the 15th IEEE/ACIS International Conference on Software Engineering Research, Management and Applications, SERA 2017*, pp. 327–332, gbr, June 2017.

[25] L. Shao, Y. Duan, X. Sun, Q. Zou, R. Jing, and J. Lin, "Bidirectional value driven design between economical planning and technical im-plementation based on data graph, information graph and knowledge graph," in *Proceedings of the IEEE International Conference on Software Engineering Research, Management and Applications*, pp. 339–344, 2017.

[26] L. Shao, Y. Duan, X. Sun, H. Gao, D. Zhu, and W. Miao, "Answering Who/When, What, How, Why through Constructing Data Graph, Information Graph, Knowledge Graph and Wisdom Graph," in *Proceedings of the The 29th International Conference on Software Engineering and Knowledge Engineering*, pp. 1–6.

[27] L. Shao, Y. Duan, X. Sun, Q. Zou, R. Jing, and J. Lin, "Bidirectional value driven design between economical planning and technical implementation based on data graph, information graph and knowledge graph," in *Proceedings of the 15th IEEE/ACIS International Conference on Software Engineering Research, Management and Applications, SERA 2017*, pp. 339–344, gbr, June 2017.

[28] D. Kong, L. Liu, and Y. Wu, "Best approximation and fixed-point theorems for discontinuous increasing maps in Banach lattices," *Fixed Point Theory and Applications*, vol. 2014, no. 1, pp. 1–10, 2014.

[29] Y. Dong, L. Song, M. Wang, and Y. Xu, "Combined-penalized likelihood estimations with a diverging number of parameters," *Journal of Applied Statistics*, vol. 41, no. 6, pp. 1274–1285, 2014.

[30] W. Sun, K. Wang, C. Nie, and X. Xie, "Energy-based controller design of stochastic magnetic levitation system," *Mathematical Problems in Engineering*, vol. 2017, no. 3, pp. 1–6, 2017.

[31] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, "A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II," in *Parallel Problem Solving from Nature PPSN VI*, vol. 1917 of *Lecture Notes in Computer Science*, pp. 849–858, Springer, Berlin, Germany, 2000.

[32] Datatang, http://www.datatang.com/data/796.

[33] S. Alonso-Monsalve, F. Garcia-Carballeira, and A. Calderon, "Fog computing through public-resource computing and storage," in *Proceedings of the 2nd International Conference on Fog and Mobile Edge Computing, FMEC 2017*, pp. 81–87, May 2017.

[34] J. Xing, H. Dai, and Z. Yu, "A distributed multi-level model with dynamic replacement for the storage of smart edge computing," *Journal of Systems Architecture*, vol. 83, pp. 1–11, 2018.

[35] J. Al-Badarneh, Y. Jararweh, M. Al-Ayyoub, M. Al-Smadi, and R. Fontes, "Software Defined Storage for cooperative Mobile Edge Computing systems," in *Proceedings of the 4th International Conference on Software Defined Systems, SDS 2017*, pp. 174–179, esp, May 2017.

[36] Y. Ito, H. Koga, and K. Iida, "A bandwidth allocation scheme to meet flow requirements in mobile edge computing," in *Proceedings of the 2017 IEEE 6th International Conference on Cloud Networking (CloudNet)*, pp. 114–5118, September 2017.

[37] D.-R. Chen, "A QoS Bandwidth Allocation Method for Coexistence of Wireless Body Area Networks," in *Proceedings of the 25th Euromicro International Conference on Parallel, Distributed and Network-Based Processing, PDP 2017*, pp. 251–254, rus, March 2017.

[38] 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.

[39] E. Kim and S. Kim, "An Efficient Software Defined Data Transmission Scheme based on Mobile Edge Computing for the Massive IoT Environment," *KSII Transactions on Internet and Information Systems*, vol. 12, no. 2, pp. 974–987, 2018.

[40] 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.

[41] J. Rowley, "The wisdom hierarchy: Representations of the DIKW hierarchy," *Journal of Information Science*, vol. 33, no. 2, pp. 163–180, 2007.

Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 8475604, 11 pages
https://doi.org/10.1155/2018/8475604

Research Article

A Service-Based Method for Multiple Sensor Streams Aggregation in Fog Computing

Zhongmei Zhang,[1,2,3] Chen Liu,[2,3] Shouli Zhang,[1,2,3] Xiaohong Li,[1] and Yanbo Han[2,3]

[1]*School of Computer Science and Technology, Tianjin University, Tianjin 300350, China*
[2]*Beijing Key Laboratory on Integration and Analysis of Large-Scale Stream Data, North China University of Technology, Beijing 100144, China*
[3]*Institute of Data Engineering, School of Computer Science and Technology, North China University of Technology, Beijing 100144, China*

Correspondence should be addressed to Zhongmei Zhang; gloria_z@126.com

Received 23 February 2018; Revised 12 April 2018; Accepted 19 April 2018; Published 28 May 2018

Academic Editor: Xuyun Zhang

A surge in sensor data volume has exposed the shortcomings of cloud computing, particularly the limitation of network transmission capability and centralized computing resources. The dynamic intervention among sensor streams also brings challenges for IoT applications to derive meaningful information from multiple sensor streams. To handle these issues, this paper proposes a service-based method with fog computing paradigm based on our previous service abstraction, which can capture meaningful events from multiple sensor streams. In our service abstraction, we utilize correlation analysis method to capture events as variations of correlation among sensor streams. Facing inconsistent frequency and shift of correlation, we propose a Dynamic Time Warping- (DTW-) based algorithm to obtain sensor streams' lag-correlation. For adaptively aggregating related events from different services, we also propose an event routing algorithm to assist the composition of cascaded events through service collaboration. This paper reports the tryout use of our method in Chinese power grid for detecting abnormal situations of power quality. Through a series of experiments based on real sensor data in power grid, we verified that our method can reduce the network transmission and computing resource with high accuracy.

1. Introduction

With the rapid development of Internet of Things (IoT), numerous sensors are deployed for event monitoring, ecological observation, and so forth, which produce an overwhelming amount of stream data [1]. Individual sensor streams are usually fine-grained, have relatively lower value density, and intervene with each other dynamically. For obtaining more valuable information, many IoT applications try to capture events from sensor streams with multiple sources in different ways [2]. The big real-time sensor streams lead to the emergency of a new computing paradigm, that is, fog computing. Compared with cloud computing, it does not require data to be stored and processed in a centralized way and can greatly lower the computation and storage costs of the cloud server through putting processing of sensor streams to the edge nodes.

Although fog computing has lots of advantages, its related researches have just started and are far from being mature. Most of current researches still focus on the architecture, wireless sensor networks, hardware, and underlying applications [3]. Few works attach enough importance on what software abstractions should be fit to put on a fog/edge node and how to program them. Actually, it is very important for an IoT application to build with the fog computing paradigm.

In an IoT application, sensor streams usually are not independent with each other. They need to cooperate for achieving complicated business goals. For capturing meaningful events from multiple sensor streams, event detection techniques [4–6] are already used in various real-time

enterprise solutions. State-of-the-art event processing models provide frameworks to represent and reason events. However, they require the detailed event definitions and work on fixed set of sensor streams [7]. Facing large amount of sensor streams, it is impractical to define all detailed definitions of each variety of events. Hence, it is crucial to adaptively choose relative stream sources and capture events without relying on predefined definitions.

Correlation analysis methods have traditionally been used to process data streams for classifying and identifying events. In many fields, such as anomaly detection and electronic trading, the variation of correlation among sensor streams can be regarded as one kind of abnormal events, which do not need detailed definitions on raw sensor streams. Meanwhile, for multiple sensor streams, the inconsistency of frequency and shift of correlation will affect the accuracy of correlation analysis and cause wrong identification of events.

A remarkable effort is to encapsulate sensor data into Web services, in which the open Web standards are supported for information sharing [8–10]. But ad hoc association and streaming-oriented queries can hardly be reflected. Following the thought of software-defined sensors, we proposed a service abstraction called ***proactive data service*** in our previous works [11, 12]. With proactive data service, the *sensor events* generated directly by sensors can be transformed into multiple *service events*, which reveal more meaningful information. In this paper, we integrate our service abstraction with fog computing paradigm. In our service abstraction, we utilize the correlation analysis method that works directly on sensor streams for capturing events as the variation of correlations. Facing the inconsistency of frequency and shift of correlation of different sensor streams, we propose a Dynamic Time Warping- (DTW-) based algorithm to obtain the lag-correlation and capture dynamic events.

With proactive data services, the sensor events generated directly by sensors can be transformed into multiple service events that reveal more meaningful information. Since the capturing of some complicated events depends heavily on the cooperation of different sensor streams, we propose an event routing algorithm for indicating the targets of certain service events and define a set of declarative rules in our service abstraction for triggering corresponding processing. Based on changeable event routing paths and declarative rules, proactive data services can adaptively and proactively collaborate with each other and generate more meaningful events from dynamic sensor streams.

Proactive data services can be deployed in edge nodes, do some preliminary analyses, and report events to cloud servers for further analyses. Relying on our DTW-based event capturing and event routing algorithms, we can capture events from multiple sensor streams and obtain complicated events through service collaboration. We preliminarily applied our method in a real-world scenario: power quality event detection for Chinese power grid; and we verified the feasibility of our method. Based on real sensor data from Chinese power grid, we evaluate the effectiveness and efficiency of our method through a series of experiments. The experimental results show that our method can greatly reduce the transformation of primitive sensor streams and improve the efficiency of event capturing with high accuracy.

2. Scenario

The Chinese power grid consists of various electric devices with complicated electrical connections. In most electric devices of Chinese power grid, a series of sensors are deployed to monitor indicators of power quality, such as voltage, current, and frequency. Due to the relationship of electric device and disturbance source, the devices can be divided into two types: disturbance source device and general device. A disturbance source device is deployed close to certain disturbance source and is directly influenced, while a general device is indirectly influenced by the disturbance source. Each sensor is in charge of monitoring one indicator and produces corresponding sensor stream. These sensor streams are affected by various disturbance sources, such as wind farm, photovoltaic power station, and electrified railway and show some abnormal status; that is, originally correlated sensor streams become uncorrelated or out of limit. A key problem of a power grid is to capture abnormal events of power quality based on these sensor streams.

To users, the primitive sensor streams are somewhat meaningless and unreadable. It is crucial to increase the value density of sensor streams, that is, to capture abnormal events from primitive sensor streams. An abnormal power quality event caused by certain disturbance generally involves multiple devices, and the sensor streams in each device show the same abnormal status (i.e., involving the same set of sensor streams). Figure 1 shows an example of abnormal power quality events, in which each column indicates a sensor stream, and the darker columns indicate abnormal and light columns indicate normal. As shown in Figure 1, the event involves multiple sensor streams from one disturbance source device and 3 general devices, and for different devices, the sensor streams that indicate the same indicators are involved. For single device, the details of involved sensor streams are shown in Figure 2.

As shown in Figure 2, the three phase voltages ($stream_1$, $stream_2$, and $stream_3$) in power quality are generally correlated, while two trains are passing by an electrified railway; the correlation among these three streams is changed, and an abnormal event "unbalanced three phase voltages" occurs in the electric device. For each device, it is challenging to capture events from multiple sensor streams, because the correlation among them may be lagged. Figure 2 also shows some examples of lag-correlation; the trends of rising and falling of $stream_4$ and $stream_5$ are similar but not synchronous strictly. And the frequency and timestamps of data records in different sensor streams are different. For example, $stream_2$ and $stream_4$ have lower frequency than other streams. There could be error correlation analysis if we do not consider these issues.

It is obvious that the sensors in each device and the devices involved in abnormal power quality events are changeable because of the influence of various disturbance sources. The sources of sensor streams for generating

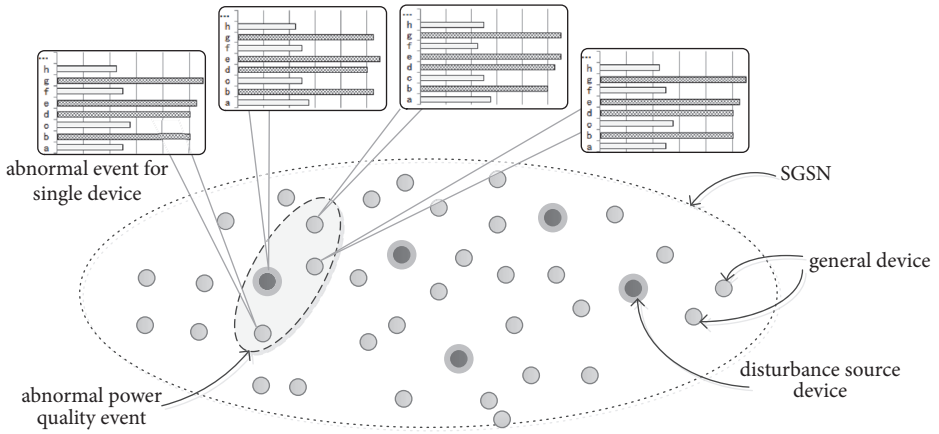


Figure 1: An example of abnormal power quality event.

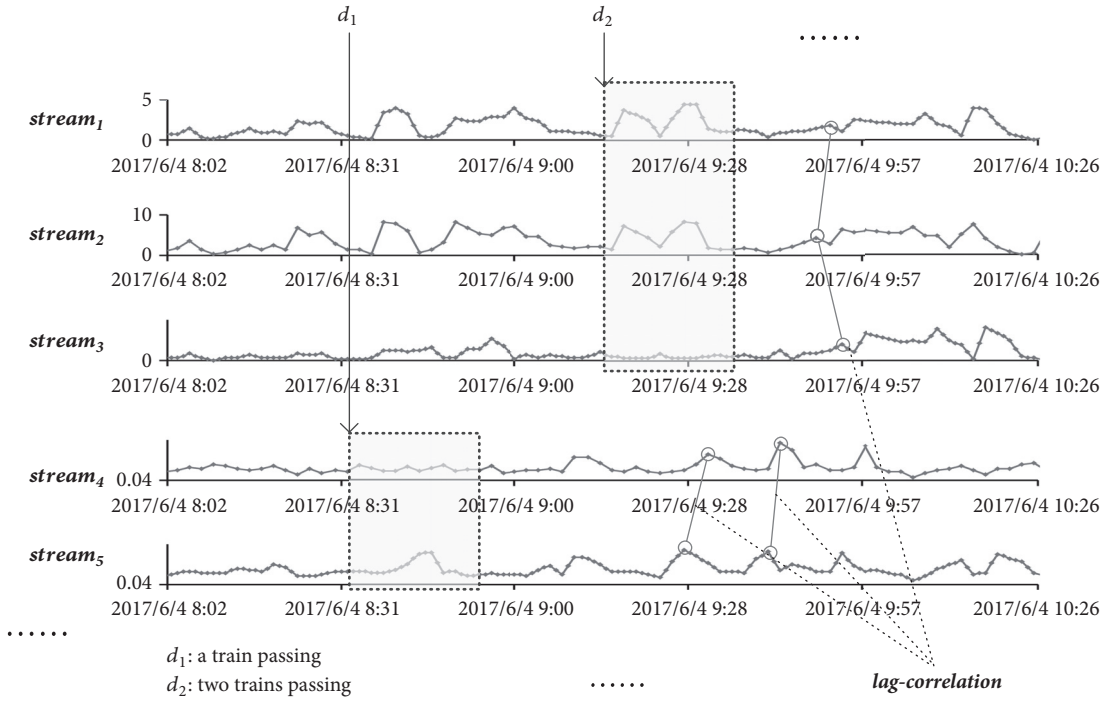


Figure 2: Examples of disturbances' affection in sensor streams.

meaningful events are uncertain. Hence, for deriving meaningful event, we need to adaptively choose related sensor stream sources, that is, sensors in single device and electric devices, and capture abnormal events for each device and aggregate abnormal events of multiple devices for obtaining final events.

Because of uncertain stream sources of events, existing methods need to collect and analyze all sensor streams in cloud server for generating events. The analysis of all sensor streams will cause vast and unnecessary consumption of computational and communication resources and cannot guarantee the requirement of timeliness. In this paper, we aim to utilize the fog computing paradigm and deal with two problems: how to capture events more accurately through considering the characters of sensor streams and how to adaptively choose relative sensor streams relying on different disturbances.

3. Overview of Our Service-Based Method

Figure 3 shows the rationale of our service-based method for dynamically aggregating multiple sensor streams and capturing meaningful events. Based on our previous proactive data service model, we can create services whose basic function is to generate events (regarded as service events in the proactive data service model) from certain set of primitive sensor events. Each proactive data service can be deployed in edge nodes and do preliminary processing for

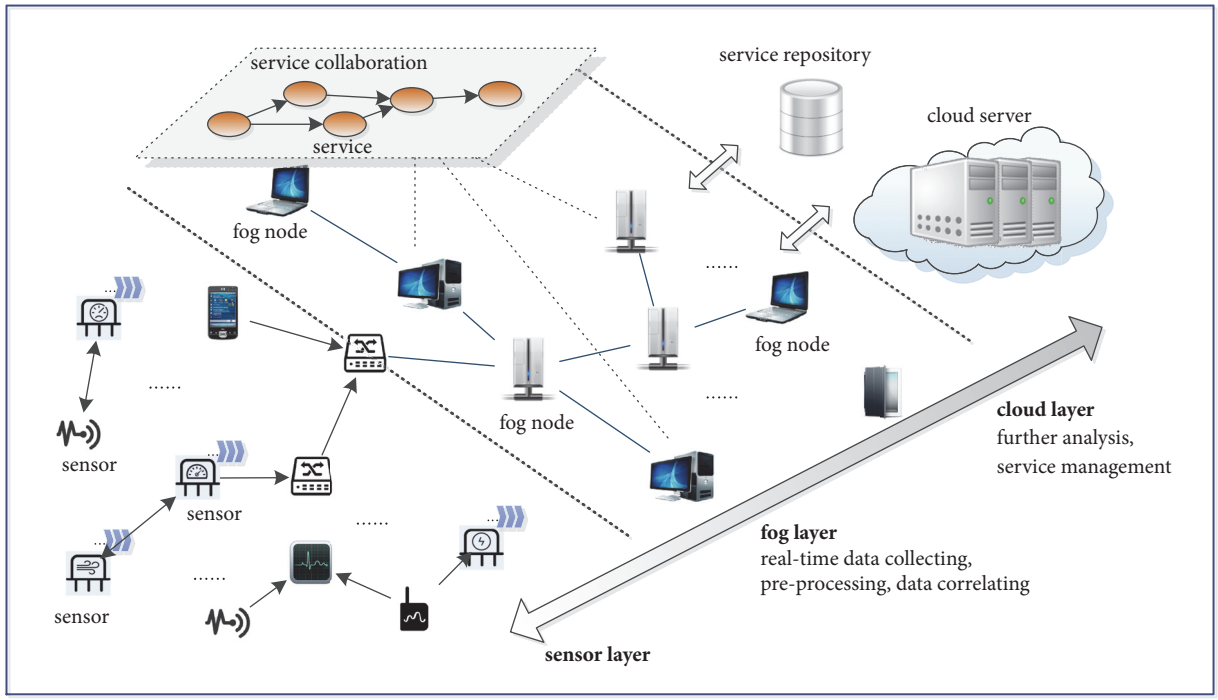


Figure 3: Rationale of our service-based method.

increasing value density of sensor streams. In this case, only meaningful events are transmitted to cloud server. For generating more meaningful and complicated events involving dynamic sources, the proactive data service also supports routing its service events to related services and adaptively collaborates with other services. Based on proactive data services, the fog layer can provide abilities such as real-time data accessing, preprocessing, and sensor data correlating. The detailed information of services can be stored in cloud service for service discovering.

A sensor stream can be defined as follows.

Definition 1 (sensor stream). A sensor stream can be represented as $ssd = \langle source_{id}, A, R\rangle$, in which $source_{id}$ is the id of stream source, A are the attribute sets, and $R = \{r_1, r_2, \ldots, r_i, \ldots\}$ is an infinite series of data record $r = \langle\{\langle a_i, v_i\rangle \mid a_i \in A\}, t\rangle$which is a set of key-value pairs with a timestamp.

Typically, events are of a certain type, have a timestamp, and hold specific data. Many event detection technologies utilize event patterns on sensor streams to represent events. For example, a set of predicates are utilized to describe the event pattern in [13]. In this paper, we regard the correlation's variations among multiple sensor streams as events. Hence, we can utilize a set of sensor sources to represent a primitive event pattern. And a more complicated event can be composited by multiple primitive events and represented as a composite event pattern $p_{com} = \langle\{source_{id}\}, R\rangle$, in which $\{source_{id}\}$ indicates the sensor stream sources in one primitive event pattern and R is a set of relations between the primitive event patterns. For example, an abnormal event "unbalanced three phase voltages" occurring in single device can be represented by three streams indicating three phase voltages, and an integrated abnormal power quality event can be composited by several "unbalanced three phase voltages" events in multiple devices.

We regard each record with its source and timestamp in sensor stream as a sensor event and abstract the ability of event processing as a proactive data service, which can access and process multiple sensor streams and generate service events with higher value. The services can communicate with each other and provide service events to cloud server based on their *uris*. Figure 4 shows the structure of our proactive data service model.

Definition 2 (proactive data service). A proactive data service can be formalized as $pds = \langle pds_{id}, event_{in}, event_{out}, DR, EP, hyperlinks\rangle$, in which pds_{id} is the unique identifier, $event_{in}$ represents the input event streams, $event_{out}$ represents the output event streams generated by EP, DR are the declarative rules, EP means the event processing function, and the *hyperlinks* are an optional parameter that indicates the targets of service events.

Presently, we utilize Pearson's correlation coefficient (PCC) [14] to measure the sensor streams' correlations for capturing the change of correlation as event. The correlation among sensor data usually shifts in time, which can be regarded as lag-correlation [15]. We consider the lag-correlation analysis problem as to find the time lag vector to maximize the PCC and proposed a DTW-based event capturing algorithm in our service.

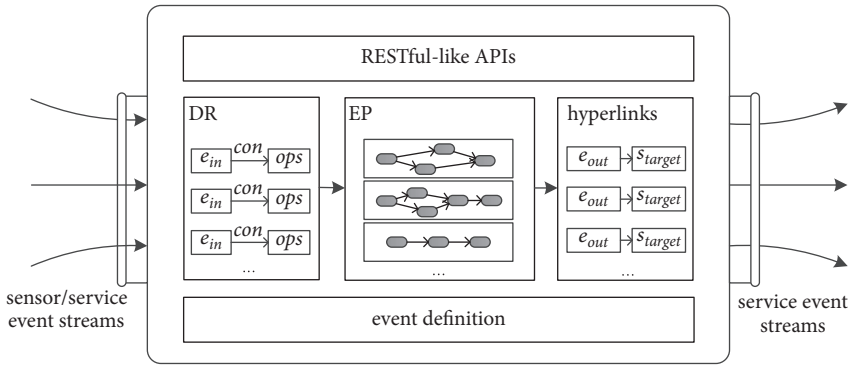


Figure 4: The structure of the proactive data service.

For supporting dynamic service collaboration, we propose an event routing algorithm for setting the hyperlinks in service, which can indicate the target services for certain service event. And we also utilize declarative rules in service to handle external events. In this paper, we define declarative rules based on Event-Condition-Action (ECA) rules, which are originally used in active database systems to provide event-driven, instantaneous responses for conventional database systems [16]. In this way, the entire process requires no intervention from users or external applications.

4. Event Capturing in Proactive Data Service

The proactive data service is created based on fixed set of sensor streams. We calculate the correlations between each two sensor streams based on time-based slide window, and once any two sensor streams' correlation is changed, an event is through to be captured. The pattern of captured event can be represented by the set of sensor streams whose correlation is changed.

When analyzing correlation among sensor data, misjudgment of correlation will also occur if correlation lags are not considered. The lag-correlation analysis problem can be considered to find the time lag vector to maximize the PCC, which can be formalized as follows.

Definition 3 (lag-correlation analysis problem). Given two sensor data sets E_i and E_j, suppose that $E_i = \{e_{i\_t1}, e_{i\_t2}, \ldots, e_{i\_tn}\}$ and $Ej = \{e_{j\_t1}, e_{j\_t2}, \ldots, e_{j\_tn}\}$ in a slide window; if there exists a time lag vector $\Delta = \{t_1', t_2', \ldots, t_n'\}$, one has

$$\mathrm{MAX}\left(cor_{(E_i,E_j,\Delta)}\right) = \frac{\sum_1^n \left(e_{i\_t} - \overline{e_i}\right)\left(e_{j\_(t+tk')} - \overline{e_j}\right)}{\sqrt{\sum_1^n \left(e_{i\_t} - \overline{e_i}\right)^2} \times \sqrt{\left(e_{j\_(t+tk')} - \overline{e_j}\right)^2}} \quad (1)$$

To analyze the correlation between two sensors' data, we can firstly obtain a time lag vector that makes $\mathrm{MAX}(cor_{E_i,E_j,\Delta})$ and then justify the lag-correlation based on the calculated coefficient. In this paper, we calculate the time lag vector making the minimum Euclidian distance of the normalized series and then calculate the PCC of the original series based on the time lag vector.

DTW algorithm [17] is a robust method used to measure similarity of time series, which can shift and distort the time series to ignore the problem of time axis scaling and shifting. In this paper, we adopt a DTW-based algorithm to find the time lag vector of two sensor data series. To ignore the problem of amplitude scaling, we align sensor data series with DTW algorithm based on normalized sensor data series. To avoid excessive time warping, we set a maximum value for the time lag vector.

Because of the inconsistent frequency of different sensor streams, the numbers of their data records in the same window may be different, and their PCC cannot be calculated directly. Hence, we firstly utilize linear interpolation method to increase the data records of the sensor stream with lower frequency. Then we utilize DTW algorithm to align the data series and calculate the correlations. The pseudocode of our event capturing algorithm executed in each window is shown in Algorithm 1 .

As shown in Algorithm 1, after choosing sources of sensor streams, we can obtain a set of data series. We firstly get all pairs of data series for calculating the correlation between each pair (line 1). For the data series pair that has different numbers of records, we utilize interpolation method to increase the number of data records for the shorter one (line 7). In this way, these two data series could have the same number of data records. Then we utilize DTW algorithm to align the data series for analyzing the lag-correlation (lines 12 and 13). At last, we verify the difference of current correlation coefficient and previous coefficient; if the difference is bigger than given threshold, an event will be captured and the execution in this window is ended.

5. Dynamic Collaboration among Services

With our proactive data service model, we can create one proactive data service for each electric device. Based on our event capturing algorithm, we can capture the abnormal status of each device. In this section, we will introduce the dynamic service collaboration for correlating sensor streams

Input:
$\boldsymbol{E} = \{E_1, E_2, \ldots E_k\}$: a set of data series in the same window;
δ_{cha}: the change threshold of PCC;
limit: the maximum value for the time lag vector;
output:
event;
(1) obtain all pairs $\{\langle E_i, E_i\rangle\}$in $\boldsymbol{E}$;
(2) for each $\langle E_i, E_i\rangle$
(3) if E_i and E_j have different numbers of records
(4) $d \leftarrow$ the difference of their records' numbers;
(5) $E_m \leftarrow$ getLongerSeries(E_i, E_j);
(6) $E_n \leftarrow$ getShorterSeries(E_i, E_j);
(7) $E_n' \leftarrow$ interpolation(E_n, d);
(8) else
(9) $E_m \leftarrow E_i$;
(10) $E_n' \leftarrow E_j$;
(11) end if
(12) $\langle E_m', E_n''\rangle \leftarrow$ aligned data series from $\langle E_m, E_n'\rangle$ based DTW
(13) calculate correlation coefficient *cor* of $\langle E_m', E_n''\rangle$;
(14) if $|cor\text{-}cor_{previous}| >= \delta_{cha}$
(15) an *event* is captured;
(16) break;
(17) end if
(18) end for

Algorithm 1: Event capturing algorithm.

from dynamic devices and obtaining the abnormal power quality events.

Each collaboration procedure is beginning with a proactive data service corresponding to one disturbance source device. When a service event is generated, it will be routed to related services and triggers corresponding processing of events. Since the collaboration is emanative from disturbance source service, we set up one event detection service (ED service) for each disturbance source for collecting the collaboration results and obtaining the final abnormal power quality events.

5.1. The Event Routing Algorithm. Since there is no electrical connection information, we propose an algorithm through considering service correlation relying on both the location of physical devices and existing collaboration results. The algorithm has two phases. In the initial phase, we regard all proactive data services near to given one as its related services based on physical devices' location. In the runtime phase, we keep updating the correlation of proactive data services based on their collaboration results for certain event pattern.

Specifically, we regard the neighbour proactive data services as the total set of routing targets $S_{possible}$ for given service. We assume that devices frequently cooccurring in power quality events are very likely to be involved in an event again. Hence, we will keep routing events to the successful collaborated set S_{valid} and reduce the possibility to route events to the failure collaborated set $S_{invalid}$. Because of the dynamic nature of events, devices that did not cooccur in one power quality event still have possibility to form a power quality event. Hence, the remaining service set $S_{alternative}$ in $S_{possible}$ excepting the actual target set $S_{practical}$ also needs to be considered.

We set up weight ω_i for each proactive data service s_i in $S_{possible}$ and obtain the actual target set $S_{practical}$ through comparing ω_i with a given threshold δ. In procedure 1, we firstly set up the weight of all proactive data services in $S_{possible}$ as δ and regard $S_{possible}$ as actual targets $S_{practical}$ in default. Any events generated by a disturbance source service will be sent to their related services. Algorithm 2 shows the pseudocode of the second phase in our algorithm.

In procedure 2, we update the weights of target proactive data services for certain service and certain event pattern based on newly received collaboration result, and because the target pattern is the same as source pattern, we then can obtain the declarative rules. After each event routing, we increase target proactive data services' weights of valid routing paths and reduce the weights of invalid paths. Meanwhile, we also increase the weight of services in $S_{alternative}$ for considering their possibility. Through comparing the new weights with given threshold, we will obtain new $S_{practical}$ and update the event handlers and links in the proactive data service.

5.2. Case Study. Taking the detection procedure of an abnormal power quality event caused by *wind farm* as an example, this section describes our service-based method. Figure 5 shows the service collaboration procedure. Due to limited space, only parts of proactive data services are shown. We create one proactive data service for each electric device and

```
(1)  while receiving collaborate result
(2)    extract source service s_i;
(3)    if the result is a composed event
(4)      extract the valid routing path ⟨pattern, s_ij⟩;
(5)      s_ij.weight = s_ij.weight+ α;
(6)    else
(7)      extract the invalid routing path ⟨pattern, s_ij⟩;
(8)      s_ij.weight = s_ij.weight- β;
(9)      if s_ij.weight< δ
(10)       remove s_ij from s_ipractical;
(11)     end if
(12)   end if
(13)   for each s_ij in s_alternative
(14)     s_ij.weight = s_ij.weight+γ;
(15)     if s_ij.weight ≥ δ
(16)       add s_ij in s_ipractical;
(17)     end if
(18)   end for
(19) end while
```

Algorithm 2: Updating the routing targets.

deploy it in edge node. The event detection service can be deployed in cloud server to collect the collaboration results. Proactive data service A is corresponding with a disturbance source device, whose event patterns, output events, and routing paths can be set up at design time. Other proactive data services are corresponding with general devices, and their pattern needs to be matched and output events and routing paths are updated at runtime.

As mentioned before, each proactive data service can capture abnormal events for single device, which can be represented by a set of sensor streams, while an integrated abnormal power quality event can be captured by multiple services, in which each service indicates one device. Table 1 shows the details of some proactive data services in the collaboration procedure.

The collaboration is beginning with proactive data service A; when an event e_1, that is, the correlation among $\{a, b, c\}$, is generated, A sends it to service B, C, and the ED service. When B and C receive this event, they will calculate the correlation among $\{a, b, c\}$, and events generated by B and Cwill be composed with event received from A to generate more meaningful events $\langle\{A, B\}, \{a, b, c\}\rangle$ and $\langle\{A, C\}, \{a, b, c\}\rangle$. The new events will be continuously routed. Since F does not generate new event after receiving event from C, it sends a failure message to ED service and stops collaborating with other proactive data services. Event generated by B is sent to D and E, and only D generates new event and sends it to G and H. Since both G and H fail to collaborate with D, the whole collaboration is finished, and a power quality event involving proactive data services A, B, C, D is generated.

At each step of the collaboration, the ED service collects collaboration results and updates corresponding routing paths. For proactive data service A and event e_1, routing paths $e_1 \rightarrow \{B, C\}$ are valid, so the weights of targets will be increased and the routing paths can be kept. For service B and event e_2, the routing path $e_2 \rightarrow E$ is invalid, so the weight of E will be decreased and the routing path may be removed. Besides, there may be proactive data services that are not in the routing paths, while they are neighbours of certain proactive data service. Their weight will also be increased, so they can be considered for detecting dynamic events.

Table 1: The details of proactive data services in our scenario.

ID	*Pattern*	$event_{out}$	*Routing paths*	*Set-up time*
A	$\{a, b, c\}$	e_1: $\langle A, \{a, b, c\}\rangle$	$e_1 \rightarrow \{B, C\}$	*Design time*
B	$\{a, b, c\}$	e_2:$\langle\{A, B\}, \{a, b, c\}\rangle$	$e_2 \rightarrow \{D, E\}$	*Runtime*
C	$\{a, b, c\}$	e_3:$\langle\{A, C\}, \{a, b, c\}\rangle$	$e_3 \rightarrow \{F\}$	*Runtime*
D	$\{a, b, c\}$	e_4:$\langle\{A, B, D\}, \{a, b, c\}\rangle$	$e_4 \rightarrow \{G, H\}$	*Runtime*

6. Evaluation

In this section, we evaluate the effectiveness and efficiency of our proactive data services with our event capturing and routing algorithms. We built one proactive data service for each electric device and evaluated the average performance of the services.

6.1. Experiment Setup

Data Set. The data set used in our experiments is from real sensor data in Chinese power grid. We selected sensor data from 2017-05-01 to 2017-06-05. There are 2653 devices with 438 disturbance source devices. Each disturbance source device can cause 2–6 types of events for single device, and there are totally 1978 kinds of event patterns. And each electric device at most deploys 2051 sensors. We simulated

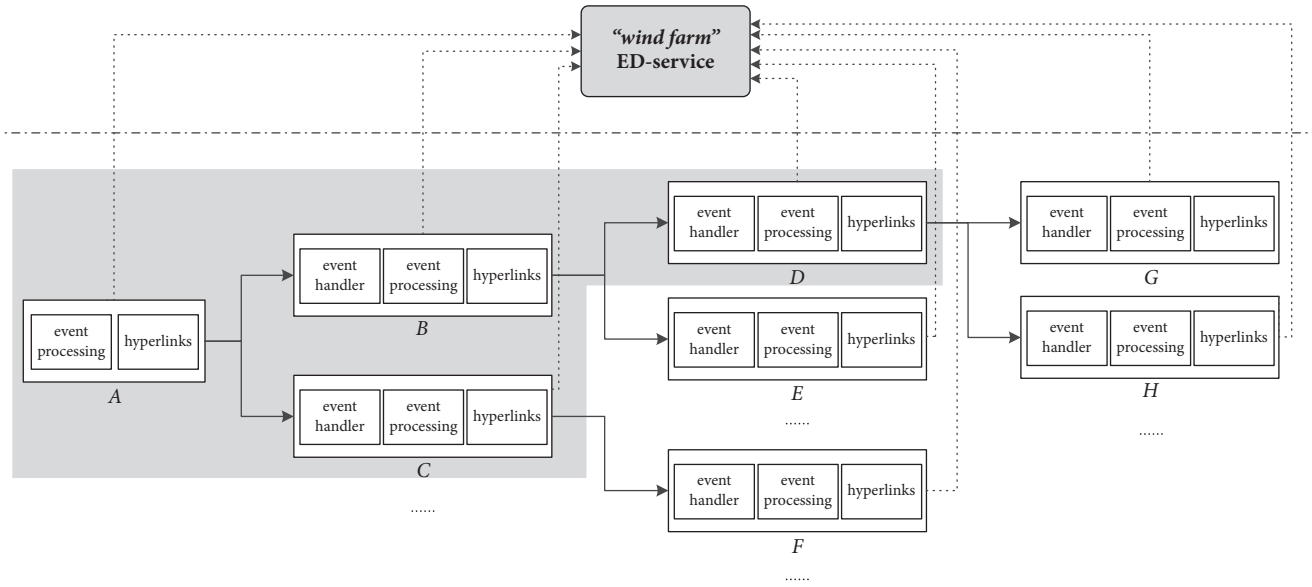


Figure 5: Collaboration procedure of abnormal power quality event detection.

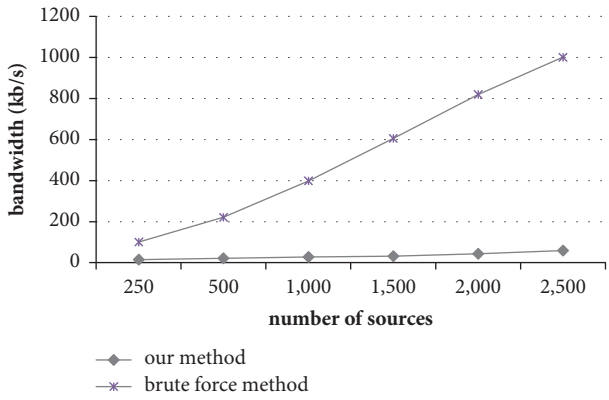


Figure 6: The bandwidth consumptions of different methods.

sensor stream for each sensor strictly according to the real timestamp.

Environment. We implemented our method in virtual machines with CentOS release 6.4, four Intel Core i5-2400 3.10 GHz CPUs and 4.00 GB RAM. All the algorithms are implemented in Java with JDK 1.8.0.

Evaluation Criteria. We design the following criteria:

(1) Bandwidth consumption: the bandwidth consumption means the data volume received by cloud server from stream sources

(2) System load: the system load of the proactive data service can be divided into the load of CPU and memory

(3) Event execution time: for each generated event, its execution time is the difference between the time when its corresponding sensor event is firstly received and the time when it is generated

(4) Accuracy: the accuracy includes precision and recall, which can be calculated as follows:

$$precision = \frac{|D \cap T|}{|D|} \times 100\%$$
$$recall = \frac{|D \cap T|}{|T|} \times 100\% \quad (2)$$

where $D = \{d_1, d_2, \ldots, d_m\}$ is detected event list and $T = \{t_1, t_2, \ldots, t_n\}$ is the actual event list.

6.2. Experiment Result and Analysis. Existing works barely focused on capturing events from fixed set of sensor streams and merely considered dynamic sensor streams correlation. At present, power quality detection system has been built to detect abnormal power quality events. Because the sources involved in power quality events are changeable, the existing method is kind of brute force method, which needs to collect and process all sensor streams in cloud server with a centralized manner. We utilize brute force method practically used as comparative and compare our method with different sensor stream sources, that is, device sets in power grid.

Figure 6 shows the bandwidth consumptions of different methods with different source sets. The brute force method needs to process all sensor streams, while in our method the disturbance source sensor stream only needs to match with corresponding patterns, and the processing of general sensor stream is relying on the processing results of previous service. In this case, only sensor streams from related sources are processed. Hence, the bandwidth consumption of our method is far less than existing brute force method. Because more events may occur with more sensor streams, the bandwidth consumption of our service abstraction has a slight increase with the growth of stream sources.

Figures 7 and 8 show the system loads and execution times of different methods with different stream sources. It

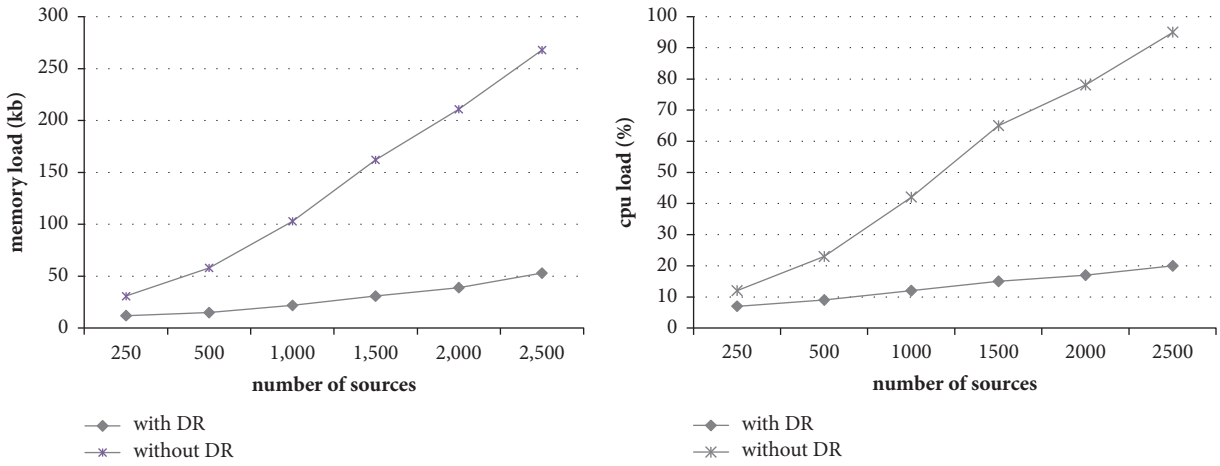


Figure 7: The system loads of different methods.

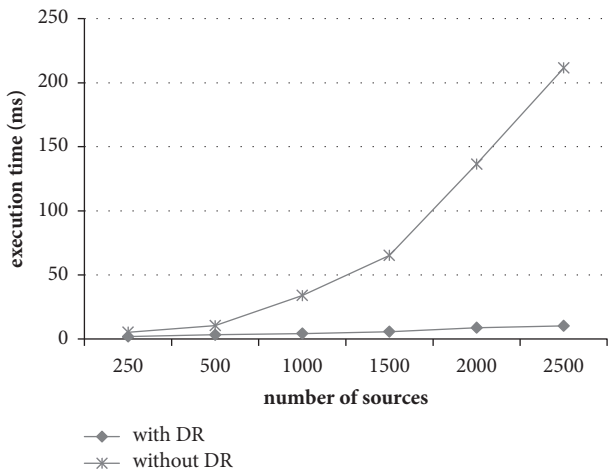


Figure 8: The execution times of different methods.

is obvious that our method has much lower system loads and less execution time, because the senor data that need to be processed are much less.

Then we evaluate our event capturing algorithm through calculating the precision and recall of the events capturing by our DTW-based algorithm and normal correlation analysis algorithm, which does not consider the inconsistency of frequency and shift of correlation among multiple sensor streams. For verifying the accuracy of our method, we analyzed all the sensor data and consulted domain expert and obtained 1783 abnormal events as the actual event list.

Figure 9 shows the average precision and recall of different event capturing algorithms with different stream sources. It is obvious that our event capturing algorithm has much higher precision and recall. This is because our event capturing algorithm adopted DTW-based correlation analysis algorithm and considers different frequency of sensor streams and the lag of correlation. As shown in Figure 9, our method has an average precision over 89% and average recall over 91%. By consulting domain experts, existing event capturing method in power grid has the precision and recall of about 60%–75%, which indicates that our method can increase the accuracy by about 20%.

In future, we aim to update the hyperlinks and declarative rules at runtime based on real-time analysis and learning and further improve the accuracy through considering the dynamic and intelligent correlations among events.

7. Related Works

Fog computing aims at bringing back partial computation load from the cloud to the edge devices. Gupta et al. [18] proposed a system, which abstracted connected entities as services and allowed applications to orchestrate these services with end-to-end QoS requirements. Researchers presented Vigil, a real-time distributed wireless surveillance system supporting real-time tracking and surveillance in enterprise campuses, retail stores, and across smart cities. Vigil utilized laptops as fog computing nodes between cameras and cloud to save wireless capacity [19]. Yuriyama and Kushida [20] virtualized a physical sensor as a virtual sensor on the cloud and automatically provided the virtual sensors on demand. Mohamed et al. [21] abstracted services and components involved in smart city applications as services accessible through the service-oriented model and enhanced integration and allow for flexible inclusion and utilization of the various services needed in a smart city application. In our research, we borrow ideas from Vigil [19] and utilize laptops as fog computing nodes. Each computing node can maintain multiple proactive data services and transmit generated event streams to the cloud server.

Many works utilized various event patterns to abstract events based on spatiotemporal relation of sensor data and transformed event detection problem into a pattern matching problem. A basic method is to detect event based on preset threshold value defined by domain expert [4]. To describe more complex event, Xue et al. [5] integrated the pattern-based approach with an in-network sensor query processing framework, which focused on the event patterns for single sensor and defined 5 common basic patterns to describe events. Xue et al. [22] also defined a spatiotemporal pattern of

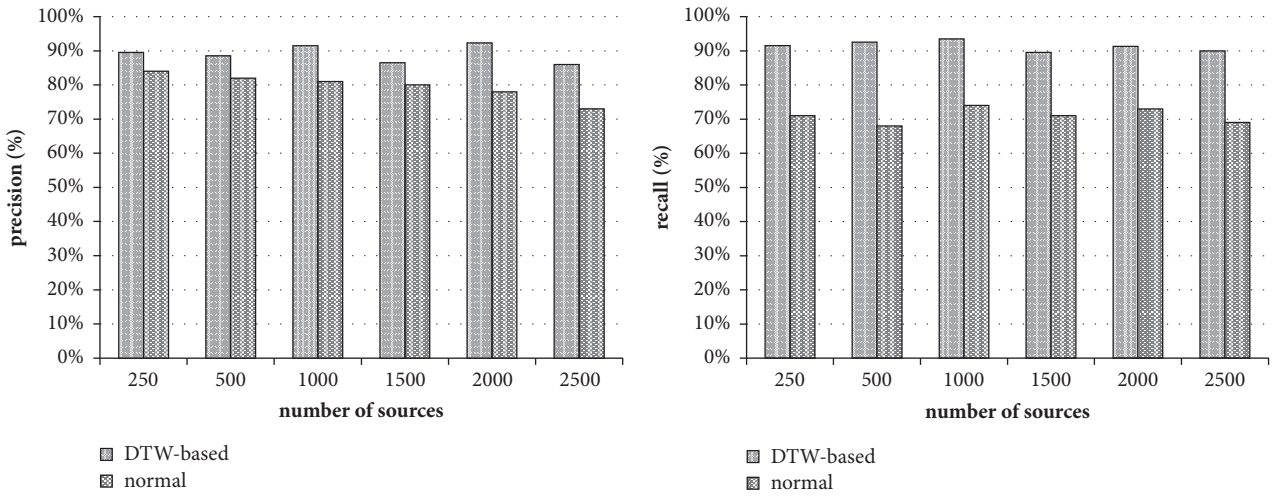


Figure 9: The accuracy of different event capturing algorithms.

events, which described events with a set of regression models over multiple spatial regions. Mao et al. [23] utilized Bayesian Network and Markov Chain to model the spatiotemporal correlation among sensors. However, the above pattern-based methods are mostly based on predefined event patterns which indicated precise sources of sensor streams, and the learning-based methods also need fixed inputs for model training and classification. Hence, they are inappropriate for extracting meaningful information from dynamic sensor streams.

One trend to promote the development of IoT is viewing IoT as Web of Thing (WoT), where the open Web standards are supported for information sharing and device interoperation. Paganelli et al. [10] proposed a framework that supported developers to model smart things as Web resources and exposed them through RESTful APIs. Facing large amount of Web service in IoT age, Qi et al. [24] proposed a novel privacy-preserving service recommendation approach based on locality-sensitive hashing to handle the distributed recommendation problem. DSWare [25] was a data-centric service middleware that defined a compound event specification, which consists of maximum detection range, time interval, and a confidence function. Cheng et al. [26] proposed a situational-aware service coordination method, which included a situational event definition language, a situational event detection algorithm, and an event-driven service coordination behaviour model based on ECA mechanism. Most existing service-based methods needed to predefine the collaboration goals for service composition or collaboration and did not support adaptive collaboration. Hence, they cannot be applied to increase value density for dynamic sensor streams. In this paper, we refer to existing methods to encapsulate sensor data as services and propose a service-based method to support adaptive aggregation of sensor streams.

8. Conclusions

With more and more sensors deployed in physical world, an overwhelming amount of stream data is produced. It is meaningful to set up suitable abstraction to increase value density and to promote widespread use for business applications and data visualization. The paper proposes a service-based method with fog computing diagram for adaptive aggregation of multiple sensor streams. In this paper, we can create a proactive data service by generating service events from fixed sensor streams defined in design time and capture events based on correlation analysis method. Facing the inconsistency of frequency and shift of correlation of different sensor streams, we utilize a Dynamic Time Warping- (DTW-) based algorithm to obtain the lag-correlation and capture dynamic events. For cooperating dynamic sensor streams and capturing more complicated events, we utilize an event routing algorithm for indicating the targets of service events and define a set of declarative rules in our service abstraction for triggering corresponding processing.

To verify our method's feasibility, we applied it for power quality event detection in Chinese power grid. In this scenario, we have to deal with dynamic sensor stream sources. We set up event detection services that can be deployed in cloud server to collect the collaboration results by obtaining finial abnormal events and assisting the adaptive collaboration of proactive data services. Based on the real sensor data set in Chinese power grid, a series of experiments demonstrate that our method have much higher efficiency than existing methods with high accuracy.

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 there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (no. 61672042), Models and Methodology of Data Services Facilitating Dynamic Correlation of Big Stream Data; Beijing Natural Science Foundation (no. 4172018), Building Stream Data Services for Spatiotemporal Pattern Discovery in Cloud Computing Environment; and the Program for Youth Backbone Individual, supported by Beijing Municipal Party Committee Organization Department, Research of Instant Fusion of Multisource and Large-Scale Sensor Data.

References

[1] J. Heidemann, M. Stojanovic, and M. Zorzi, "Underwater sensor networks: applications, advances and challenges," *Philosophical Transactions of the Royal Society A: Mathematical, Physical & Engineering Sciences*, vol. 370, no. 1958, pp. 158–175, 2012.

[2] 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.

[3] 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.

[4] K. Kapitanova, S. H. Son, and K. Kang, "Using fuzzy logic for robust event detection in wireless sensor networks," *Ad Hoc Networks*, vol. 10, no. 4, pp. 709–722, 2012.

[5] W. Xue, Q. Luo, and H. Wu, "Pattern-based event detection in sensor networks," *Distributed Parallel Databases*, vol. 30, no. 1, pp. 27–62, 2011.

[6] Y. Singh, S. Saha, U. Chugh, and C. Gupta, "Distributed event detection in wireless sensor networks for forest fires," in *Proceedings of the 15th International Conference on Computer Modelling and Simulation*, pp. 634–639, Cambridge, UK, April 2013.

[7] O. P. Patri, A. V. Panangadan, V. S. Sorathia, and V. K. Prasanna, "Sensors to events: semantic modeling and recognition of events from data streams," *International Journal of Semantic Computing*, vol. 10, no. 4, pp. 461–501, 2016.

[8] D. Guinard and V. Trifa, "Towards the web of things: web mashups for embedded devices," in *Proceedings of the International World Wide Web Conferences*, pp. 1506–1518, 2009.

[9] D. Zeng, S. Guo, and Z. Cheng, "The web of things: a survey," *Journal of Communications*, vol. 6, no. 6, pp. 424–438, 2011.

[10] F. Paganelli, S. Turchi, and D. Giuli, "A web of things framework for RESTful applications and its experimentation in a smart city," *IEEE Systems Journal*, vol. 10, no. 4, pp. 1412–1423, 2016.

[11] Y. Han, G. Wang, J. Yu, C. Liu, Z. Zhang, and M. Zhu, "A service-based approach to traffic sensor data integration and analysis to support community-wide green commute in China," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 9, pp. 2648–2657, 2016.

[12] Y. Han, C. Liu, and S. Su, "A decentralized and service-based apporach to proactively correlating stream data," in *Proceedings of the International Conference on Internet of Things*, pp. 93–100, 2016.

[13] H. Wu, J. Cao, and X. Fan, "Dynamic collaborative in-network event detection in wireless sensor networks," *Telecommunication Systems*, vol. 62, no. 1, pp. 43–58, 2016.

[14] T. Guo, S. Sathe, and K. Aberer, "Fast distributed correlation discovery over streaming time-series data," in *Proceedings of the 24th ACM International Conference on Information and Knowledge Management*, pp. 1161–1170, October 2015.

[15] S. Wu, H. Lin, W. Wang et al., "RLC: ranking lag correlations with flexible sliding windows in data streams," *Pattern Analysis & Applications*, vol. 20, no. 2, pp. 601–611, 2016.

[16] A. Paschke, "ECA-RuleML: an approach combining ECA rules with temporal interval-based KR event/action logics and transactional update logics," *Computer Science*, 2006.

[17] R. J. Kate, "Using dynamic time warping distances as features for improved time series classification," *Data Mining and Knowledge Discovery*, vol. 30, no. 2, pp. 283–312, 2016.

[18] H. Gupta, S. B. Nath, S. Chakraborty, and S. S. K. Ghosh, "SDFog: a software defined computing architecture for qos aware service orchestration over edge devices," https://arxiv.org/abs/1609.01190.

[19] T. Zhang, A. Chowdhery, P. Bahl, K. Jamieson, and S. Banerjee, "The design and implementation of a wireless video surveillance system," in *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*, pp. 426–438, September 2015.

[20] M. Yuriyama and T. Kushida, "Sensor-cloud infrastructure—physical sensor management with virtualized sensors on cloud computing," in *Proceedings of the 13th International Conference on Network-Based Information Systems (NBiS '10)*, pp. 1–8, September 2010.

[21] N. Mohamed, J. Al-Jaroodi, I. Jawhar, S. Lazarova-Molnar, and S. Mahmoud, "SmartCityWare: A service-oriented middleware for cloud and fog enabled smart city services," *IEEE Access*, vol. 5, pp. 17576–17588, 2017.

[22] W. Xue, Q. Luo, and H. K. Pung, "Modeling and detecting events for sensor networks," *Information Fusion*, vol. 12, no. 3, pp. 176–186, 2011.

[23] Y. Mao, X. Chen, and Z. Xu, "Real-time event detection with water sensor networks using a spatio-temporal model," in *Database Systems for Advanced Applications*, pp. 194–208, 2016.

[24] L. Qi, H. Xiang, W. Dou, C. Yang, Y. Qin, and X. Zhang, "Privacy-preserving distributed service recommendation based on locality-sensitive hashing," in *Proceedings of the IEEE International Conference on Web Services*, pp. 49–56, Honolulu, Hawaii, USA, June 2017.

[25] S. Li, Y. Lin, S. H. Son, J. A. Stankovic, and Y. Wei, "Event detection services using data service middleware in distributed sensor networks," *Telecommunication Systems*, vol. 26, no. 2-4, pp. 351–368, 2004.

[26] B. Cheng, D. Zhu, S. Zhao, and J. Chen, "Situation-aware IoT service coordination using the event-driven SOA paradigm," *IEEE Transactions on Network and Service Management*, vol. 13, no. 2, pp. 349–361, 2016.

Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 2940952, 11 pages
https://doi.org/10.1155/2018/2940952

Research Article

RePage: A Novel Over-Air Reprogramming Approach Based on Paging Mechanism Applied in Fog Computing

Jiefan Qiu, Sai Li, and Bin Cao

College of Computer Science, Zhejiang University of Technology, Hangzhou, China

Correspondence should be addressed to Bin Cao; bincao@zjut.edu.cn

Received 26 January 2018; Accepted 25 March 2018; Published 14 May 2018

Academic Editor: Xuyun Zhang

In fog computing, fog nodes running different tasks near the sources of data are required. Limited to on-board resource, fog node finds it hard to execute multiple tasks and needs over-air reprogramming to rearrange them. With respect to reprogramming, energy efficiency is one of the key issues for over-air reprogramming. Most of traditional reprogramming approaches focus on the energy efficiency during data transmission within network. However, program rebuilding on fog node is, as another significant energy cost, caused by writing/reading local high-power memory. We present a novel incremental reprogramming approach, RePage, in three stages. Firstly, we design a function paging mechanism that makes similar functions to one function page and caches them in low-power volatile memory to save energy. Secondly, we design new cache replacement algorithm for function page considering both modification times and range on the page. At last, further reducing writing/reading operations, we also redesign function invocation manner by centralized managing function addresses. Experiment results show that RePage reduces the sum of reading/writing operations on volatile memory by 89.1% and 92.5% compared to EasiCache and Tiny Module-link, and its hit rate is improved by 10.4% to Least Recently Used (LRU) algorithm.

1. Introduction

In some applications of traditional front-end networks such as wireless sensor networks or ad hoc networks, the users are familiar with the deployment environment after a period of data collection and computing by nodes. More effective functionality modules can be developed and installed on the node by over-air reprogramming to optimize the local software.

Moreover, with emergence of fog computing, more and more systems of fog computing paradigm witness success in many different fields. But the most existing fog computing systems are similar to the traditional sensor networks, which are to solve the specific problem or meet the specific requirements. In the construction of such systems, developer adopts different hardware devices, software module, and even programming language. On the other hand, fog computing systems require dealing with date by fog nodes which are nearby data source. It is inevitable that resource-limited fog nodes execute multiple tasks; for example, the nodes deployed in building need to monitor sound wave and detect vibration damage; the limited storage results in them being unable to achieve the above tasks at the same time; and it is wasteful and inefficient to deploy two kinds of nodes for each task. This necessitates frequently modifying executing application or rearranging tasks through over-air reprogramming.

In most cases, the resource-limited fog nodes are powered by battery. Frequently, over-air reprogramming will shorten node's life and becomes the main bottleneck of large-scale application. The energy cost of over-air reprogramming mainly comes from communication cost when transmitting codes and from rebuilding cost when reading and writing the memory cell. Now, most of research works about reprogramming focus on how to reduce the communication cost. For example, incremental reprogramming approaches [1–8] only need to transmit the differences between executing old program image and the new one with less transmission cost than entire new image.

However, most of incremental reprogramming approaches need to rebuild the new image in the local memory of sensor node. The popular sensor node TelosB [9], for instance, owns internal flash + RAM within the MSP430

Table 1: Energy cost (uJ) of r/w operation in different memories with 1 kB for TelosB node [9].

Operation	Average cost
Read external flash	1015
Read program Flash	785
Read RAM	<50
Write external flash	2458
Write program flash	1850
Write RAM	126

series MCU and independent on-board external flash. When rebuilding code, the different codes and update-relevant instructions, which are packaged into *delta* script [3], need to be firstly read in the external flash. Then, executing update-relevant instructions, the different codes are combined with old program image to generate new image. At last, write new program image into internal flash in MCU. As shown in Table 1, read and write (r/w for short) flash has higher energy cost than RAM. Actually, inserting some instructions may bring about sharp increase of rebuilding cost, because each instruction is contiguously stored, and to avoid covering useful origin one, most parts of instructions may be moved in new address by r/w flash and make room for the inserted instructions. That possibly makes rebuilding cost actually close to the communication cost.

With respect to the fog computing applications, several users possibly access deployed node at the same time. Due to existing large-capacity on-board external flash, several programs can be prestored in this flash and switched according to the needs of different users in updating node's software. Also, incremental reprogramming approach makes only storing difference codes in the flash to save memory space. In such cases, the reprogramming cost will largely depend on rebuilding cost.

In our previous work, we have proposed an incremental reprogramming approach named EasiCache [10] for sensor nodes reprogramming. In this approach, frequently changing codes are stored and executed in low-power and limited-capacity volatile memory such as RAM without the participation of nonvolatile memory. EasiCache caches part of codes on the function as unit, and function's size is uncertain. Updating too frequently causes storage fragmentation and wastes storage space. Therefore, we propose a novel incremental reprogramming approach named RePage based on function paging mechanism applied in fog nodes. The core idea of this approach is to organize functions in the form of function page to solve the decline problem of cache usage and improve the hit rate. We study this approach from the following three aspects.

First of all, the uncertain function size leads to the waste of storage space. In RePage, we analyze the invocation relationship among functions and define function similarity degree (FSD). And then, based on FSD, we design one novel function paging mechanism, which makes similar functions being put into one function page and updates them at the same time.

Secondly, we improve the cache usage and hit rate through a novel cache replacement algorithm considering modification times and range of each function page. This replacement algorithm is able to adaptively adjust the impact from modification times and range.

Thirdly, in order to improve the similarity between new and original program images, we redesign function invocation method according to the characteristics of function page. We use the register relative addressing instead of direct addressing and manage the function address at centralized manner. By this, once function pages are moved, the runtime system will not modify each function's entry address.

Finally, in order to verify the effect of the paging mechanism, we conduct a series of experiments. In experiments, the resource-limited fog nodes are continuously updated by reprogramming without redeployment, and we observed the performance of RePage from hit rage, energy cost, and storage cost.

The rest of this paper is structured as follows: Section 2 shows the current problem in caching codes; Section 3 introduces the design and implementation of RePage; Section 4 describes experimental scenarios and evaluates our approach; Section 5 gives related work and Section 6 gives conclusion.

2. Motivation

In our previous study called EasiCache [10], the code segments are cached and executed in the type of function. When old function is replaced, runtime system for EasiCache distributes memory space to store new function. However, after multiple reprogramming processes, it leads to a storage fragmentation problem and eventually decreases cache performance. In order to describe the storage fragmentation problem, we define the fragmentation degree as follows:

$$\text{fragmentation degree} = \frac{\text{free space of cache}}{\text{space of cache}}. \quad (1)$$

We test EasiCache in randomly updating scenario on TelosB node (details shown in Section 4.1). TelosB equips MSP430 series MCU with 10 KB RAM and 48 KB internal flash [9]. The size of cache is set to 8 KB. Figure 1 shows the EasiCache continuous updating cases with 25 times and their fragmentation degrees.

We can clearly see that the cache fragments began to increase after more than 5 times reprogramming. After 23 times, the fragmentation degree has been more than 30%. This means that new functions are hard to be cached in RAM. The only way to eliminate fragments is to restore all of caching functions in new addresses and that inevitably brings about huge r/w operations on flash.

In addition, in the real process of caching codes, the replaced functions will be saved in the original position of internal flash. Actually, EasiCache has learned from the previous literature [11]. If the function size increases during caching, the size-increased function is put into free space of the main program (.text segment) directly to avoid tuning the other functions' position. However, the internal flash capacity is also tight. After several updates, the free space may run out,

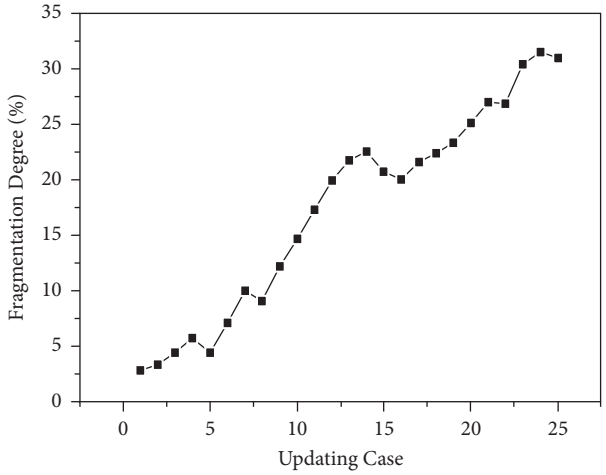


Figure 1: Fragmentation degree in continuously updating cases.

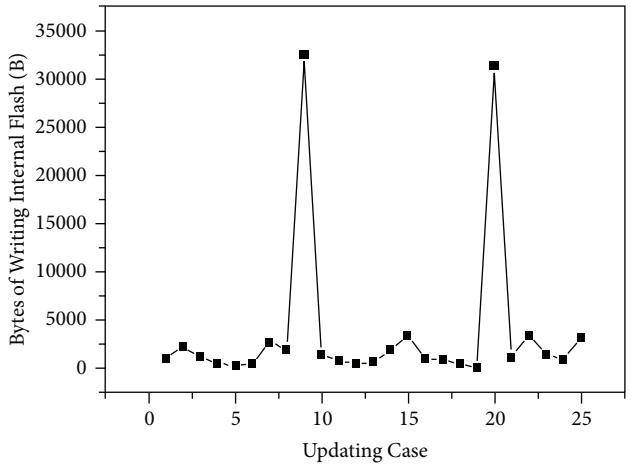


Figure 2: Bytes of writing internal flash in continuously updating cases.

and all functions still need to be rearranged. Figure 2 shows bytes of r/w internal flash when updating node software in a continuous manner. In most of continuous updating cases, the writing operation in internal flash is less, because only the replaced functions need to be written back to internal flash, but if free space is lacking (e.g., 9th and 20th updating cases), the r/w operations on internal flash must increase sharply.

To solve this storage fragmentation problem, we put forward a novel incremental reprogramming approach named RePage. This approach employs the paging mechanism to avoid internal flash space running out and reduce the r/w operations on flash by improving the utilization of cache.

3. Design and Implementation of RePage

In this section, we will discuss the paging mechanism based on function similarity degree and a novel function invocation method. The former mainly solves the storage fragmentation problem, while the latter reduces the r/w operation caused by caching and replacing page. At the same time, we also introduce a cache replacement algorithm that will improve cache hit rate.

3.1. Function Paging Mechanism Based on Function Similarity Degree. The reprogramming mainly adjusts some parameters or functionalities of program. Functions are still relatively independent, but the functionality adjustment tends to modify multiple relative functions. We define function similarity degree (FSD) to quantify such relationship among modified functions and have the function paging mechanism based on FSD. It will aggregate several similar functions into one function page and make each function page size-fixed.

By function paging, the program images will be equally divided into multiple function pages. Meanwhile, in order to ensure that each function page (fp_x) can be incremental and changeable flexibly, we add slop region (E_x) in each tail of each function page fp_x after consulting Koshy and Pandey's approach [12] shown in Figure 3. Slop region is specifically used for the location adjustment for a size-increased function.

Through a lot of reprogramming experiments, we found that the modification of function is usually unidirectional, which means that caller function will be changed with high possibility after any callee function changed; the opposite situation happens less. For example, if one function u's entry address has changed, all call instructions that call function u must be modified. It is clear that two functions have more common caller functions with more possibility being modified at the same time. Therefore, such unidirectional modification can describe the FSD between functions.

RePage figures the FSD, adopting the concept of collaborative filtering applied in recommendation system. Given functions u and v, make $N(u)$ and $N(v)$, respectively, be the callee function in u and v. The FSD between u and v can be described by Jaccard formula:

$$s_{uv} = \frac{|N(u) \cap N(v)|}{|N(u) \cup N(v)|}. \tag{2}$$

In fact, many functions do not have common caller function; namely, $|N(u) \cap N(v) = 0|$. Thus, we can firstly figure the callee function (u, v) of $|N(u) \cap N(v) \neq 0|$ for reducing the computation overhead.

After obtaining each S_{uv} between two functions, we calculate the whole function similarity degree (S) of function page with different weight as follows:

$$S(f_1, f_2, \ldots, f_i) = \sum_{u=1}^{i} \sum_{v=1}^{i} \beta_{uv} s_{uv}$$
$$\beta_{uv} = \frac{1}{\text{size}(u) * \text{size}(v)}, \tag{3}$$

where β_{uv} is weighted value, which is inversely proportional to the size of functions u and v. It ensures that if u and v own smaller size and have higher FSD, the whole degree of function page will be higher; otherwise, in the case of higher degree of functions u and v inasmuch as larger size, the similarity degree of current function page should be reduced.

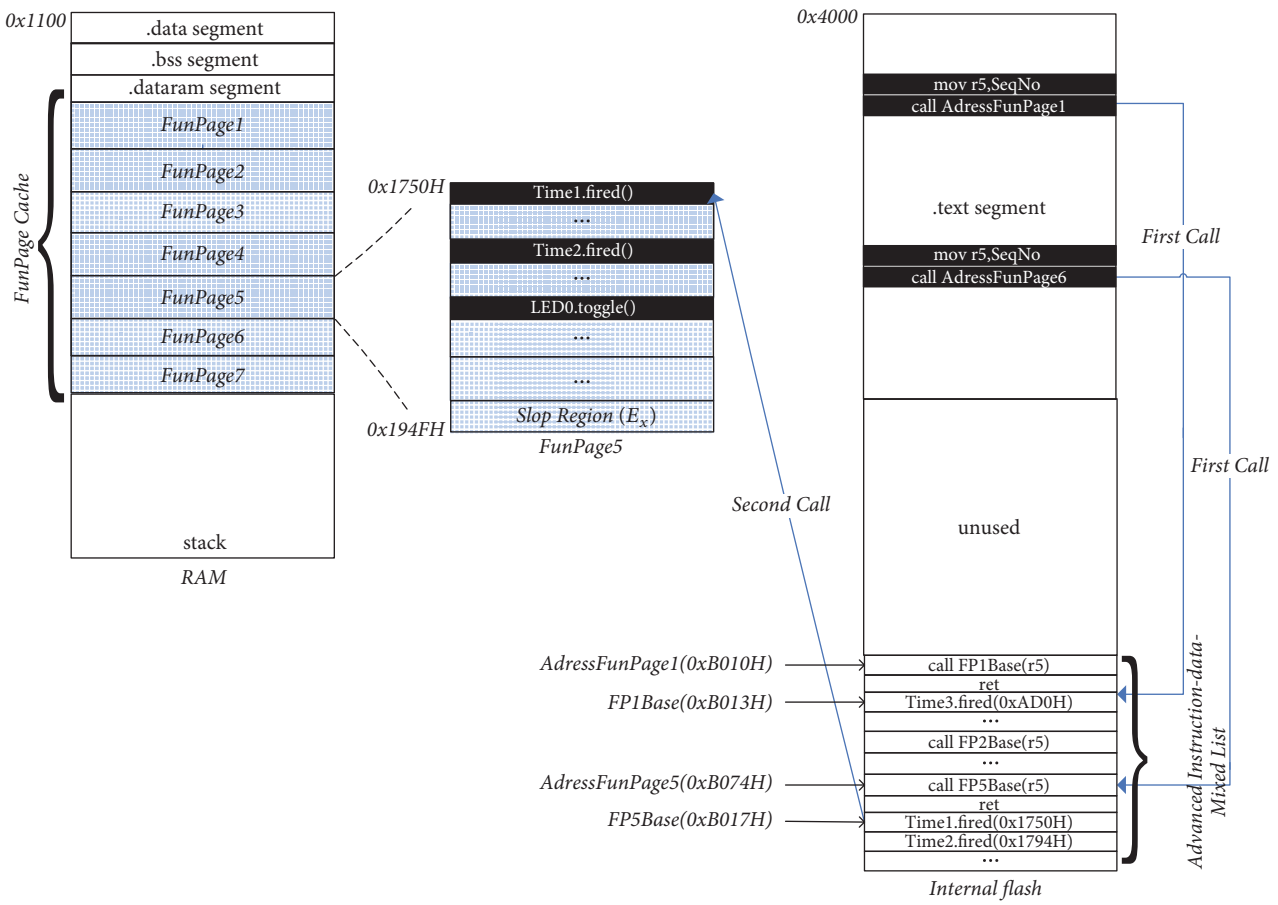


FIGURE 3: Memory layout and two-step calling process: for instance, function Time1.fired contained has address 0x1250 and is cached in page FunPage1. Once one call instruction needs to invocate Timer.fired, it will jump to AddressFunPagex saved in IDML and then obtain real entry address of Timer.fired (0xB013H) by register relative addressing. After obtaining real entry address, the invocation procedure is completed.

The aim of paging mechanism is to put more and more similar functions in one function page. So, we convert paging problem into an optimization problem shown in the following formula:

$$\begin{aligned} \max \quad & \sum_{k=1}^{K} S_k\left(f_i, f_{i+1}, \ldots, f_j\right) \quad i, j \in [1, F] \\ \text{s.t.} \quad & \mathrm{sz}\left(fp_x\right) = \mathrm{sz}\left(f_i\right) + \cdots + \mathrm{sz}\left(f_j\right) + \mathrm{sz}\left(E_x\right) \\ & \sum_{x=1}^{K} \mathrm{sz}\left(fp_x\right) < s_{\text{memory}}. \end{aligned} \tag{4}$$

The optimization objective is to make the similarity degree of function pages maximization. S_k is the similarity degree of function page k, which includes several functions. Parameter F is the number of functions within program image; K is the number of function pages which is much relative to page partition. $\mathrm{sz}(fp_x)$ is the size of each function page, which is up to the sum of sizes of similarity functions $(f_i, f_{i+1}, \ldots, f_j)$ and slop region (E_x). Due to slop region existing, it will make the size of paged program image bigger than the original one. Therefore, it needs to add the restricted condition to make the paged program image able to be accommodated by internal flash whose size is $\mathrm{sz}_{\text{memory}}$.

3.2. The Cache Replacement Algorithm of Function Page. The RAM is always rare, especially within MCU; thus, only small part of function pages can be cached, and most of function pages were still stored and executed in program. Therefore, a cache replacement algorithm is needed to choose which pages should be cached. Existing cache replacement algorithms such as least recently used (LRU) mainly focus on improving the execution efficiency, and RePage's algorithm focuses on reducing the r/w flash operations.

For example, a function page is replaced from cache, according to the LRU, due to not being modified recently. However, once this function page is modified, if a number of codes are changed, the page should stay in cache. Therefore, except the modification time, there is the other factor to decide whether page is being cached or not.

In terms of the hit rate of function page, we propose a recent modification range (RMR) algorithm that considers the modification times and range of each function page at the same time.

Table 2: Advanced instruction-data-mixed list.

Quantity of fun	Update times	AddressFunPageX		Function entry address		
		Call inst.	Ret inst.			
5	1	call FP1base(R5)	ret	fun1	fun2	⋯
10	3	call FP2Base(R5)	ret	fun12	fun23	⋯
7	8	call FP3Base(R5)	ret	Fun27	Fun9	⋯

With RMR, we define $\mathrm{RF}_{(i,m)}$, the replacement factor of function pages i after M times updating, as follows:

$$\mathrm{RF}_{(i,M)} = N_i^{(1-a)} * \left(\frac{\sum_{m=1}^{M} r_{(i,m)}}{M * \mathrm{sz}\left(fp_i\right)} \right)^a \quad a \in (0,1). \tag{5}$$

In formula (5), N_i represents modification times of function page i, and $r_{(i,m)}$ is modification range in page i and during mth updating. Rapidly increasing $r_{(i,m)}$ means that more codes changed within page i. Weight a is limited in range $[1, 0]$ and is used to adjust the impact of N_i and $r_{(i,m)}$ on $\mathrm{RF}_{(i,M)}$. In fact, if modification ranges tend to be the same in each updated function page, the replacement factor $\mathrm{RF}_{(i,M)}$ is largely up to modification times; otherwise, if modification ranges are enormously different with respect to different function pages, it is reasonable to increase impact of N_i, which will make large-modification-range pages be cached with high probability. Therefore, we define the weight a as relative to variance of $r_{(i,M)}$ as follows:

$$a = e^{-1/\sqrt{\sum_{i=1}^{n}(R - r_{(i,M)})^2/n}}. \tag{6}$$

In Mth updating, given n function pages that need to be updated, R is average of $r_{(i,M)}$ of n function pages that need to be updated. Clearly, if variance of $r_{(i,M)}$ is large, a tends to 1; otherwise, a tends to 0.

3.3. The Call and Load of Function. In our previous research EasiCache, function entry address must be changed because of being cached and replaced. If we modify every call instruction, it will cause a mass of r/w operations. To deal with that, we use a centralized management to save all entry address functions in a mixture call list and invocate the functions by two-step calling process.

In RePage, similar functions are put into one function page, and then all entry addresses must change when the function page is cached or replaced. For this, advanced instruction-data-mixed list (IDML) is given to manage the function address shown in Table 2.

Every item in the list focuses on one function page; the first two elements are the quantity of functions and the updating times of current function page. There is a call instruction applying register relative addressing and a return instruction within AddressFunPage$x$. The last column of item contains real function entry address. Once functions are put into certain function page, a sequence of functions in function page is fixed. And the real function entry address in IDML is saved in terms of the sequence.

When calling a function, two-step calling is needed. In the first calling, instruction pointer (IP) skips to the AddressFunPage rather than the real functions. Figure 3 shows the invocation procedure. Before entering the function, the sequence number (SeqNo) of the function in the function page should be assigned to a certain register, for instance, using register r5. Register r5 is taken as index register and used to continuously visit all functions in function page. In second calling, by register relative addressing, register r5 combining with base address (such as FP1base) points to the memory unit that stores the real entry address.

When a function page has been cached in the RAM or replaced for restore in internal flash, the function entry addresses must be changed. We can modify the entry address in the IDML to avoid modifying every instruction in functions. According to the difference between new address of function page and the original one, it is easy to count each new function entry address and write it in IDML.

Due to the register relative addressing, RePage needs to add a register assignment instruction before calling a function, shown in Figure 3. This adding process is transparent to user, but it may cause a register using conflict. We solved this problem by adding the instruction during the precompiling stage. We used C and assembly hybrid programming and directly put the instruction written by assembly-language in C-language source program. By this, the complier can automatically allocate register to avoid the register using conflict.

4. The Experimental Results and Analysis

To verify the function page mechanism, we need to test RePage performance in a continuous updating scenario. Continuously updating scenario is relative to the single updating scenarios, and it means using reprogramming approach to continuously update programs without redeploying node. Now most experiments of reprogramming approaches do not involve caching mechanism; thus, they usually took the single updating experiment for testing. Meanwhile, we mainly focus on continuous updating in experiment.

4.1. Experimental Scenario Introduction. The experiments take TinyOS operating system and TelosB node as the hardware and software platform comprehensively. TelosB nodes are equipped with CC2420, a 2.4 GHz ZigBee RF transceiver, and a 16-bit MCU MSP430F1611. With respect to continuous updating, we design two scenarios.

In the first updating set scenario, the experimental objects are standard routines program named Oscilloscope included

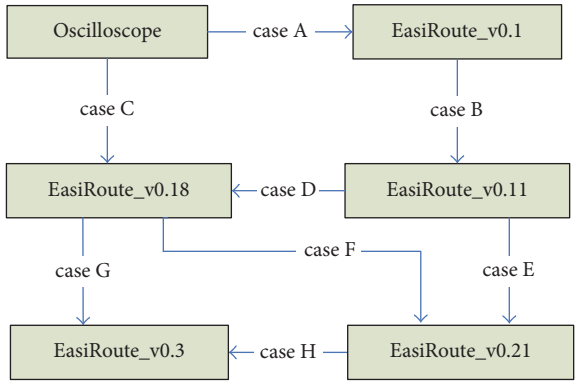


Figure 4: Roadmap of continuous updating.

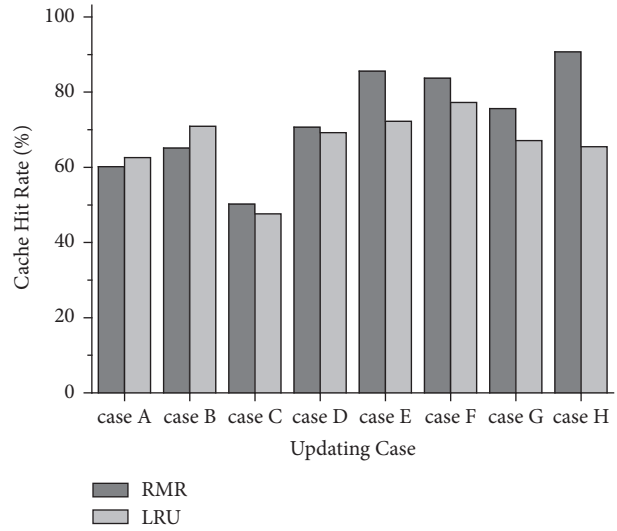


Figure 5: Cache hit rates of updating scenario according to alphabetical order.

in TinyOS application library and five versions of program named EasiRoute (versions from v0.1 to v0.3). These five programs are designed for EasiNet system that has been deployed in the Forbidden Palace Museum and applied for monitoring temperature and humidity [13]. Each version means a major upgrade. As shown in Figure 4, six programs are continuously updated according to alphabetical order.

Updating Case A. Oscilloscope is a standard data collection and transmission procedures. By adding routing functionality, we update it to EasiRoute_v0. 1.

Updating Case B. Update museum monitoring procedures from v0.1 to v0.11. After this update, a sensing-data storing functionality is added, and it realizes storing data in external flash chip.

Updating Case C. Directly update Oscilloscope to v0.11 and add routing and storing functionalities in resource-limited fog node.

Updating Case D. Update museum monitoring procedures from v0.11 to v0.18. After the update, a bug will be fixed. The bug leads that resource-limited fog node to send data too frequently to sink node, when museum is closed at night.

Updating Case E. Update v0.11 to v0.21. We add a load balancing algorithm and a sleep mechanism considering the battery energy remaining.

Updating Case F. Update v0.18 to v0.21. The new version will remove the sensing function from the node. It only retains routing function and forwards data from other ones.

Updating Case G. Update v0.18 to v0.3. After this update, node will install the UDP protocol that supports users directly interacting with the nodes through web browser.

Updating Case H. Update v0.21 to v0.3.

In the second updating set scenario, the above six programs will be randomly updated without fixed direction.

We will test RePage's performance from cache hit rate, energy cost, and storage cost and compare it with current reprogramming approaches such as EasiCache and Tiny Module-link.

4.2. Cache Hit Rate. First of all, we define cache hit rate as ratio between sum size of changed codes cached and the sum size of ones that need to be modified. Figure 5 shows hit rates of continuously updating cases according to alphabetical order shown in Figure 4. We set cache size in 6144 bytes, and the function page size is set in 512 bytes. As shown in Figure 5, we compare the RMR of RePage and Least Recently Used (LRU) algorithm.

RePage's adopted RMR algorithm considers modification times and range of each function page. However, in beginning phase of continuous updating, because the differences of modification range among each function page are small, weight a is close to 0, and impact of modification times gives more effect on replacement factor (RF). Performance of RMR is similar to LRU.

In updating case A, the codes of data acquisition and wireless transceiver have already been achieved in Oscilloscope. In addition, collection tree protocol (CTP) is usually taken as part of dead code and has been preinstalled by the dead code elimination technique referring to literature [7]. In fact, this updating procedure mainly activates CTP in application layer and adjusts the corresponding function according to EasiRoute_v0.1. However, only parts of changed codes are cached, and RMR's hit rate is only about 60% closed to LRU. In updating cases C and D, in order to fix the bug, some new functions need to be created in node and cannot be cached; thus, RMR's and LRU's hit rates are low.

During beginning phase, it is hard to predict which function pages are "hot." Therefore, RePage's algorithm has 62% hit rate compared with LRU's 63%. After beginning phase (from A to D), the average cache hit rate (from E to H) of RMR and LRU is improved. The reason is that RePage and LRU have enough history information to decide which pages should be cached. Especially in updating case H, two cached

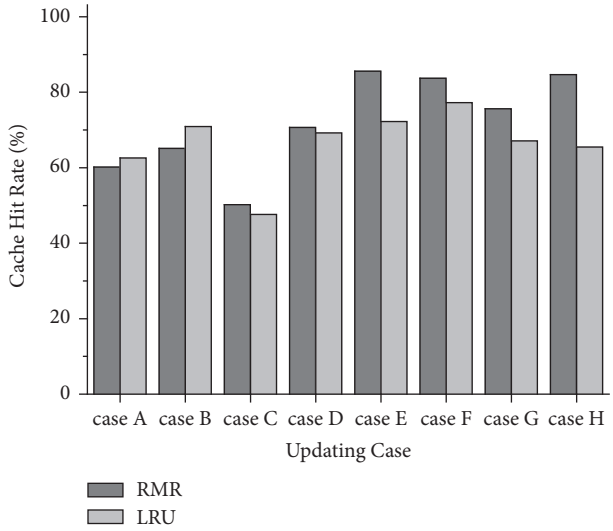


Figure 6: Cache hit rates of randomly updating scenario.

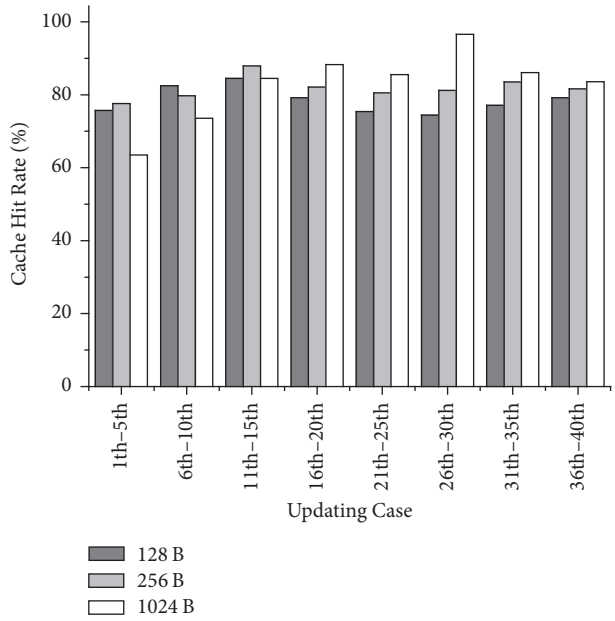


Figure 7: Cache hit rates of randomly updating scenario under different function page size.

function pages are replaced from cache during updating cases D and E according to LRU algorithm that only considers the modification times. But the two functions were largely modified in case B, so they have high RF and cannot be replaced according to RMR.

To further study the performance of RMR, we continuously update these six programs at random way without assigning the updating roadmap and observe hit rate of RMR and LRU. We record hit rates of five times updating and count the average value of hit rates taken as the average hit rate in random continuous updating. Figure 6 shows the average hit rates of 40 times updating. Obviously, in the first 15 updating cases, RMR's average hit rate reached 77.2% compared with 75.1% of LRU.

However, during 16th–40th updating, average hit rate of RMR increases and reaches 90.1%, and LRU only has 79.5%. The variation in performance between RMR and LRU is relative to modification range. In these updating cases, several function pages were not modified in most of cases and were therefore replaced, but once they are modified, this means a number of codes in these pages changed.

As shown in Figure 7, we also test the influence brought about by function page sizes to cache hit rate given cache capacity (6144 B). Small-sized function page means a flexible cache behavior. At the beginning phase, the 128 B function page obtained higher average hit rate. But limit of functions page size also makes some of larger functions be unable to be put in function page. Clearly, after 20 times continuous updating, the average hit rate of 128 B began to decline. But the larger-size function page (1024 B) still increases during 21st–35th updating. What needs to be pointed out is that the flexibility of 128 B page size makes updating hit rate more than 98% in the 10th and 14th updating cases, and its average hit rate has an advantage over 512 B and 1024 B page size during 1st–15th updating case. However, from the perspective of the stability of cache hit rate, the 512 B page size has a good performance during whole continuous updating procedure and is recommended.

4.3. Rebuilding Cost. The main energy cost of over-air reprogramming produced by communication cost is caused by wireless transmitting codes and the rebuilding cost is caused by r/w operations on memory. Table 1 has illustrated the energy cost of r/w operation on different memory. Obviously, energy cost in volatile memory such as RAM is far less than that in the nonvolatile memory such as internal/external flash. Take the TelosB nodes as example: the energy cost of r/w operation on the internal flash is 14.6 times that on RAM, and the energy cost of r/w operation on external flash is 19.3 times that on RAM. Therefore, when studying rebuilding cost of reprogramming, we focus on the number of r/w operations on internal/external flash (in/ex flash for short).

We compare RePage, Tiny Module-link [14], and EasiCache [10] under continuously updating scenario. Tiny Module-link firstly considered the energy cost of r/w operations. It rebuilds the function in low-power RAM, and then the updated function still needs to be written back into internal flash. EasiCache like RePage uses the low-power RAM caching frequently updating codes.

Tables 3 and 4 show the data size of r/w operations on in/ex flash of MCU in updating cases A–H with respect to three reprogramming approaches. RePage and EasiCache can directly modify codes in RAM and internal flash and do not require participation of external flash. In contrast, Tiny Module-link not only stores the incremental script in external flash but also must write back the modified functions into internal flash from RAM. So, Tiny Module-link must execute more r/w operations on internal flash than on external flash in some updating cases.

Table 3: Comparison of reading operations on external/internal flash among RePage, EasiCache, and Tiny Module-link (bytes).

		Case A	Case B	Case C	Case D	Case E	Case F	Case G	Case H
Tiny Module-link	R Ex flash	3278	1252	4072	1578	1314	872	4214	4562
	R In flash	3025	2456	2122	2894	2848	3010	35780	33678
EasiCache	R Ex flash	—	—	—	—	—	—	—	—
	R In flash	902	614	1322	918	594	828	36678	32172
RePage	R Ex flash	—	—	—	—	—	—	—	—
	R In flash	356	314	422	204	768	266	2232	1866

Table 4: Comparison of writing operations on external/internal flash among RePage, EasiCache, and Tiny Module-link (bytes).

		Case A	Case B	Case C	Case D	Case E	Case F	Case G	Case H
Tiny Module-link	W Ex flash	3678	1252	4072	1578	1314	872	4214	4562
	W In flash	3214	2022	3678	2760	3090	1320	30258	31142
EasiCache	W Ex flash	—	—	—	—	—	—	—	—
	W In flash	578	456	624	1342	1090	1278	30958	31588
RePage	W Ex flash	—	—	—	—	—	—	—	—
	W In flash	636	278	728	186	1488	672	2356	2712

EasiCache and RePage put codes in low-power RAM and effectively reduce the r/w operations on internal flash. The average hit rate of RePage is 6.1% higher than EasiCache in updating B, D, and F of RePage. The reason is that paging mechanism puts the similar functions clustering in one function page and keeps the page in cache. In contrast, EasiCache does not consider the relationship between functions, and sometimes its behavior is like a bouncing ball, which means several functions are replaced and cached repeatedly. High hit rate means that more functions have been saved in the RAM; and there is no need to read internal flash, and written back codes are less.

Updating case E is special. In this scenario, although the hit rate of RePage is still higher than EasiCache, one function page needs to be replaced and written back to the internal flash. Exchanging 512 B function page brings about 34% increase of r/w operations.

In updating cases G and H, several functions are inserted in program due to adding UDP protocol. They need to be written into the internal flash. EasiCache and Tiny Module-link E adopt the same strategy to preserve inserted functions in the free space at the end of the main program (.text segment). However, in this case, the size of function, writing in the internal flash, has been beyond the capacity of internal flash. Given both approaches, runtime system has to readjust the storage location for all functions in internal flash and increase a large number of r/w operations on internal flash. RePage uses paging mechanism and makes each page own independent slop region, which ensures the adjustment occurring in function page. Compared with EasiCache and Tiny Module-link, the number of r/w operations on in/ex flash of RePage decreases by 93.8% and 93.0%.

4.4. Storage Cost. Caching codes inevitably causes a fragmentation problem. Figure 8 shows the fragmentation degree in continuous upgrading cases with respect to RePage and EasiCache. The definition of fragmentation degree has been given by formula (1).

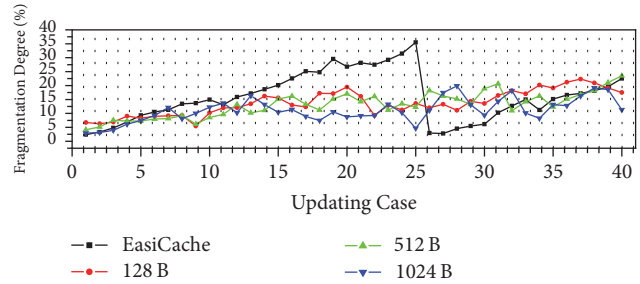


Figure 8: Fragmentation degrees of randomly continuously updating scenario.

In RePage, each function page has an independent slop region $E(x)$ that actually causes storage fragmentation. Therefore, in the beginning of the continuous updating cases, the fragmentation degree of both EasiCache and RePage is not zero. However, due to uncertain function size, after several times of upgrading, the fragmentation degree of EasiCache sharply increases.

EasiCache can reduce fragmentation degree by adjusting function position. For example, during the 5th case, some instructions are deleted from several modified functions which become small, and runtime system switches position of both size-decreasing functions and size-increasing functions. However, the fragmentation degree is not stable. With increased modified functions, after 13th updating case, the size-decreasing functions are unable to provide enough space for size-increasing functions which must be written back to internal flash. Clearly, fragmentation degree of EasiCache is increased generally. In the 25th upgrading case, when the fragmentation degree exceeds 35%, it is hard to cache

Table 5: The decreasing of execution efficiency (%).

Program name	Dec.	Program name	Dec.
Oscilloscope	10.6	EasiRoute_v0.1	12.3
EasiRoute_v0.11	14.5	EasiRoute_v0.18	13.6
EasiRoute_v0.21	12.6	EasiRoute_v0.3	17.9

new functions. All functions have to be readjusted in 26th updating case and fragmentation degree drops to about 3%.

With respect to RePage, each function page has own slop region. The function location adjustment only impacts these functions that are clustered in the same function page. In addition, the size of function pages will also have influence on the fragmentation degree. In Figure 8, we compared fragmentation degree of all upgrades with different page sizes (128 B, 512 B, and 1024 B). Large-size function page contains more similar functions than small-size one and its slop region does not obviously increase. Therefore, 1024 B page has fewer fragmentation spaces and high utilization in the beginning. However, in large-size page, the slop region is relatively far from each function, and the adjustment process is also more difficult than small-size page. With the progress of continuous updating, for the large-size page, each adjustment procedure will cause fragmentation degree to have serious fluctuation obviously shown in vibration curve in Figure 8.

With respect to small-size page, it is convenient to adjust the function entry address due to slop region being closed to functions. Therefore, the average of fragmentation degree of 128 B and 512 B page function decreases by 5.5% and 10.8% compared to 1024 B. Figure 8 shows that the curves of fragmentation degree of 128 B and 512 B also appear smoother than 1024 B.

4.5. The Impact of Two-Step Calling Process on Execution Efficiency. In RePage, invocating functions need two-step calling process. In the first step, it is necessary to assign register (r5) before calling function. In the second step, RePage employs register relative addressing to find the real entry address. With respect to two-step calling process, add an assignment (mov), a calling (call), and a return instruction (ret.). The three instructions, respectively, need 3, 8, and 5 clock cycles. Therefore, a total of 16 clock cycles are added, caused by two-step calling process.

Table 5 shows the decrease of execution efficiency after two-step calling, which includes six programs. The biggest impact is EasiRoute_v0.3, where the execution efficiency is down by 17.9% from the original program. This program is more complex due to joining the UDP protocol module. The execution efficiency of the other five programs declines by 10.6% at least. Obviously, the two-step calling process does have influence on the execution efficiency. However, considering the dynamic updating of the fog computing system, two-step calling is able to reduce modification and manages the function page facilitating. Compared with the energy cost of reprogramming, the added energy cost is negligible, caused by decreasing of execution efficiency.

5. Related Work

The energy efficiency of reprogramming has been a key issue in resource-limited networks always taken as front-end network of fog computing. The early reprogramming researchers pay less attention to energy efficiency and burden of network. For example, Deluge [15] needed to transmit the complete program image and related updating protocol. During the transmission period, reprogramming process makes network unavailable in a long time. Stream [16] attempts to reduce the size of data transmitted by preinstalling the reprogramming protocol in the target node, but it still transmits the entire program image. CORD [17] employs two stages to reduce energy cost. In the first stage, new image is given to special nodes, and then, in the second stage, these special nodes will transmit new image to the rest of ones. Mobile Deluge [18] introduces a mobile base station to ensure the receiver and transmitter within one hop. By this way, it improves quality of transmission link and decreases burden of network. In addition, according to discrete and dynamic characters of resource-limited front-end networks, researchers have proposed the encoding reprogramming approach [19–21]. They adopt the redundant code to guarantee success of transmission. All of them need to transmit the whole image and even redundant codes that caused huge energy cost.

In order to reduce the communication cost in reprogramming, some studies introduce compression algorithm, such as adaptive compression algorithm [22]. Tsiftes et al. directly employ GZIP to compress program image [23]. However, compressing image means a decompression procedure needs to be completed in local resource-limited fog nodes. This procedure must bring up computation cost and storage cost.

In recent years, a lot of incremental reprogramming approaches [1–8] were proposed. The concept of incremental reprogramming is to minimize the communication overhead by merely transmitting the different binary codes between the new and old images. Current incremental reprogramming approaches can fall into two categories: one increases the similarity between new and old images and the other improves the algorithm's efficiency of generating different codes.

In the works of Zephyr [2] and Hermes [1], researchers attempt to fix entry address of function and static variable to improve the similarity of images. Li et al. [3] designed an update-conscious compiler that rearranges registers in new program image according to old version, and, by this way, the similarity degree between two versions is largely improved. On the other side, Hu et al. [5] propose reprogramming with minimal transferred data (RMTD) algorithm that is based on Rsync algorithm; this algorithm reduces the time complexity of Rsync from $O(n^3)$ to $O(n^2)$, but the space complexity reaches $O(n^3)$. That may bring about heavy storage burden for upper computer given large-size program image. Dong et al. combined the advantages of Zephyr and RMTD and proposed R2 [7] and R3 [8], and both approaches take advantage of relative address codes to initialize reference addresses for improving similarity. In addition, they also modify the RMTD algorithm to reduce the space complexity to $O(n)$. All above incremental reprogramming approaches need to rebuild new program image locally. Meanwhile,

rebuilding procedure may incur a large number of read and write operations on nonvolatile memory with high energy cost.

Most of current reprogramming approaches are designed based on the traditional resource-limited networks such as sensor network; the program running in node is hard to change, and researchers consider less rebuilding process within frequent and continuous updating. In Zephyr's work [3], for example, researchers have tested rebuild cost of r/w operation on nonvolatile memory and questioned current reprogramming approach. Kim et al. [14] put forward Tiny Module-link that rebuilds the new program image in the RAM. But it still requires writing back new function into nonvolatile memory. In addition, Koshy and Pandey [12] design extra slop region at the end of each function and support directly modifying function in the nonvolatile memory. In other studies, even if low-power volatile memory is used, the purpose is not absolutely to reduce the rebuilding cost. For example, in Elon's work [24], the researcher uses RAM to store frequently updated code and solves the writing flash failure problem caused by the battery voltage drop.

6. Conclusion

Fog computing system requires frequent task rearrangement, and thus fog nodes may experience continuous update by over-air reprogramming. The energy cost is always bottleneck of reprogramming applied in fog computing system. This paper introduces an incremental reprogramming approach based on function page mechanism applied in fog computing, named RePage. With respect to increment reprograming approaches, they must rebuild new program image in local node, which may incur an amount of rebuilding energy cost. For decreasing the rebuilding cost, RePage tries to cache frequently updated function in low-power volatile memory and use function paging mechanism to combine similar functions into one function page for mitigating fragmentation degree.

In order to improve hit rate, we design a novel page replacement algorithm named recent modification range (RMR) that considers the impact factor of modification range and times at the same time.

Through the continuous update experiment, we showed the performance of RePage from the cache hit rate, storage cost, and energy cost. In future work, we will continue to research differences-code generation method according to the function paging mechanism. This method will simplify rebuilding instructions according to such function similarity and further reduce the amount of transmitting data.

Conflicts of Interest

The authors declare that there are no conflicts of interest.

Acknowledgments

This research work is supported by National Natural Science Foundation of China (NSF) (Grant no. 61502427), by the Zhejiang Provincial Natural Science Foundation of China (Grant no. LY16F020034), and by Key Research and Development Project of Zhejiang Province (no. 2015C01034 and no. 2015C01029).

References

[1] R. K. Panta and S. Bagchi, "Hermes: Fast and energy efficient incremental code updates for wireless sensor networks," in *Proceedings of the 28th Conference on Computer Communications, IEEE INFOCOM '09*, pp. 639–647, Brazil, April 2009.

[2] R. K. Panta, S. Bagchi, and S. P. Midkiff, "Zephyr: efficient incremental reprogramming of sensor nodes using function call indirections and Difference Computation," in *Proceedings of the USENIX Annual Technical Conference*, 2009.

[3] W. Li, Y. Zhang, J. Yang, and J. Zheng, "Towards update-conscious compilation for energy-efficient code dissemination in WSNs," *ACM Transactions on Architecture and Code Optimization (TACO)*, vol. 6, no. 4, p. 14, 2009.

[4] J. Jeong and D. Culler, "Incremental network programming for wireless sensors," in *Proceedings of the First Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks, IEEE SECON '04*, pp. 25–33, Santa Clara, CA, USA, 2004.

[5] J. Hu, C. J. Xue, Y. He, and E. H.-M. Sha, "Reprogramming with minimal transferred data on wireless sensor network," in *Proceedings of the IEEE 6th International Conference on Mobile Adhoc and Sensor Systems, MASS '09*, pp. 160–167, China, October 2009.

[6] M. Ajtai, R. Burns, R. Fagin, D. D. Long, and L. Stockmeyer, "Compactly encoding unstructured inputs with differential compression," *Journal of the ACM*, vol. 49, no. 3, pp. 318–367, 2002.

[7] W. Dong, Y. Liu, C. Chen, J. Bu, C. Huang, and Z. Zhao, "R2: incremental reprogramming using relocatable code in networked embedded systems," *Institute of Electrical and Electronics Engineers. Transactions on Computers*, vol. 62, no. 9, pp. 1837–1849, 2013.

[8] W. Dong, B. Mo, C. Huang, Y. Liu, and C. Chen, "R3: Optimizing relocatable code for efficient reprogramming in networked embedded systems," in *Proceedings of the 32nd IEEE Conference on Computer Communications, IEEE INFOCOM '13*, pp. 315–319, Italy, April 2013.

[9] J. Polastre, R. Szewczyk, and D. Culler, "Telos: enabling ultra-low power wireless research," in *Proceedings of the 4th International Symposium on Information Processing in Sensor Networks (IPSN '05)*, pp. 364–369, April 2005.

[10] J.-F. Qiu, D. Li, H.-L. Shi, W.-Z. Du, and L. Cui, "EasiCache: A low-overhead sensor network reprogramming approach based on cache mechanism," *Jisuanji Xuebao/Chinese Journal of Computers*, vol. 35, no. 3, pp. 555–567, 2012.

[11] "MSP430F15x, MSP430F16x, MSP430F161x Mixed Signal Microcontroller," http://focus.ti.com/lit/ds/symlink/msp430f1611.pdf.

[12] J. Koshy and R. Pandey, "Remote incremental linking for energy-efficient reprogramming of sensor networks," in *Proceedings of the 2nd European Workshop onWireless Sensor Networks, EWSN '05*, pp. 354–365, Turkey, February 2005.

[13] D. Li, W. Liu, and L. Cui, "EasiDesign: An improved ant colony algorithm for sensor deployment in real sensor network system," in *Proceedings of the 53rd IEEE Global Communications Conference, GLOBECOM 2010*, USA, December 2010.

[14] S.-K. Kim, J.-H. Lee, K. Hur, K.-I. Hwang, and D.-S. Eom, "Tiny module-linking for energy-efficient reprogramming in wireless sensor networks," *IEEE Transactions on Consumer Electronics*, vol. 55, no. 4, pp. 1914–1920, 2009.

[15] J. W. Hui and D. Culler, "The dynamic behavior of a data dissemination protocol for network programming at scale," in *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems (SenSys '04)*, pp. 81–94, ACM, 2004.

[16] R. K. Panta, I. Khalil, and S. Bagchi, "Stream: Low overhead wireless reprogramming for sensor networks," in *Proceedings of the 26th IEEE International Conference on Computer Communications (IEEE INFOCOM '07)*, pp. 928–936, USA, May 2007.

[17] L. Huang and S. Setia, "CORD: Energy-Efficient Reliable Bulk Data Dissemination in Sensor Networks," in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM '08)*, pp. 574–582, Phoenix, AZ, USA, 2008.

[18] X. Zhong, M. Navarro, G. Villalba, X. Liang, and Y. Liang, "MobileDeluge: Mobile code dissemination for wireless sensor networks," in *Proceedings of the 11th IEEE International Conference on Mobile Ad Hoc and Sensor Systems, MASS 2014*, pp. 363–370, USA, October 2014.

[19] A. Hagedorn and D. Starobinski, "Rateless deluge: over-the-air programming of wireless sensor networks," in *Proceedings of the IEEE International Conference on Information Processing in Sensor Networks (IPSN '08)*, Proceeding of ACM, April 2008.

[20] M. Rossi, G. Zancas, L. Stabellini, R. Crepaldi, A. F. Harris III, and M. Zorzi, "SYNAPSE: A network reprogramming protocol for wireless sensor networks using fountain codes," in *Proceedings of the 5th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks, SECON 2008*, pp. 188–196, USA, June 2008.

[21] W. Du, Z. Li, J. C. Liando, and M. Li, "From rateless to distanceless: enabling sparse sensor network deployment in large areas," in *Proceedings of the 12th ACM Conference on Embedded Networked Sensor Systems (SenSys '14)*, pp. 134–147, Memphis, Tenn, USA, November 2014.

[22] C. M. Sadler and M. Martonosi, "Data compression algorithms for energy-constrained devices in delay tolerant networks," in *Proceedings of the 4th International Conference on Embedded Networked Sensor Systems (SenSys '06)*, pp. 265–278, ACM, 2006.

[23] N. Tsiftes, A. Dunkels, and T. Voigt, "Efficient sensor network reprogramming through compression of executable modules," in *Proceedings of the 5th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON '08)*, pp. 359–367, June 2008.

[24] W. Dong, Y. Liu, C. Chen, L. Gu, and X. Wu, "Elon: Enabling efficient and long-term reprogramming for wireless sensor networks," *ACM Transactions on Embedded Computing Systems*, vol. 13, no. 4, article no. 77, 2014.

Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 3681270, 11 pages
https://doi.org/10.1155/2018/3681270

Research Article

A Novel UDT-Based Transfer Speed-Up Protocol for Fog Computing

Zhijie Han,[1] **Weibei Fan**,[2] **Jie Li**,[3] **and Miaoxin Xu**[1]

[1]*Institute of Data and Knowledge Engineering, Henan University, Kaifeng, China*
[2]*School of Computer Science and Technology, Soochow University, Suzhou, China*
[3]*Software College, Henan University, Kaifeng, Henan, China*

Correspondence should be addressed to Zhijie Han; hanzhijie@126.com

Received 24 January 2018; Revised 24 March 2018; Accepted 3 April 2018; Published 13 May 2018

Academic Editor: Xuyun Zhang

Fog computing is a distributed computing model as the middle layer between the cloud data center and the IoT device/sensor. It provides computing, network, and storage devices so that cloud based services can be closer to IOT devices and sensors. Cloud computing requires a lot of bandwidth, and the bandwidth of the wireless network is limited. In contrast, the amount of bandwidth required for "fog computing" is much less. In this paper, we improved a new protocol Peer Assistant UDT-Based Data Transfer Protocol (PaUDT), applied to Iot-Cloud computing. Furthermore, we compared the efficiency of the congestion control algorithm of UDT with the Adobe's Secure Real-Time Media Flow Protocol (RTMFP), based on UDP completely at the transport layer. At last, we built an evaluation model of UDT in RTT and bit error ratio which describes the performance. The theoretical analysis and experiment result have shown that UDT has good performance in IoT-Cloud computing.

1. Introduction

Internet of Things (IoT) and cloud computing are two very different technologies that are both already part of our life. IoT has received attentions for years and is considered as the future of Internet and also recognized as one of the most important areas of future technology and is gaining vast attention from a wide range of industries [1–3]. Despite the increasing usage of cloud computing, there are still issues unsolved due to the inherent problem of cloud computing such as unreliable latency, lack of mobility support, and location-awareness [4]. However, cloud computing is considered as a promising computing paradigm due to the limited computation/storage on smart devices, which can provide elastic resources to applications on those devices.

Fog computing is proposed to enable computing directly at the edge of the network, which satisfies the strongly requirements in processing big data influx in real-time and functioning the available bandwidth bounds. Fog computing is an extension of cloud computing to the emerging IoT. Fog computing is not made up of powerful servers but consisted of a variety of functional computers with weaker performance and more dispersive performance. It is connected to a variety of devices, including mobile phones, wearable devices, smart TV, smart homes, smart cars, and even smart cities. It is a novel paradigm realizing distributed computing, network services, and storage from beyond cloud computing data centers up until the devices. At present, the amount of equipment to collect data and the amount of data processing are increasing exponentially. Public cloud computing provides computing space for processing the data through a remote server. However, after uploading these data to remote servers for analysis, it will take time to transfer the results to the original location, which will slow down the process of real-time quick response. Fog computing is a newly introduced concept that aims to put the cloud closer to the end users (things) for better quality of service [5, 6].

The transmission control protocol (TCP), the de facto transport protocol of the Internet, substantially underutilizes network bandwidth over high-speed connections with long delays [7]. Although the transmission control protocol (TCP) plays a major role in the Iot-cloud today, TCP does not suit the high network bandwidth-delay product (BDP) because of its congestion control algorithms. UDT is a

UDP-based approach, to the best of our knowledge, it is the only UDP-based protocol that employs a congestion control algorithm targeting shared networks. Compared to transmission control protocol (TCP), UDP does not yield retransmission delay, which makes it attractive to delay sensitive applications. In Iot-Cloud, most of all, the high performance of the congestion control of TCP is composed of four core algorithms: slow start, congestion voidance, fast retransmit, and fast recovery. In order to prevent network congestion where the congestion window size (CWnd) grows too fast, we need to set a threshold (ssthresh) of slow start state. With the changing of CWnd, different algorithms are used in different scene. Gu and Grossman [8] proposed the UDT protocol, which is an improvement on UDP. Considering the characteristics of the fog computing that concentrates data, data processing, and application on the edge of the network, we improve the UDT protocol, which can be better applied to the data transmission of fog computation.

With the size and complexity of cloud computing expanding, the architecture pattern is becoming more crucial on the performance of cloud computing. Different functions of cloud computing have different requirements for architecture patterns. The traditional UDT protocol is based on the Client and Server (C/S) mode, which is the most common distributed computing model [9]. The C/S model is very suitable for the situation where resources are relatively centralized. It is also easier to implement, but it may cause a slow response to all clients when the amount of access becomes larger. In modern data centers, the adoption of more high-speed Ethernet uplink has been the mainstream. The host (server) connection has been transferred from the 1G network to the 10G network and continues to evolve towards the 25G network. The connection between switches will evolve from 10G to 40G and even 100G. Just as the data center of Facebook is built [10], every 10G bandwidth top switch is connected to the optical fiber switch with a 40G uplink and then access the underlying server. The traffic inside the data center is a huge number. Its scale is usually 1000 times the outflow of the data center, and more and more data centers are experiencing the change of data transmission speed.

Peer-to-peer (P2P) is a mature network that share computer resources and services through direct exchange of information between network nodes [11]. The P2P architecture eliminates the central position of the server, such that servers can share computer resources and services directly in each system through exchange. The concept of fog computing itself does not clearly define the location and distribution of the calculation, and this is not the main problem to be solved in the fog computing. In order to take advantage of the increasing computing power of personal devices, the concept of P2P cloud is proposed [12]. We improve the UDT protocol with combining the characteristics of fog computing and P2P network and propose a UDT-based transfer speed-up protocol for fog computing.

The rest of the paper is organized as follows. Section 2 gives an overview of the UDT protocol. Section 3 presents an overview of the UDT and describes its design and implementation. Section 4 gives a comparison between RTMFP and UDT and gives an experimental evaluation of the UDT performance. Section 5 concludes the paper.

2. Related Work

New network devices are emerging and starting to connect. The data generated and sent to the cloud will increase exponentially. If storage and computing power follow Moore's law, it means that they will double every 18 months. The relative speed of bandwidth seems to be much slower. Some research institutions estimate that the global bandwidth rate is growing less than 40% per year. This means that there will be more data to be sent to the cloud, but it will be constrained by bandwidth speed.

As a consequence, real-time and latency-sensitive computation service requests to be responded by the distant Cloud data centers often endure large round-trip delay, network congestion, service quality degradation, etc [13]. A new transport protocol as a timely solution is required to address this challenge. With IoT becoming a major component of fog computing, it is becoming more and more important to improve quality of service (QoS) in fog computing networks. UDT is a UDP-based protocol that proposed by Gu and Grossman [8], which is a radical change by introducing a new transport layer protocol involving changes in routers. The new protocol is expected to be easily deployed and easily integrated with the applications, in addition to utilizing the bandwidth efficiently and fairly.

A lot of researches have been carried out by this protocol. Currently most interaction between the IoT devices and the supporting back-end servers is done through large scale cloud data centers, location awareness [14], and widespread geographical distribution for the IoT. Liu et al. presented analytical results based on UDT that highlight transport characteristics specific to dedicated connections and established the monotonicity, concavity, and stability of the throughput profiles under various configurations. Park et al. [15] proposed a congestion degree based MTR estimation algorithm, which tried to classify more depending on the congestion degree to estimate more actual available bandwidth. Their proposed method showed that the fairness problem among the competing flows is significantly resolved in comparison with that of UDT. In [16], Park et al. also proposed a congestion degree based MTR estimation method, which can be preemptively estimated compared with that of round trip delay information on sender side. The simulation results showed that the competing flows with the proposed method can fairly occupy the network bandwidth. Murata et al. [17] proposed a novel transport protocol, named high-performance and flexible protocol (HpFP), to show high throughputs for the HbVRS in LFNs with packet loss. They examine basic performances of multiple streams of the HpFP and the UDT on a laboratory experiment simulating an international LFN.

As main concepts of cloud computing, fog provides service which includes computation, storage, and networking services to end users, but in the end of the network. Despite the countless benefits of fog, the research in this field is still immature, and many researchers still are working on defining

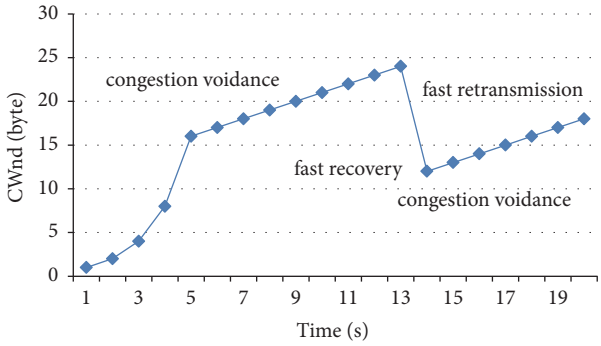


Figure 1: The congestion control algorithm of TCP.

vision, basic notions, and challenges of fog computing [4, 14, 18, 19]. Bernardo and Hoang [20] outlined security requirements for UDT implementation and proposed practical encryption methods for securing UDT within the network layer.

An open research challenge is to explore the potential benefits of fog computing and IoT. In other words, one must study that quality of transparent service will be improved by having a layer of fog nodes between IoT and the cloud. Recent work addressed the design of TCP or UDP for transmission from mobile subscribers to edge clouds, to achieve a power-delay trade-off. Despite their solid contributions, the proposed approach is limited to the cellular network infrastructures. It assumes an IoT device can be only a User Equipment (UE) and that edge servers must be attached to base stations. Additionally, by not considering the cloud in the approach, scenarios where IoT-cloud or fog-cloud communication takes place are not handled.

3. Improved UDT Protocol for Fog Computing

For fast retransmission and fast recovery in IoT network, when an Iot client which we called sender receives three Native Acknowledgements (NAKs), sender will retransmit the lost packets right away and cut the ssthresh by half. At the same time, the algorithm sets congestion window CWnd equal to ssthresh and congestion voidance starts. The whole process is shown in Figure 1.

In Figure 1, the initial window size is 16, when the CWnd equals 16, the slow start stops, and the congestion voidance begins. In the stage of congestion voidance, the increase of CWnd is additive. As shown in Figure 1, when CWnd equals 24, network starts blocking, and at this time, the ssthresh is set to 12 (half of 24).

The window based AIMD (additive increase multiplicative decrease) control algorithm of TCP suffers from random loss as the BDP increases to higher [21, 22]. In order to resolve the TCP's inefficiency problem in high BDP links, we use a new protocol, called UDT. UDT is an application level program and, in transport layer, it uses UDP to transmit data. UDT is a reliability and security streaming data transport protocol which is end-to-end, connection oriented, unicast, and duplex.

As illustrated in Figure 1, countless dividing lines exist between H_1 and H_2. The abscissa represents time and the ordinate represents the window size. For two-dimensional space, we want to find the best divider line. For the n-dimensional space, our ultimate goal is to find the best classification of the super-plane, that is, to find the final decision-making boundaries.

Table 1: UDT data packets header.

	Packet sequence number
F	Message number
Time stamp	
Destination socket ID	

3.1. UDT Structure. The UDT layer has five function components: the API module, the sender, the receiver, the listener, and the UDP channel. There are four data components: sender's protocol buffer, receiver's protocol buffer, sender's loss list, and receiver's loss list [6].

The API module is responsible for interacting with applications. When application sends data, the data packets are passed from sender's buffer to UDP channel. On the other hand, receiver reads data from UDP channel and the packets will be reordered in receivers buffer. Receiver will check receiver's loss list; if there are packets in receiver's loss list, receiver will send a NAK to sender and when sender receives a NAK, the sender's loss list will be updated, and sender sends this packet again until the receiver receives this packet. The main transmitting procedure is shown in Figure 2.

3.2. The Packets Type of UDT. UDT has two kinds of packets: the data packets and the control packets. The types of packets are distinguished by the first bit (flag bit) of the packet header, and when the flag is 0, it represents a data packet, and otherwise it represents a control packet. As shown in Table 1, a UDT data packet contains a packet-based sequence number, a message number, a destination socket id, and a relative timestamp. "*F*" represents the position of this packet in the flow as follows: "10" is the first packet, "01" is the last one, "11" represents that there is only one packet, and "00" is any packet in the middle. The next bit whose value is 0 represents that the message is sent out of order, and if the value is 1 in this position shows that the message is sent in order. The 32-bit time stamp is a relative value starting from the time when the connection is set up.

The control information is shown in Table 2. The comparison of control packets with data packets in type and revered field is illustrated in Table 1. There are 8 types in control packets, such as handshake, keep-alive, ACK, and NAK. The revered field defines a new type of control packet or adds new variables in an existing control package for a new congestion control algorithm.

3.3. PaUDT Protocol Mode. In the structured P2P model, all information is scattered in different nodes of the network in the form of Hash list, and a huge distributed hash table (DHT) is constructed by the whole network (Figure 4) [23]. The whole network information is stored and retrieved distributed in the application layer. Kademlia is a typical

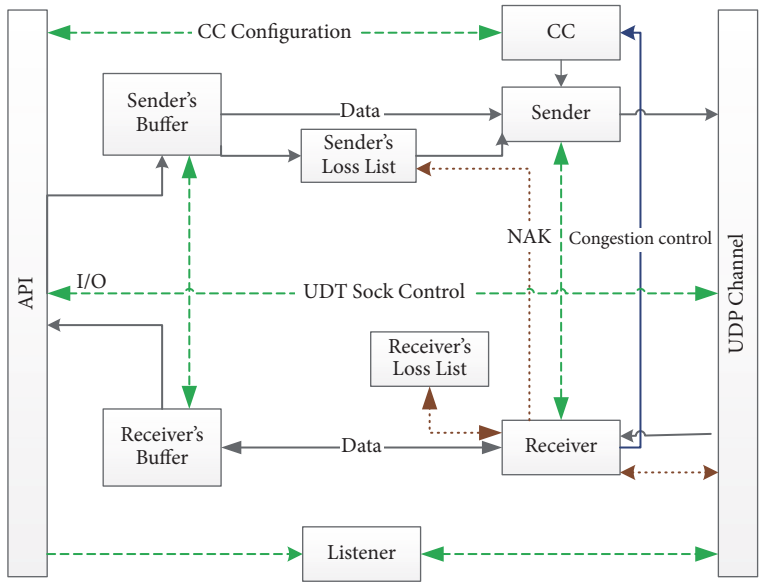


Figure 2: UDT/CCC implementation.

Table 2: UDT control packets header.

Type	Revered
Additional info	
Time stamp	
Destination socket ID	
Control information field	

structured model, which uses "XOR" to measure the distance between network nodes. Such a measure does not affect the efficiency and scalability of the network and provides better fault tolerance and flexibility. Consistent hash functions are widely used in distributed data storage, which guarantees dynamic changes in the distributed environment. The mapping space of the Hash function will be slowly changed and still maintain the load balance, thus ensuring the availability and efficiency of the system. The P2P network is also a dynamically changing distributed system, where nodes continue to join or leave the hash space in constant change. It relies on the consistency hash function to ensure the efficiency and load balancing of its location, query, data storage, and replication.

DHT technology is to add a separate DHT layer between the P2P network application layer and the network layer to locate and locate the resources of the P2P network. See Figure 3. DHT uses the Hash function to speed up the lookup, improve security, and make management convenient, without taking up too much of the network bandwidth. Kademlia is a typical structured P2P model, which assigns a unique and random node ID to each node. Each object assigns a similar object ID (also called key) and generates ID using 160 bit SHA.1 function [24]. The object index is responsible for the node ID nodes closest to the object ID (the proximity is measured in a foreign or distance). In the Kademlia model, all nodes are treated as leaves of a two forked trees, and the location of each node is uniquely determined by the shortest prefix of its ID value. For any node, this binary tree can be decomposed into a series of continuous subtrees without its own subtrees. The Kademlia model ensures that each node knows at least one node of its subtrees, as long as these subtrees are not empty. Thus any node can be positioned on the node by the ID value of other nodes.

The Kademlia model uses XOR operation as a measure of the distance between two nodes, assuming that there are two nodes in the Kademlia model, and the node ID is x and y, respectively. Then the distance between them is $d(x, y) = x \oplus y$, such as $x = 1011$, $y = 0010$, $d(x, y) = 1011 \oplus 0010 = 1001$. Although the difference or distance is non-European (non-Euclidean geometric measure), it is reasonable. It is obvious that $d(x, x) = 0$; $d(x, Y) > 0$, if $x \neq y$, and for any x, y, there is $d(x, y) = d(y, x)$, which is called symmetry. In addition, the XOR distance also has a triangle attribute $d(x, y) + d(y, z) \geq d(x, z)$ and triangular attributes from the following two lemmas:

$$d(x, z) = d(x, y) \oplus d(y, z); \quad (1)$$

For arbitrary $a \geq 0$,

$$b \geq 0, \quad (2)$$

there is $a + b \geq a \oplus b$.

The XOR distance is unidirectional in Kademlia models. Unidirectionally refers to any point X and distance $d > 0$, and only the only point y in the network satisfies the $d(x, y) = d$. Unidirectionally guarantees that all positioning of the same data object will eventually get together in the same path, and the more likely the more backward converging will be.

The following is the schematic diagram of the communication implementation for the PaUDT protocol of the P2P model.

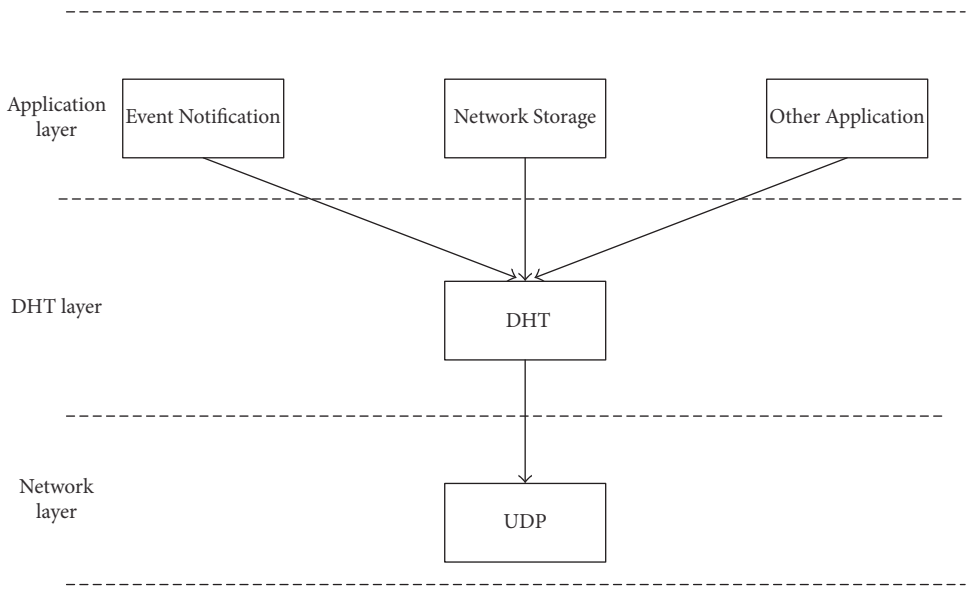


Figure 3: Structure of DHT.

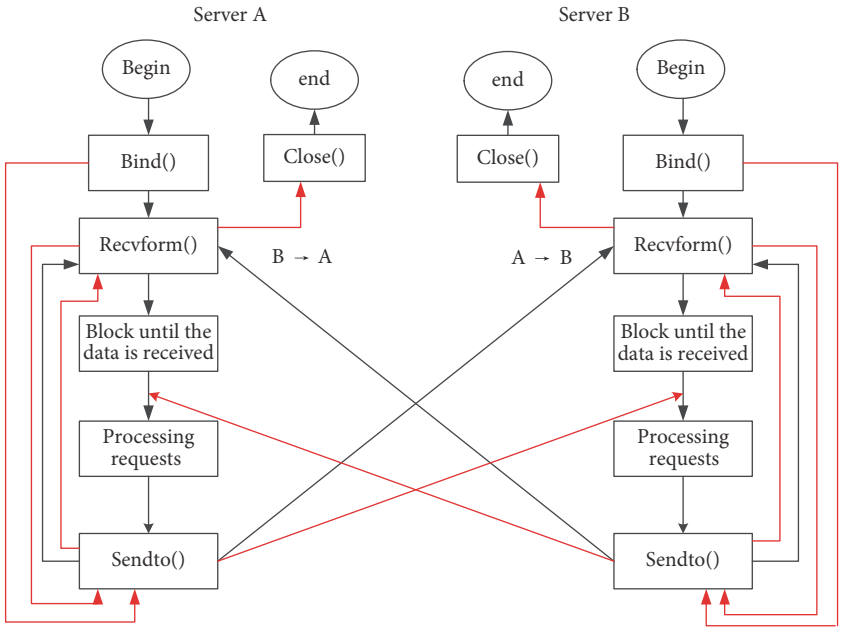


Figure 4: PaUDT communication in P2P mode.

We assume that there are two equal communication sides: server A and server B, and the black line part flow chart is the running process as server; part of the red line flow chart is the running process as the client.

4. RTMFPs Congestion Control Algorithm

In this section, we present the experimental results, which evaluate and characterize the RTMFP and PaUDT, in terms of congestion control algorithm.

RTMFP and UDT are end-to-end, connection-oriented, high reliability, and full-duplex protocol, and there are some same places. For example, they establish connections through four-way handshake, and both application layer protocols which are built on UDP in transport layer. Firstly, the type of data which is transported by RTMFP and UDT is different. RTMFP is used for the real-time and bulk data, but UDT only transports bulk data in the present circumstances. Secondly, unlike UDT, RTMFP is both supporting CS model and P2P model. The last and the major difference is the congestion control algorithm. Through the CCC between RTMFP and UDT, we know which protocol has higher efficiency. Section 2 introduces UDTs CCC simply; this section gives some information about RTMFPs CCC.

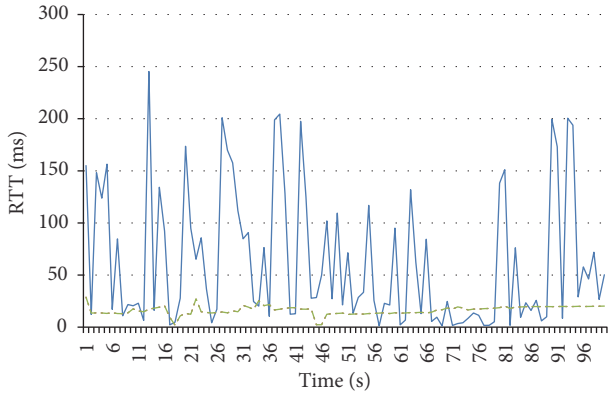


Figure 5: The value of RTMFPs RTT.

A sender of RTMFP increases and decreases its CWnd according to ACK and loss (NAK or timeout) and according to the congestion control and avoidance algorithms [25] like the CCC of UDT. In RTMFP, algorithm also starts from slow start; but not like UDT, slow start's exponential increase rate is adjusted to double every approximately 3 round trips and the multiplicative decrease cuts CWnd by one-eighth on loss and the additive increase is done at half the normal rate (incrementing at 384 bytes per round trip). Because of real-time data, RTMFP has a signal (Time Critical Reverse Notification) which is used to notice other senders who just send original data to decrease the value of CWnd, and if a sender receives a Time Critical Reverse Notification when it is in slow start phase, the sender must stop exponential increase within 800 milliseconds, unless the sender is itself currently sending time critical data to the far end.

From Sections 2 and 3, we know the difference between RTMFP and UDT in congestion control algorithm, and in order to compare which algorithm has more advantages, we design an experiment as follows.

5. Experiments of RTMFP and UDT

In order to contrast which protocol has higher efficiency, we design a experiment in a local area network. This experiment needs six computers; in three computers Linux system for UDT was installed and in others Win7 system for RTMFP is installed.

Install an open platform Adobe Flash Media Server (FMS). Then a computer as server uses FMS to transfer a video to clients, and clients receive the video; at the same time, one client use a capture tool (Wireshark) to capture packets.

After getting these packets, we use a filter tool in Wireshark to choose related packets (these packets are transferred by server), and calculate RTT according to the timestamp of packet's header, through an open UDT source code to transfer the same video with RTMFP. As was stated above, we use Wireshark to capture related packets and calculate UDTs RTT. Contrast the RTT's value of RTMFP and UDT.

As shown in Figure 5, the solid line represents RTMFP's RTT and the change of UDT's RTT is drawn by dotted line. The lines vary greatly. Since a lot of peaks were caused by the curve of RTMFP's RTT, the line of UDT's RTT remains relatively smooth and with smaller value. Although the experiments are in the same environment, the process of receiving video is different. When disposing video, the method of RTMFP is more trouble than UDT. RTMFP plays the video when the video is downloaded, but UDT adopts the method that after this video is received completely, this video is played. Because of different processes, it does not pay special attention to the difference of the value of RTT between UDT and RTMFP, and we need to consider the steadiness of UDT and RTMFP in the experiment.

5.1. Analysis of the Impact of PaUDTs RTT. Based on the system model, performance evaluation is mainly to solve the following problems:

(1) What conditions that network traffic need to satisfy to achieve optimal system performance.

(2) How different parameters influence optimal performance and we set two scenes to analysis the problem.

5.1.1. Calculation of the Most Suitable Maximum Segment Size (MSS). When UDT sends data packets, UDT always tries to pack application data into fixed size packets (the fixed size of data packets is the MSS that negotiated by sender and receiver), unless there is not enough data to be sent [8]. As well known, the bigger the MSS, the smaller the number of packets, but as the buffer of router (MTU) and the error rate in network are limited, the MSS is limited. If the MSS we set in advance is bigger than the value of MTU, this packet will be divided into several fragmentations in network layer, and the receiver must spend more time to reassemble these unordered fragmentations by sequence number, and because in network layer, there is no acknowledge mechanism, if one fragmentation is lost, the sender has to retransfer the whole packet. We know the bigger packets can cause higher error rate and have the danger of segmentation collapse [26], but smaller packets cause more overhead because of their packet headers. So in order to make the total transmission time the shortest and the channel utilization the highest, we must calculate the optimal MSS in the network.

Before calculating the optimal MSS, we give some definitions. P_b is the bit error ratio, P_f represents the wrong frame rate, and l_f is the total length of data part plus control information part and it represents the length of frame and we set its length to N bits. R is the sending rate, and $t_f = l_f/R$ represents the time that a frame is delivered [8, 27]. It is known that the bit error ratio has a relation of the wrong frame rate, and in order to extrapolate the frame error ratio from the bit error rate, CCITT (International Telephone and Telegraph Consultative Committee) puts forward a simple Poisson distribution model or binomial distribution model that is used to describe the random error. Taking the binomial distribution model, we can get the rate ($P_N(m)$) that m bits are wrong in one frame and the result is shown in formula (1) [8, 27]:

$$P_N(m) = C_N^m P_b^m \left(1 - P_b\right)^{N-m}. \quad (3)$$

If one bit in the frame is wrong, this frame is wrong, so we can get the wrong frame rate according to formula (2) as follows:

$$P_f = 1 - P_N(0) = 1 - \left(1 - P_b^N\right). \tag{4}$$

If $P_b \to 0$, according to the Poisson distribution,

$$P_N(m) = C_N^m \times P_b^m \times \frac{(1-P_b)^{N-m}(NP_b)^m}{m!e^{-NP_b}}. \tag{5}$$

So

$$P_f = 1 - P_N(0) = 1 - e^{-NP_b}. \tag{6}$$

When $NP_b \ll 1$, we can get an approximate value of P_f as formula (3):

$$P_f \approx N \times P_b = l_f P_b. \tag{7}$$

We need to calculate the average time of a frame transmission (t_{av}). So as to get an average delay (Delay), we send data in which length is equal to $(n + H)$, so $l_f = n + H = N$. $t_a = A/R$ represents the time that the confirm frame (ACK or NAK) is sent and A is the length of confirmed frame. t_{out} is the minimum time that the retransmission timer is set. In order to get the value of t_{out}, we must know the process time in a node (t_{pr}) and the propagation delay (t_p). The minimum retransmission timer must satisfy a frame that is received by receiver and a sender receives a confirmed frame from a receiver, so $t_{\text{out}} = t_p + t_{\text{pr}} + t_a + t_p + t_{\text{pr}} = 2 \times t_p + 2t_{\text{pr}} + t_a$. t_T is the minimum interval time for the two frames to be successfully sent.

$$t_T = t_f + t_{\text{out}}. \tag{8}$$

So according to the minimal interval time and the wrong frame rate, we can calculate the average time:

$$t_{\text{av}} = \frac{t_T}{\left(1 - P_f\right)}. \tag{9}$$

Now, if the length of file that we want to transport is sum, the average transmission delay equals the average transmission time of a frame multiply by the number of frames which are transported:

$$\text{Delay} = t_{\text{av}} \times \frac{\text{sum}}{n} \tag{10}$$

$$= \frac{(\text{sum}/R)\left[n + H + A + 2\left(t_{\text{pr}} + t_p\right) \times R\right]}{n\left(1 - nP_b - H \times P_b\right)} \tag{11}$$

$$= \frac{(\text{sum}/R)\left(n + H + t_{\text{out}} \times R\right)}{n\left(1 - nP_b - HP_b\right)}. \tag{12}$$

In order to acquire the optimal MSS, we need derivation of Delay:

$$P_b \times n^2 + 2P_b \times \left(H + t_{\text{out}}R\right) \times n - \left(H + t_{\text{out}} \times R\right) \times \left(1 - HP_b\right) = 0. \tag{13}$$

That is,

$$n = \frac{\left(\left(1 - HP_b\right)/P_b\right)}{1 + \sqrt{\left(1 + \left(1 - HP_b\right)\right)/\left(P_b\left(H + t_{\text{out}} \times R\right)\right)}}. \tag{14}$$

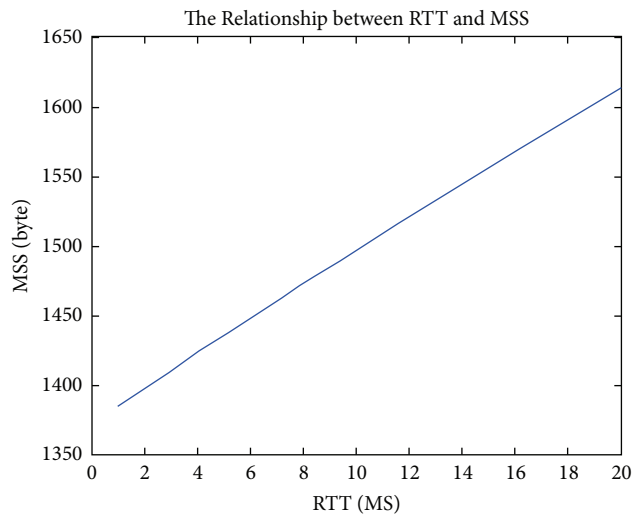


Figure 6: The relationship between RTT and MSS.

If we set A equals H, so we can get a simple new formula (11) from formula (10):

$$n \approx \sqrt{\frac{C}{P_b}} = \sqrt{\frac{2 \times H + 2 \times \left(t_p + t_{\text{pr}}\right)R}{P_b}}. \tag{15}$$

And formula (11) must satisfy the condition that $l_f \times P_b \ll 1$. Figure 6 introduces the relationship between RTT and MSS. Figure 7 describes the relationship between the bit error ratio and MSS. From these two figures, we know that the MSS is advanced with the growth of RTT but is reduced with the growth of the bit error ratio. In the physics experiment, we can get the length of header from wireshark, so $H = 52$ bytes, RTT = 0.01 s, and according to experiments, the average rate in channel is 2 MB/S, so $R \approx 2$ MB/s. $P_b = 6.915 \times 10^{-6}$ bit. So in order to get Figures 6 and 7, we set the rate as 2 MB/S, and when analyzing the effect of RTT on MSS, the bit error ratio is 6.915×10^{-6} bit, and, conversely, when analysing the effect of the bit error ratio on MSS, the RTT is set to 0.01 s.

As shown in Figure 6, there are some values of MSS bigger than 1500 bytes, but now in data link, the biggest value of MSS is 1500 bytes, so we set the value which is equal to 1500 bytes or bigger than 1500 bytes to 1500 bytes.

According to formula (11) and the length of header, the rate in channel, the error bit rate, and the value of RTT, we can get that the optimal MSS equals 1500 bytes.

Although we can get a minimum transmission time using the optimal MSS in network in theory, there may be a practical problem that the MTU is less than the optimal MSS. If the optimal MSS is more than MTU, the packet (the size equals to the optimal MSS) must be fragmented in network layer. So in order to avoid fragment in network, we must set the most suitable MSS in upper layer. This asks us to compare the optimal MSS and the MTU and choose the minimum value as the most suitable MSS.

In order to complete this comparison, UDT is shown in Figure 8. UDT's sender and receiver negotiate the MSS during connection setup. As shown in Figure 8, at first, sender sends

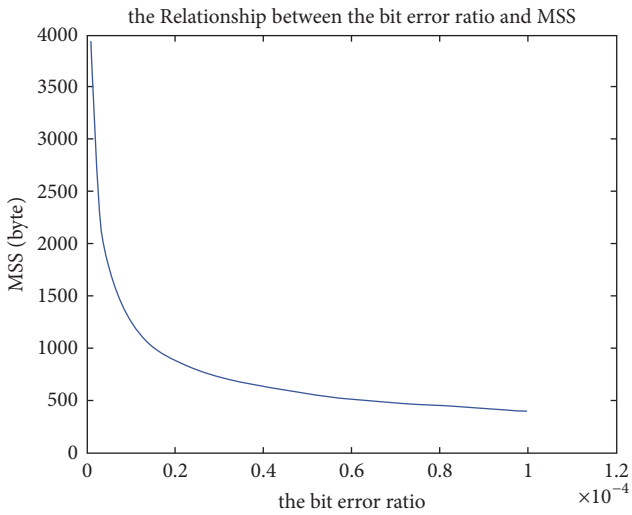


Figure 7: The relationship between MSS and the bit error ratio.

$$\text{MSS} = \sqrt{\frac{2 \times H + 2 \times (t_p + t_{\text{pr}}) \times R}{P_b}};$$

if $rt \rightarrow rt_r mx \cdot rmx_m tu$ **then**
  if $rt \rightarrow rt_r mx \cdot rmx_m tu <$ MSS **then**
    MSS $= rt \rightarrow rt_r mx \cdot rmx_m tuCH$;
  end if
end if

Algorithm 1

a Connection Handshake control packet which includes MSS information; if sender sends packets via a complex network, the MSS will be compared to the MTU of intermediate equipment in network, and if MTU < MSS, the most suitable MSS must equal the minimum MTU, but if MTU > MSS, the most suitable MSS also equals the MSS. After the receiver receives the handshake control packet and gets the most suitable MSS, the receiver sends an ack packet with the most suitable MSS. The sender gets the most suitable MSS and chooses it as the fixed number of a packet. This way has two benefits: first, choosing the minimum MSS as the final MSS can avoid frame being fragmented in network layer and, second, choosing the optimal MSS or the minimum MSS as the final MSS can increase utilization ratio of bandwidth.

In order to achieve the negotiation about MSS, we need to get the value of MTU in intermediate equipment. It is known that there are some structures in the head file of router, for example, reentry and rt_{metrics} structure. In the structure of rt_{metrics}, there is a variable-rmx_{mtu} that is used to get the value of MTU of router and reentry contains the rt_{metrics} structure. In UDT, we set the most suitable MSS as shown in Algorithm 1.

In the statements, $rt \rightarrow rt_{rmx} \cdot rmx_{mtu}$ represents the value of MTU in network, and H is the length of header of UDT.

In our experiments, the MTU of the routers in the data link is 1500 bytes, and the optimal MSS also equals 1500 bytes.

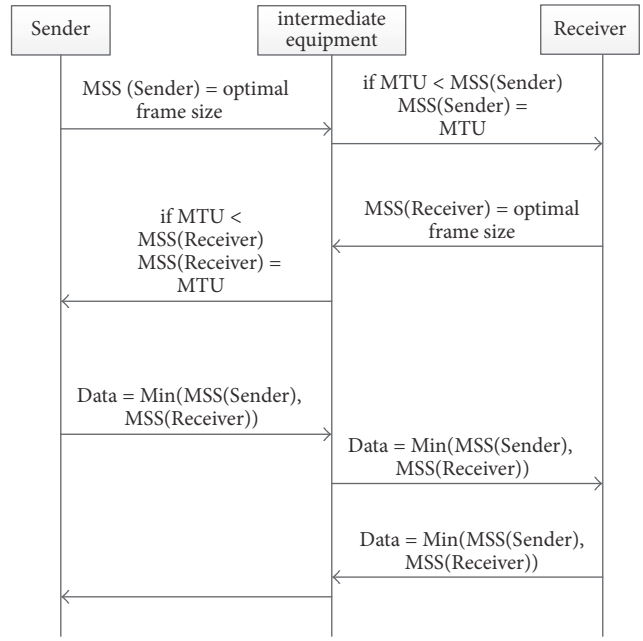


Figure 8: UDT negotiates the MSS between sender and receiver.

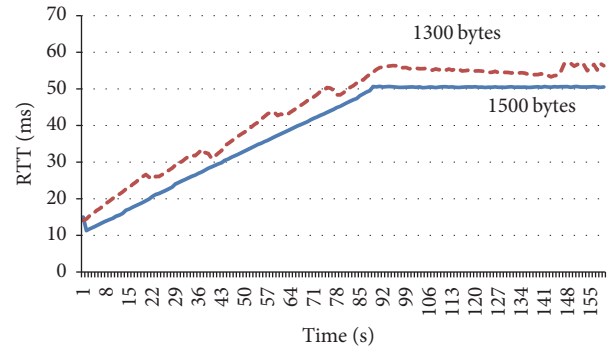


Figure 9: Relationship between UDTs RTT and packet size.

According to the information above, we can get that the real data packet size is 1500 bytes

5.1.2. Verify the Most Suitable MSS and Test the Influence of Different MSS to PaUDTs RTT. In order to check whether the packet size does affect the UDT's RTT and verify whether 1500 bytes is the most suitable MSS in our experiment's environment, we design experiments as follows. In this experiment, we need two computers in a local area network, and every computer is both the sender and the receiver. In order to distinguish them, we call one computer A and the other computer B. Before the experiment, we need to modify the packet size in the open code. In the experiment, we set the packet size to 1500 bytes, 1300 bytes, and 512 bytes, respectively.

When application begins to send data, we record the value of RTT in computer A and computer B. After we get these values, we draw two diagrams like Figures 9 and 10. In the figures, the dotted line represents the value of RTT when the

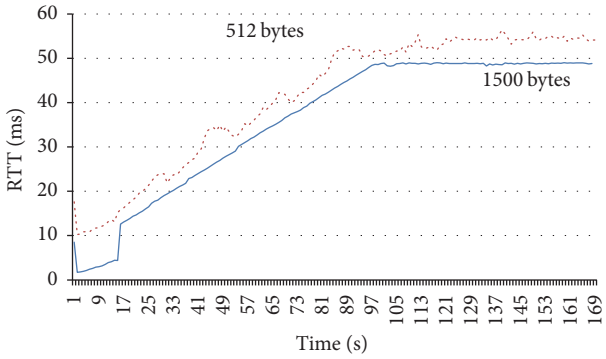


Figure 10: Relationship between PaUDTs RTT and packet size.

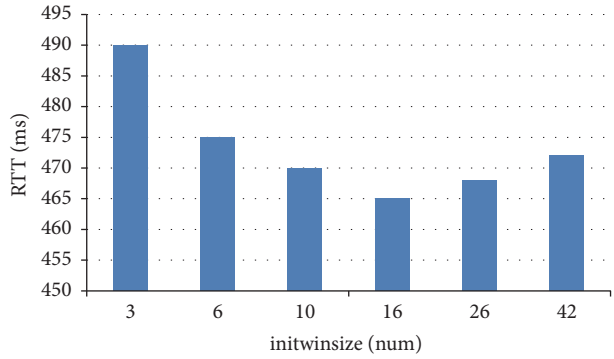


Figure 11: Relationship between initwinsize and latency.

MSS is 1300 bytes or 512 bytes and the solid line represents the RTT which is the result of MSS equaling 1500 bytes.

As shown in Figure 9, when MSS equals the most suitable MSS, the value of RTT is almost increasing linearly at the beginning, and when the time reaches 90, RTT's value stops increasing and the value keeps in range of 50 ms. Conversely, when MSS is equal to 1300 bytes, the value of RTT is not linearly increased. We can clearly see that there are some waves in the growth process. When the value of RTT stops increasing and tends to be stable, the line is not like a solid line. A relatively stable value is maintained, and the value of RTT is higher than 1500 bytes. Figure 10 shows the comparison curve between 1500 and 512 bytes. From the figure, we know obviously that the curve of 1500 bytes is more stable than the curve of 512 bytes and when MSS equals 1500 bytes, the average RTT is smaller than the MSS which equals 512 bytes.

So we can get a conclusion that the different MSS has a relationship to RTT, and when the MSS equals the most suitable MSS, the average RTT is the smallest and the efficiency of packets which are sent or received is the highest.

5.1.3. The Influence of the Size of the Initialize Windows to PaUDTs RTT. There are two variables in PaUDT's congestion control algorithm; one is MSS and the other is initwinsize. Section 5.1 introduces the influence of MSS to PaUDT's RTT, and this section will give some experiments to examine the relationship between initialize windows and PaUDT's RTT.

Before verifying the influence of initwinsize to PaUDT, we introduce the influence to TCP.

Higher initwinsize will offer an advantage for TCP which is reduced latency, as described in formula (12):

$$\left\lceil \log_{\gamma} \left(\frac{S(\gamma - 1)}{\text{initwinsize}} + 1 \right) \right\rceil \times \text{RTT} + \frac{S}{C}. \tag{16}$$

In formula (12), S represents the transfer size, C is bottleneck link-rate, γ is 1.5 or 2 depending on whether acknowledgments are delayed or not, and $S/\text{initwinsize} \geq 1$ [28]. As shown in formula (12), if we increase the initwinsize, the latency will be increased. Although the larger initwinsize can reduce latency, the oversize initwinsize may cause buffer overrun and packet drop, routers experiencing congestion, or congestion collapse.

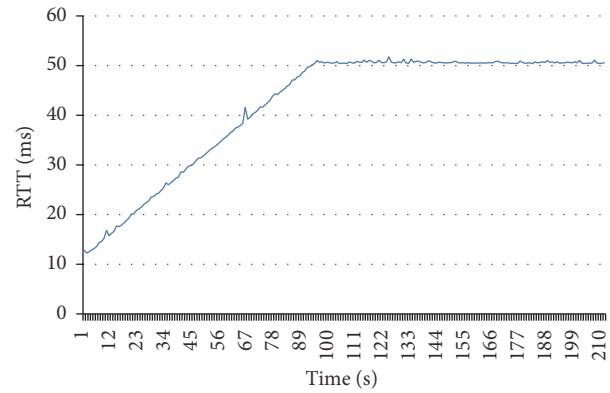


Figure 12: Relationship between PaUDT RTT and initialize window size (8).

Figure 11 shows the average TCP latency for Google Web search as initwinsize is varied from 3 to 42 segments, in a small scale experiment conducted concurrently on six servers in the same data center.

Google Web study shows the effect of initWinseize on TCPs latency and from Figure 11 we know that if the value of initwinsize is 16, the latency is minimum. Section 1 introduces that the congestion control algorithm of TCP begins from slow start algorithm like PaUDT, so the initialize window size also has influence on PaUDT. So we set the default value of initwinsize as 16 depending on the MSS in PaUDT.

In order to confirm the relationship between initwinsize and latency, we set experiments as follows. First, we modify the initwinsize to 8, 16, 32, and 64, respectively.

From Figures 12–14, we can see the difference of RTT as the change of initwinsize. As shown in Figures 12, 13, and 14, the distinction of RTT in different initialize window size is inconspicuous. The RTT value increased from 10 ms to 50 ms and remained stable. These three figures show that when the initwinsize is set to 16, the average of RTT's value is minimum. From the four figures, the most obvious change is Figure 15. When the initwinsize equals 64, the curve which describes the value of RTT causes a lot of volatility.

By analyzing the reasons for the dramatic changes in the RTT value, we found that the changes in intwize affect the value of CWnd. When the sender receives an ACK packet, the value of CWnd is calculated by formula (1). In conclusion,

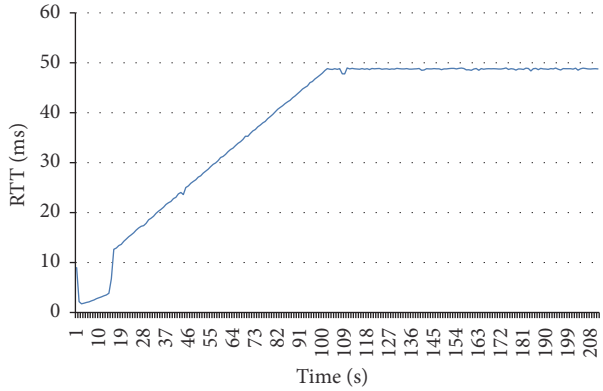


Figure 13: Relationship between PaUDT RTT and initialize window size (16).

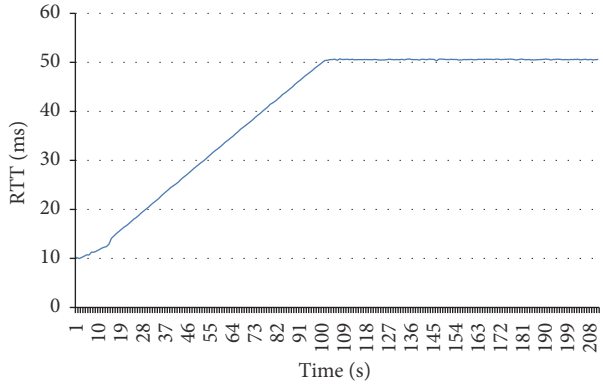


Figure 14: Relationship between PaUDT RTT and initialize window size (32).

initwinsize is bigger, and the value of CWnd is greater, and the excessive CMND may lead to new network congestion.

From Figures 12, 13, 14, and 15, we confirm that the initwinsize also influences the PaUDT like TCP.

6. Conclusion

Fog computing concentrates data, data processing, and applications on devices at the edge of the network instead of storing them almost entirely in the cloud, as does cloud computing. With the expansion of the network scale, the traditional C/S model will increase the burden on the central server, thereby reducing the data transmission efficiency. In this paper, We propose an improved PaUDT protocol with combining the PaUDT and P2P networks. We discussed a number of factors in congestion control algorithm which influence the RTT of PaUDT, applied in IoT-Cloud. In the future, we continue to research the congestion control algorithm of PaUDT and analyze the most important role that parameters play which influences RTT of PaUDT. C/S model increases the load on the server and increases RTT.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

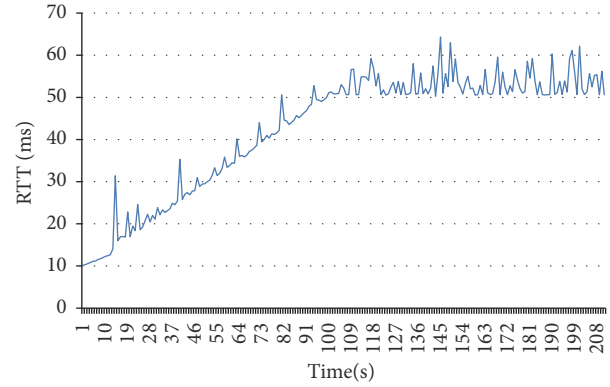


Figure 15: Relationship between PaUDT RTT and initialize window size (64).

Acknowledgments

This work is supported by National Natural Science Foundation of China (6167220961701170), China Postdoctoral Science Foundation funded project (2014M560439), Jiangsu Planned Projects for Postdoctoral Research Funds (1302084B), Scientific & Technological Support Project of Jiangsu Province (BE2016185), and Jiangsu High Technology Research Key Laboratory for Wireless Sensor Networks Foundation (no. WSNLBKF201701)

References

[1] I. Lee and K. Lee, "The Internet of Things (IoT): applications, investments, and challenges for enterprises," *Business Horizons*, vol. 58, no. 4, pp. 431–440, 2015.

[2] F. Xiao, Z. Wang, N. Ye, R. Wang, and X.-Y. Li, "One more tag enables fine-grained RFID localization and tracking," *IEEE/ACM Transactions on Networking*, vol. 26, no. 1, pp. 161–174, 2018.

[3] H. Zhu, F. Xiao, L. Sun, R. Wang, and P. Yang, "R-TTWD: robust device-free through-the-wall detection of moving human with WiFi," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 5, pp. 1090–1103, 2017.

[4] 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.

[5] 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 '12)*, pp. 13–16, August 2012.

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

[7] M. Duke, R. Braden, W. Eddy, E. Blanton, and A. Zimmermann, "A roadmap for transmission control protocol (TCP) specification documents," RFC Editor RFC7414, 2015.

[8] Y. Gu and R. L. Grossman, "UDT: UDP-based data transfer for high-speed wide area networks," *Computer Networks*, vol. 51, no. 7, pp. 1777–1799, 2007.

[9] K. G. Jijimol and S. Renjith, "Load-balanced optimal client-server assignment for internet distributed systems," *International Journal of Applied Engineering Research*, vol. 10, no. 69, pp. 216–221, 2015.

[10] A. Roy, H. Zeng, J. Bagga, G. Porter, and A. C. Snoeren, "Inside the social network's (datacenter) network," *Computer Communication Review*, vol. 45, no. 5, pp. 123–137, 2015.

[11] E. Kurdoglu, Y. Liu, and Y. Wang, "Dealing with user heterogeneity in P2P multi-party video conferencing: layered distribution versus partitioned simulcast," *IEEE Transactions on Multimedia*, vol. 18, no. 1, pp. 90–101, 2016.

[12] O. Babaoglu, M. Marzolla, and M. Tamburini, "Design and implementation of a P2P cloud system," in *Proceedings of the 27th Annual ACM Symposium on Applied Computing (SAC '12)*, pp. 412–417, March 2012.

[13] R. Mahmud, R. Kotagiri, and R. Buyya, "Fog computing: a taxonomy, survey and future directions," in *Internet of Things*, pp. 103–130, Springer Singapore, 2018.

[14] F. Xiao, W. Liu, Z. Li, L. Chen, and R. Wang, "Noise-tolerant wireless sensor networks localization via multi-norms regularized matrix completion," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 3, pp. 2409–2419, 2018.

[15] J. Park, H. Jang, and G. Cho, "Congestion degree based available bandwidth estimation method for enhancement of UDT fairness," *Journal of the Institute of Electronics and Information Engineers*, vol. 52, no. 7, pp. 63–73, 2015.

[16] J.-S. Park, D.-S. An, and G.-H. Cho, "Available bandwidth estimation method adaptive to network traffic load considering fairness with UDT flows," in *Proceedings of the 5th International Conference on IT Convergence and Security (ICITCS '15)*, August 2015.

[17] K. T. Murata, P. Pavarangkoon, K. Yamamoto et al., "Multiple streams of UDT and HpFP protocols for high-bandwidth remote storage system in long fat network," in *Proceedings of the 7th IEEE Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON '16)*, pp. 1–6, October 2016.

[18] 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.

[19] F. Xiao, J. Chen, X. Xie, L. Gui, L. Sun, and R. Wang, "SEARE: a system for exercise activity recognition and quality evaluation based on green sensing," *IEEE Transactions on Emerging Topics in Computing*, pp. 1–10, 2018.

[20] D. V. Bernardo and D. B. Hoang, "End-to-end security methods for UDT data transmissions," *International Conference*, vol. 6485, pp. 383–393, 2010.

[21] W. Feng and P. Tinnakornsrisuphap, "The failure of TCP in high-performance computational grids," in *Proceedings of the ACM/IEEE SC 2000 Conference*, IEEE, Dallas, TX, USA, November 2000.

[22] T. V. Lakshman and U. Madhow, "The performance of TCP/IP for networks with high bandwidth-delay products and random loss," *IEEE/ACM Transactions on Networking*, vol. 5, no. 3, pp. 336–350, 1997.

[23] N. Shah, A. Ahmad, B. Nazir, and D. Qian, "A cross-layer approach for partition detection at overlay layer for structured P2P in MANETs," *Peer-to-Peer Networking and Applications*, vol. 9, no. 2, pp. 356–371, 2016.

[24] P. Maymounkov and D. Mazières, "Kademlia: a peer-to-peer information system based on the XOR metric," in *Peer-to-Peer Systems*, vol. 2429 of *Lecture Notes in Computer Science*, pp. 53–65, Springer-Verlag, 2002.

[25] M. Thornburgh, "Adobe's Secure Real-Time Media Flow Protocol," RFC Editor RFC7016, 2013.

[26] P. N. D. Bukh, "The art of computer systems performance analysis, techniques for experimental design, measurement," *Simulation and Modeling*, 1992.

[27] Y. Gu, X. Hong, and R. L. Grossman, "Experiences in design and implementation of a high performance transport protocol," in *Proceedings of the 2004 ACM/IEEE Conference on Supercomputing (SC '04)*, pp. 6–12, November 2004.

[28] N. Dukkipati, T. Refice, Y. Cheng et al., "An argument for increasing TCP's initial congestion window," *Computer Communication Review*, vol. 40, no. 3, pp. 26–33, 2010.

Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 5176569, 13 pages
https://doi.org/10.1155/2018/5176569

Research Article

Boundary Region Detection for Continuous Objects in Wireless Sensor Networks

Yaqiang Zhang[1], **Zhenhua Wang**[1], **Lin Meng**[2] **and Zhangbing Zhou**[1,3]

[1]*School of Information Engineering, China University of Geosciences, Beijing, China*
[2]*College of Science and Engineering, Ritsumeikan University, Kyoto, Japan*
[3]*Computer Science Department, Telecom SudParis, Évry, France*

Correspondence should be addressed to Zhenhua Wang; wangzh@cugb.edu.cn

Received 26 January 2018; Accepted 1 April 2018; Published 9 May 2018

Academic Editor: Xuyun Zhang

Industrial Internet of Things has been widely used to facilitate disaster monitoring applications, such as liquid leakage and toxic gas detection. Since disasters are usually harmful to the environment, detecting accurate boundary regions for continuous objects in an energy-efficient and timely fashion is a long-standing research challenge. This article proposes a novel mechanism for continuous object boundary region detection in a fog computing environment, where sensing holes may exist in the deployed network region. Leveraging sensory data that have been gathered, interpolation algorithms have been applied to estimate sensory data at certain geographical locations, in order to estimate a more accurate boundary line. To examine whether estimated sensory data reflect that fact, mobile sensors are adopted to traverse these locations for gathering their sensory data, and the boundary region is calibrated accordingly. Experimental evaluation shows that this technique can generate a precise object boundary region with certain time constraints, and the network lifetime can be prolonged significantly.

1. Introduction

The concept of fog computing has attracted more and more attention nowadays, since fog computing promises to provide a relatively low latency and high-efficiency service [1, 2]. Different from cloud computing, fog computing is geographically closer to the network edge, where the majority of heterogeneous Internet of Things (IoT) devices are located. Consequently, applications can completely or, at least partially, be achieved upon edge nodes, and, hence, sensory data may not be required to be routed to the cloud in order to reduce the network traffic. One of the typical applications for fog computing in industrial IoT is disaster tracking and management [3–5], where toxic liquid like vitriol would cause soil acidification and, even worse, toxic gases like ammonia gas would be harmful to human body once leaked. In industrial applications like factory monitoring, sensors are deployed to monitor gas, liquid, or other kinds of dangerous objects. In this article, we concentrate on continuous object boundary detection, where traditional techniques have explored this topic [6–8]. As shown in Figure 1, wireless sensor networks (WSNs) serve as the foundation of IoT. In the context of fog computing, sensory data provided by sensor nodes are routed to contiguous fog nodes for further processing, and these sensory data are required to be routed to the cloud since fog nodes are weak in capacity to handle these data. Therefore, analyzing these data at the fog nodes in an efficient manner and, thus, detecting the boundary region of continuous objects in an accurate and (near) real-time manner are a promising research topic [9].

Energy efficiency is the key issue in IoT, since IoT smart things are mostly battery-powered, and they are hardly to be recharged due to the harsh working environment. Traditional techniques have been proposed, where an energy-efficient algorithm for boundary detection and monitoring (COBOM) is proposed for boundary detection by selecting a set of boundary nodes [10]. Each sensor node saves its neighbor nodes status by a bit array. The head sensor node suppresses sensory data of its neighbors and reports the aggregated data packet to the sink node. In a certain round of data reporting, a few representative boundary nodes would be selected to report their sensory data.

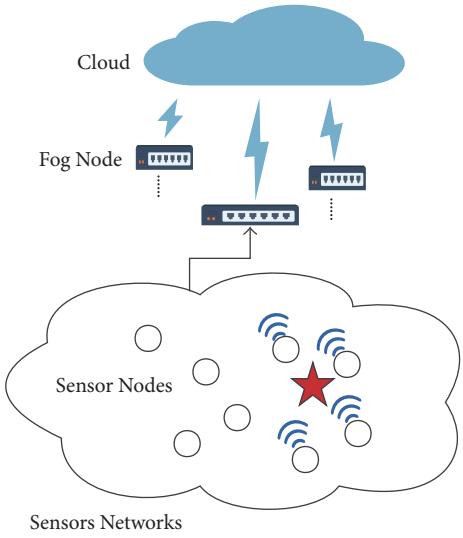


Figure 1: The network model in fog computing environment.

Sleep scheduling is applied to reduce energy consumption in sensor networks. In [11], the authors proposed a sleep scheduling scheme for object boundary monitoring. A few sensor nodes around the already existing boundary should be activated, while other nodes are stayed on the sleep mode. Generally, these schemes adopt partial of sensor nodes for data gathering purpose and, thus, for decreasing the network overhead. However, this strategy may lead to a coarse object boundary detection, since many sensor nodes may not report their sensory data to the sink to support the decision-making.

Research has been conducted on precise object boundary detection. In [12], the authors proposed a dynamic cluster structure for detecting and tracking a precise object boundary. Boundary nodes are dynamically organized into certain clusters. The head node in each cluster collects sensory data from other sensor nodes and sends these data to the sink node. More energy is consumed on data transmission so as to get a precise boundary. In [13], a data aggregation scheme is proposed to reduce the number of data packages among sensor nodes in the network. To detect and track a precise boundary, a piecewise quadratic polynomial interpolation scheme is applied. However, this technique does not check the predicted boundary as it may not be correct in some part of the predicted boundary.

To cover sensing holes and detect a precise boundary of continuous objects [14, 15], the authors proposed a mechanism through adopting mobile sensors for a precise and efficient boundary detection, where obstacles are not considered [16]. Generally, a few static sensor nodes are deployed, while mobile sensors are applied to detect sensing holes. In traditional schemes, object boundary shape is influenced by density and deployment of sensor nodes. This means that the more the sensor nodes are deployed, the more precise the boundary should be derived. Besides, the relatively large amount of sensory data should cause the network congestion. Note that a static sensor network could hardly generate a more precise object boundary when the object keeps relatively stable, since there may be very few sensor nodes deployed in the boundary region. In this proposed scheme, mobile sensors traversing along an interpolation-based estimated boundary would detect a more precise object boundary.

This article proposes an efficient mechanism for object boundary detection in fog computing environment. Mobile sensors collaborate to bypass sensing holes for generating a precise object boundary, while the energy efficiency is a main concern. The main contributions of this proposed mechanism are summarized as follows:

(i) Spatial interpolation algorithms are adopted to estimate the object boundary in the network region, where sensory data provided by static sensor nodes serve as the foundation. A threshold, which reflects the occurrence of potential events, is prescribed, and the possible object boundary is generated and represented as a curve. Potential points on the curve, which correspond to cruising stopping locations of mobile sensor nodes, are identified, in order to evaluate the applicability of the object boundary according to corresponding sensory data.

(ii) An energy balanced mechanism is developed to schedule the itineraries of mobile sensors. Specifically, the time of cruising time of mobile nodes is estimated through calculating the time required to visit all points in the curve. The number of mobile sensors is derived when considering the time constraint of certain applications. The itinerary scheduling of multiple mobile sensors can be reduced to a multiobjective optimization problem, which can be solved through adopting heuristic algorithms.

Extensive evaluations are conducted to evaluate the accuracy of object boundary and the performance of mobile sensors deployment technique. The results show that the proposed mechanism for object boundary region detection can effectively find a precise object boundary. Meanwhile, the utilization of mobile sensors can discover sensing holes with certain time constraints.

This article is organized as follows. Section 2 introduces the concepts and related techniques. Section 3 presents the network initialization mechanism. The precise boundary area detection mechanism is developed in Section 4. Section 5 evaluates the technique developed in this article. Section 6 reviews and discusses relevant techniques, and Section 7 concludes this article.

2. Preliminary

This section introduces the network model and defines relevant concepts. Algorithms including spatial interpolation and planar graph are presented.

2.1. Networks Model. Sensor nodes are deployed in a two-dimensional network region. There are two types of sensing mode of sensor nodes. Fixed sensing range means that the sensing radius of sensor nodes cannot be changed, while adjusted sensing range means otherwise. In this article, we

adopt a fixed sensing range for sensor nodes. We define that the sensing range for both types of sensor nodes are nonadjustable. Different from [17], we give the probabilistic sensing model, and the probability of a sensor detecting an event at a distance x is defined as follows:

$$P(x) = \begin{cases} 1 & \text{if } x \le r_s \\ 0 & \text{if } x > r_s, \end{cases} \tag{1}$$

where r_s represents the sensing radius of sensor nodes.

Generally, the communication radius of mobile nodes is larger than that of static sensors. The energy consumption of mobile nodes is mainly for their movement and communication process. To define a shape and a boundary for objects, a curve should be identified and corresponding to the threshold that defines the occurrence of a certain event.

Definition 1 (boundary line (BL)). It is a curve where the sensing value is equal to the prespecified threshold. BL represents the boundary of an object. An object is surrounded by BL.

In this article, static sensors are divided into different types according to their sensing attributes. A sensor node whose sensing value is higher than the threshold is called Inside Node (IN). A sensor node whose sensing value is lower than the threshold is called Outside Node (ON). BL separates the object and its surrounding.

As shown in Figure 2, all blue and black nodes that are located in the yellow object region have sensed the object, and they are INs. On the contrary, all the green nodes that are located on the outside of the object region are ONs.

Definition 2 (inside boundary node (IBN)). A node belongs to IN and there are both inside nodes and outside nodes on its one-hop neighbors.

Definition 3 (outside boundary node (OBN)). A node belongs to ON and there are both inside nodes and outside nodes on its one-hop neighbors.

All of IBNs belong to IBs, and all of OBNs belong to OBs. IBN and OBN are considered to be more close to the BL, because, around them, there exist different types of static sensors.

2.2. Spatial Interpolation. Spatial interpolation algorithms are widely used in geographic information systems like disaster risk analysis, agriculture, and event prediction particularly. To predict the condition of an unknown object, a spatial interpolation algorithm, namely, *Inverse Distance Weighted* (IDW), is applied in this article.

Assume that p represents a position which we want to acquire its estimated value $f(p)$, and $f(p)$ is influenced by all the points that exist a true sensing value. D_i is the distance between p and p_i and μ is a parameter that influences the result of interpolation; it is usually set to 2. The influence of p_i to p is inversely proportional to D_i:

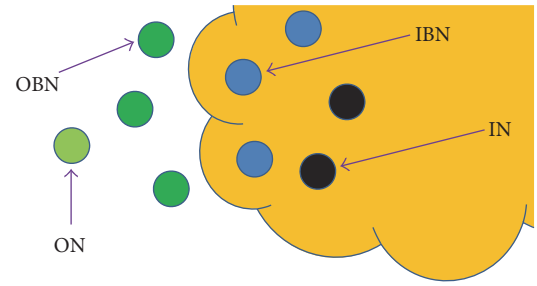


Figure 2: An example of static sensors distinction near the object boundary.

$$D_i = \sqrt{\left(x_p - x_{p_i}\right)^2 + \left(y_p - y_{p_i}\right)^2}, \tag{2}$$

$$f(p) = \frac{\sum_{i=1}^{n} \left(z_i / (D_i)^{\mu}\right)}{\sum_{i=1}^{n} \left(1 / (D_i)^{\mu}\right)}. \tag{3}$$

The main steps of *Inverse Distance Weighted* are listed:

(i) Calculate the distance D_i between the point p and all known points.

(ii) Calculate each point's weight $1/(D_i)^{\mu}$, which is the reciprocal of distance between p and p_i.

(iii) Calculate the predicted value of the point p according to formula (3).

2.3. Planar Graph. The planar graph is a kind of graph which consists of three parts: vertexes, edges, and faces. In the planar graph, each face is a closed polygon that consists of N vertexes ($N \ge 3$). In WSNs, the planar graph is usually applied to organize the sensor network, and there have developed many kinds of planarization algorithms. The *Gabriel Graph* (GG) and the *Relative Neighborhood Graph* (RNG) are two well-known kinds of the planar graph as presented in [18] and [19], respectively.

Figure 3 shows the rule of RNG and GG. In RNG, if a node w is located in the intersection region of circles about u and v with radius $d(u, v)$, then the edge $e(u, v)$ is deleted and it builds the edge $e(u, w)$, $e(v, w)$ separately. In GG, if a node w is located in the circle about the midpoint of u and v with diameter of $d(u, v)$, then the edge $e(u, v)$ is deleted and separately builds the edge $e(u, w)$, $e(v, w)$.

Based on the planar graph, we define the following.

Definition 4 (boundary face (BF)). A polygon area is divided by the planar graph which contains both IBN and OBN on its vertexes.

Definition 5 (boundary area (BA)). It is a continuous area that consists of all BFs.

An example is shown in Figure 4. The yellow area is the object. All blue and black nodes represent IN. All green nodes

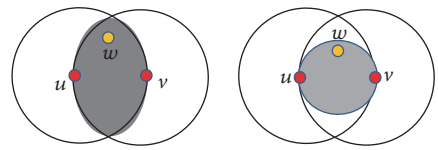


Figure 3: (a) shows the mechanism of generating the edge in RNG. If there exists a node w in the gray area, the edge between u and v will be broken and they connect with w , respectively. (b) shows the mechanism of generating the edge in GG. If there exists a node w in the gray area, the edge between u and v will be broken and they connect with w, respectively.

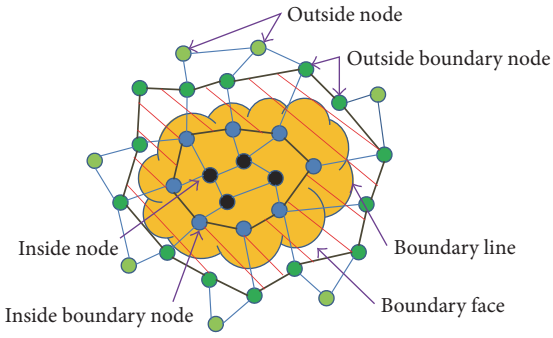


Figure 4: An example of object detection model.

represent ON. All polygon regions with both IBN marked by blue color and OBN marked by deep green color are named BF and they form the BA which is marked by red lines. BL is the curve that surrounds the yellow region.

3. Sensory Data Interpolation

In this section, the strategy for initializing the networks is presented. Planar graph algorithms are always applied to build topology networks for data transmission and network dividing. Spatial interpolation algorithms are applied to predict the object boundary.

3.1. Static Sensor Data Gathering. Static sensors are deployed in the network region at first. Once deployed, their positions are fixed. WSNs are often looked on as a self-organization network [20]. Static sensors communicate with their nearby nodes; then those sensors can transmit their position information to the base station leveraging network topology [21].

Static sensors are limited by energy storage. To reduce energy consumption, sensing time slot is introduced. At first time slot, static sensors start to sense their surroundings and transmit data to the base station. After that they stop sensing until it comes to the next sensing time slot.

Different from dense networks with so many sensors deployed in network region, this article works on a sparse network which has deployed less static sensors relatively. So there are some sensing holes in the region. In the proposed strategy, static sensors are on a small scale deployment and we gather all sensors' data at each time slot, instead of partly gathering sensory data as mentioned in many researches.

We do not focus on static sensors' data transmission strategy mainly because many researches have worked on data transmission routing problem. In this section, we apply planar graph algorithms for building a routing map for static sensors and divided the network region. Different from the grid dividing mechanism in [22, 23], the network in this article is divided into many small faces. In each face region, edge connects two nodes within a one-hop communication range. Each sensor in this map could transmit its position information data and sensory data by multihop network to base station.

3.2. IDW Spatial Interpolation Mechanism. Once all sensors transmit their sensory data to the base station, then the data will be analyzed in following steps:

(i) Based on planar graphs, a coarse boundary region BA is generated by BFs. The whole network region is divided into many subregions and static sensors are the vertexes of subregions according to planar graphs. Then BFs which have both IBNs and OBNs on their vertexes are selected and BA consists of all those BFs.

(ii) Based on the real data sensed by all static sensors, we apply *Inverse Distance Weighted* to generate a predicted value for each point that has no real sensing data. According to the interpolation result, we generate a curve called predicted BL where all the interpolation values are equal to the threshold.

4. Routing for Mobile Sensors

It is thought that sensing holes exist in the network because of the sparse static sensors deployment. After a predicted BL is generated, to reflect the truth in the curve and make a more precise object boundary area, mobile sensors are applied to round the curve. In this section, we propose a strategy of stops selection at first. The mobile sensors deployment and routing strategy are presented then. Finally, the object boundary area precision is discussed.

4.1. Stop Stations Selection. Based on a predicted BL, we try to get the real data on it. Traversing and sensing the whole predicted BL are unpractical, and it is a common sense that trying to traverse some representative positions is effective. Mobile sensors are also with the ability of sensing and a limited sensing range. In the proposed strategy, assume that sensing range for mobile sensors is R. To find some suitable stops that could mainly reflect the true situation of the curve, the distance between two stops should be appropriate. We

Require:
(i) D_i: Distance between a stop station s_i and first stop s_1.
(ii) R: sensing range of mobile sensors.
(iii) L: The predicted BL where a series of stops would be selected
Ensure:
(i) A series of stop stations on curve L.
(1) Randomly choose the first stop station s_1.
(2) Randomly choose the second stop station s_2 from a point where points are in a $0.9R$ to R range from s_1.
(3) initial $D_i = 2R$
(4) initial $i = 3$
(5) **while** $D_{i-1} > R$ **do**
(6) find a point set PS_i where points are in a $0.9R$ to R range from s_{i-1}.
(7) **for** $j = 2$ to $(i - 1)$ **do**
(8) $s\mathrm{PS}_i$ = a subset of PS_i where $D_{ij} > R$
(9) **end for**
(10) Randomly choose a stop station s_i from selected $s\mathrm{PS}_i$
(11) **end while**
(12) remove the stops which are so close to the already existed stops or static sensors from the stops set.

Algorithm 1: Stop station selection.

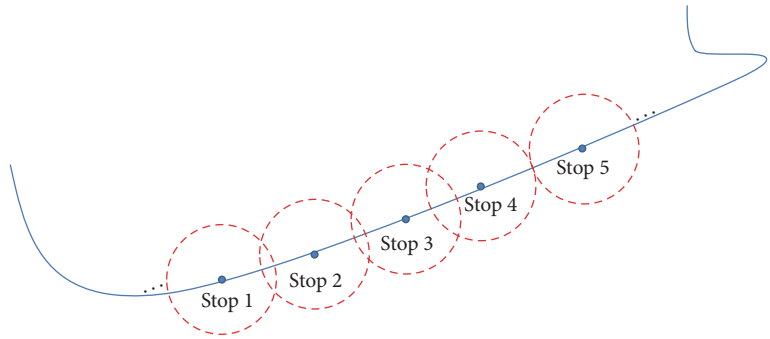


Figure 5: An example of stop selection. Each stop is selected and there is a suitable distance range between two adjacent stops.

suggest the distance between two adjacent stop stations is within the scope of $0.9R$ and R. As shown in Figure 5, it guarantees that the curve is covered completely. And a suitable distance can also reduce overlap.

Algorithm 1 is proposed to select stop stations. At first, an initial stop point s_1 is randomly selected in L, and the second point s_2 which is in $0.9R$ to R range far from s_1 is chosen (lines 1-2). Then a stop position s_i is picked from the point set PS_i generated according to s_{i-1} (lines 6–10). PS_i is a point set in which points are in $0.9R$ to R range far from s_{i-1}. After selection from PS_i, some points remain which are not so close to already existing stop points ($D_{i,j} > R$) (lines 7–9). Then it randomly chooses a point remaining in PS_i (line 10). When a set of stop points have been selected, then it checks whether a stop is within the sensing range of another point or sensors where sensory data exists already (line 12).

4.2. Mobile Sensor Determination and Routing Schemes. In this part, we discuss how to deploy mobile sensors and optimize the routing for them. In some applications, not only energy efficiency, but also the time constraints are taken into consideration.

4.2.1. Mobile Sensor Determination. As a set of stop points have been selected, then we deploy mobile sensors to traverse all those points in order to get the real sensory data. Time constraints are taken into consideration firstly. We assume that time spending on sensing of a sensor's surroundings is instant compared with moving. So we only concern about the time consumption on mobile sensor's movement. Based on this assumption, we propose a simple method by deploying a certain number of mobile sensors to ensure that all of the stop stations could be traversed on time.

At first, an estimated distance D_t which links all of the stop stations is generated. D_t is generated by choosing a stop station and linking it with its nearest stop, then the next stop also links with its nearest stop until all stops have been linked together. Then based on the speed of mobile sensors V_m, an estimated total time $T_t = D_t/V_m$ is calculated. According to the time constraint T_c, we get a minimum number of mobile sensors $N_s = \lceil T_t/T_c \rceil$.

4.2.2. Mobile Sensor Routing Schemes. In terms of the number of mobile sensors, in this part, a mechanism based on a heuristic algorithm is proposed. The target of this proposed scheme is to find optimal routings for all mobile sensors.

Require:
(i) BA_C: Current object boundary area.
Ensure:
(i) BA_N: A new object boundary area.
(1) According to the data gathered by both static sensors and mobile sensors, interpolate to the BA_C.
(2) Applying Algorithm 1 to find a group of new stop stations.
(3) Deploying a suitable number of mobile sensors.
(4) Routing for each mobile sensor.
(5) mobile sensors move to stop stations and get the sensory data there.
(6) generating the BA_N based on planar algorithms.

Algorithm 2: Boundary area precision.

It requires that energy consumption and time consumption are similar for each mobile sensor. It can be formulated as a multiobjective optimization problem. Some parameters are represented as follows:

(i) D_i is the moving distance of a mobile sensor M_i.

(ii) N_{Stop} is the number of stop stations that a mobile sensor M_i would traverse.

(iii) $\text{EM}_i(D_i, v)$ is the energy consumption in movement in distance D_i with speed v. We assume that all those sensors have same movement ability.

(iv) $\text{ES}_i(\text{data}_S, N_{\text{Stop}})$ is the energy consumption in data transmission. data_S is the package of sensory data generated in a stop.

(v) T_i is time a mobile sensor would spend. $T_i = D_i/v$. The constraint is $T_i < T_c$.

(vi) $V(\text{EM}_i)$ represents the variance of energy consumption in a sensor's movement.

(vii) $V(\text{EC}_i)$ represents the variance of energy consumption in a sensor's sensing.

Output is a tuple which includes a series of sequence denoted as $\{squence_1, squence_2, \ldots, squence_i\}$ where $squence_i$ represents an ordered stop points array that a mobile sensor M_i would traverse.

The fitness function is defined as follows:

$$\text{fitness} = k1 * V\left(\text{EM}_i\right) + k2 * V\left(\text{ES}_i\right), \quad (4)$$

where $k1$ and $k2$ are weights of two subobjects. They can adjust the importance of two subobjects. By adjusting the value of $k1$ and $k2$, we can determine the importance of two subobjects.

To solve this multiobjective and multiconstrained optimization problem, heuristic algorithm like ACO can be applied to solve this kind of issues [24, 25]. ACO is an evolutionary algorithm inspired by ants for finding foods behavior. It simulates the process of biological evolution to find the optimal solution. We apply it to find an optimal stops array for each mobile sensor. Through adopting ACO technique, an approximately optimal $\{squence_1, squence_2, \ldots, squence_i\}$ is found.

Table 1: Parameter settings.

Parameter name	Value
Region size	500 m × 500 m
Number of static sensors (n)	100–500
Radius of object (r)	40 m–120 m
Velocity of mobile sensors (V)	5 m/s
Energy consumption in data gathering (ECS)	1 unit/stop
Energy consumption in moving (ECM)	20 units/m
Sensing range of sensors (R)	10 m
Range of distance between stops (d)	$0.5R$–$1.5R$

4.3. Object Boundary Area Precision. Mobile sensors traverse alone their own routings, and they gather the sensory data of all stops located in the predicted BL. The sensory data reflects the truth in stops and it may be different with their predicted values in some stops. With more and more stops being found, We use planar algorithms to redivide the networks again; then we could get a new BA. The faces which are closed to the BL could be smaller and thus the new BA could be smaller and the shape of new BA could be more precise to reflect the object boundary.

In order to reflect the true condition around the object boundary, the previous steps are reused to get a more and more precise BL. The details are represented in Algorithm 2. Firstly, it uses existing sensory data to interpolate in current BA (line 1). Then it selects a set of stop stations in the new BL according to Algorithm 1 and deploys some mobile sensors to traverse those stop stations (lines 2–5). Based on new sensory data gathered by mobile sensors, the planar algorithm is applied to divide the network region and a more precise BA is generated (line 6).

5. Implementation and Evaluation

The prototype has been implemented in a Java program and a hybrid WSN is constructed. Experimental settings and results are presented as follows.

5.1. Experimental Settings. The parameter settings for our experiments are presented in Table 1. There is a

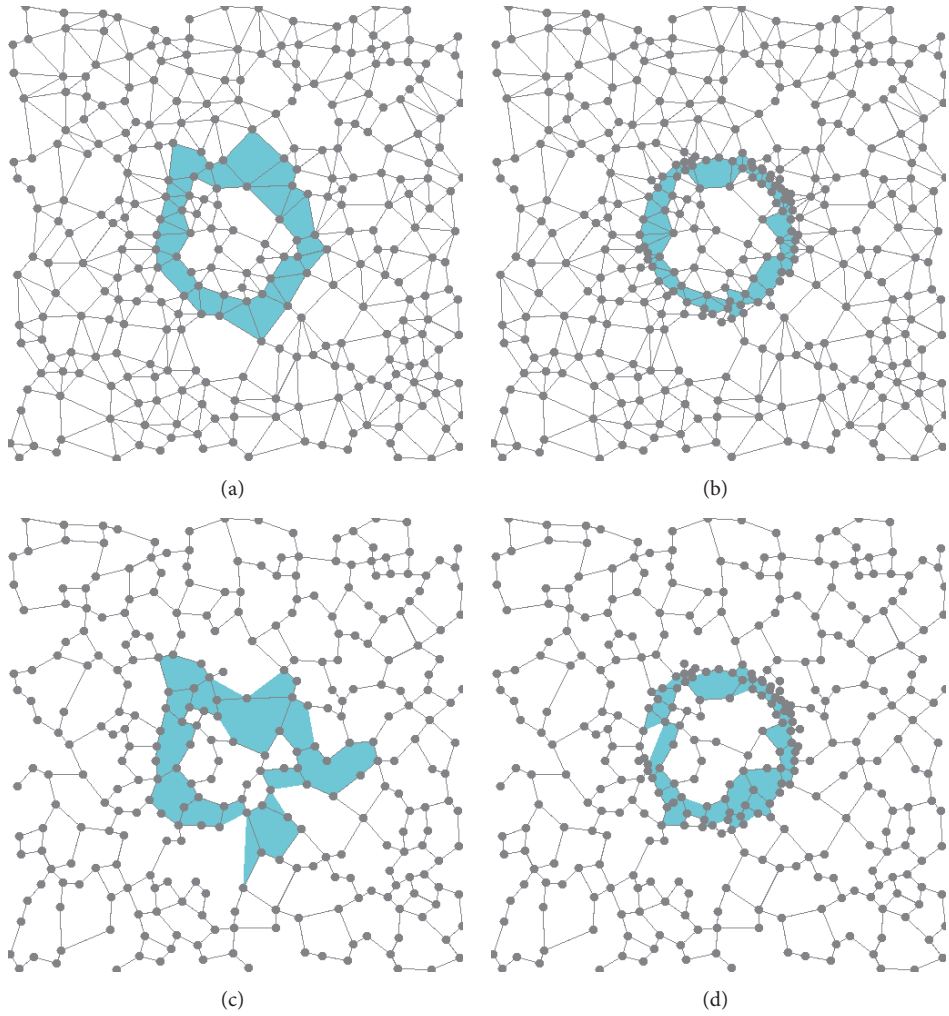


Figure 6: Object boundary regions of proposed mechanism. (a) shows the initial BA based on GG. (b) shows the BA after 10 time slots based on GG. The blue region in (c) shows the initial BA based on RNG and (d) shows the RNG-based BA after a period of time.

500 m × 500 m network region and the number of static sensors ranges from 100 to 500 in increments of 100. We assume that the object is a circle. And there are three different types of objects whose radius range from 40 m to 120 m increased by 40 m. The sensing value in the center of the object is 300 and the threshold of the object is set to 216.5. That means a position whose sensing value is higher than the threshold inside the object. In this section, the performance of the proposed mechanism is verified by different restricted conditions.

5.2. Boundary Area Precision Analysis. In this part, we discuss the precision of BA. The impact of different planar algorithms, density of static sensors, object sizes, and distance range between stops is taken into consideration in this paper. We analyze the impact of those four factors in this part.

5.2.1. Impact of Planar Algorithm. Two different planar algorithms GG and RNG are applied in this article. Figure 6 shows the variation of the shape of BA. There are 300 static sensors deployed in the area, and the radius of the object is 80 m. The blue area marks the BA of object. Compared with the initial network, the BAs with iterations are more regular and could more accurately reflect the real boundary region. And it is clear that GG and RNG result in different shape of BA. In fact, GG and RNG organize the network topology in different way, and the size and shape of their faces are different from each other. This results in a difference between the two algorithms.

5.2.2. Impact of Different Amount of Static Sensors. In this part, both GG and RNG are applied to initialize the network. And the number of static sensors n ranges from 100 to 300. Figures 7 and 8 show the variation of the size of BA and the number of stop stations. In Figure 7, with the increase of time slots, the size of BA is decreased in all of the curves. And after a period of time, the tendency of each curve holds steady. And with the increment of n, the size of BA is decreasing within GG and RNG, respectively.

Figure 8 shows the increment of stop stations. The number of stations increases but the growth rate decreases. The reason is that more and more selected stops are located in the predicted BL and they have covered the predicted BL

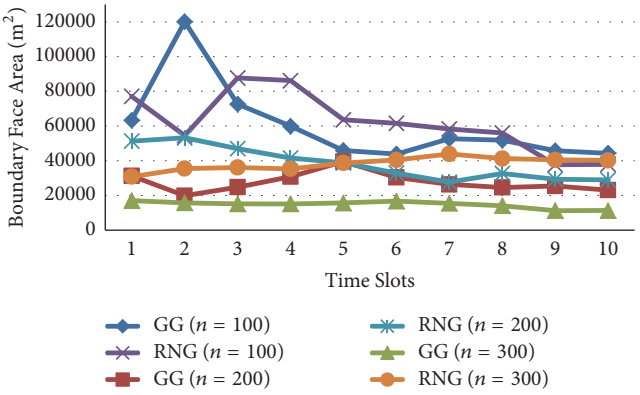


Figure 7: Comparison of the size of BA when different initializations and different n are applied.

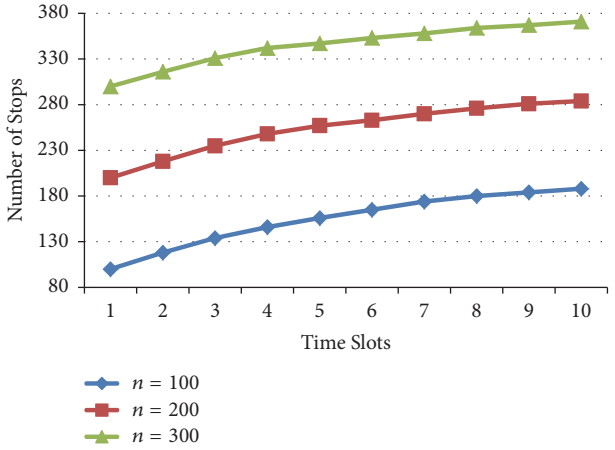


Figure 8: Comparison of the number of stop stations when different n are applied.

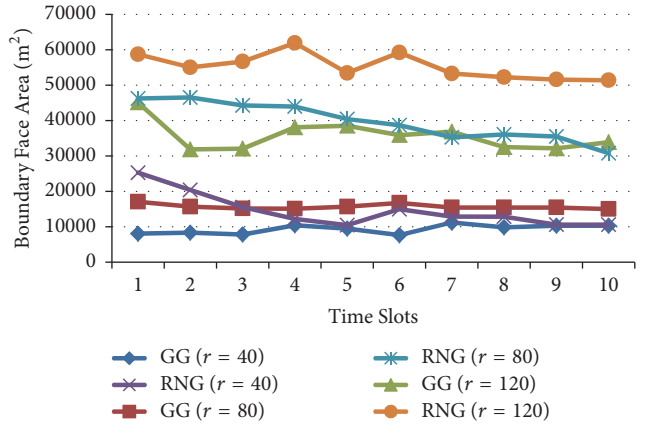


Figure 9: Comparison of the size of BA when different initializations and different r are applied.

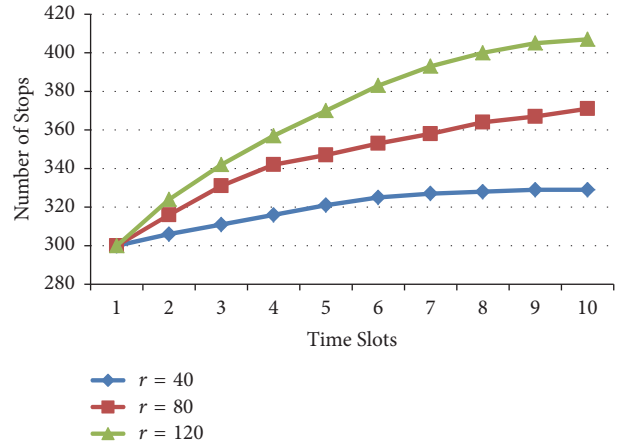


Figure 10: Comparison of the number of stop stations when different r are applied.

almost. In fact, when the number of stops is increasing, the size of boundary faces near the BL is increasing, and the number of boundary faces is decreasing. Finally, BL is covered by a narrow boundary region.

5.2.3. Impact of Object Size. Figures 9 and 10 show the simulation results. The radius of objects is set to 40 m, 80 m, and 120 m, respectively. Figure 9 shows the size of BA with the increment of time slots. Making a comparison under the same object size, GG and RNG perform differently in the BA's acreage. It is obvious that the size of BA when GG applied is mostly smaller than that of RNG in each time slot.

Figure 10 is the result of the increment of stop stations. The number of stop stations increases with the increasing of time slot. The growth rate is decreasing but growth rates are similar in all those curves. The scales of stop stations' number are different when the radius of objects are different. The smaller the object radius is, the less stop stations it selects in the predicted BL.

5.2.4. Impact of Distance between Stops. Figures 11 and 12 show the simulation results. The distance between stops ranges from about 0.5R to 1.5R in increments of 0.5R. In Figure 11, with the increment of time slot, the size of BA is decreasing in all of the curves. And after a period of time, the tendency of each curve holds steady. In GG, the range of BA's acreage is increasing when the distance between stops increases. In RNG, when the distance between two nearest stop stations is on a range of R, the range of BA's acreage is the smallest. It is obvious that different distance ranges between stops have an influence on the size of BA.

Figure 12 shows the number of stop stations. In each curves, the number of stops increases with the time going on. And the growth rate of stop stations number is decreasing. When the distance is on a range of 0.5R, the increasing of stops number are less than that of R and 1.5R. And the tendency of the increment of stops when the distance between two nearest stops is R or 1.5R are similar. In fact, if there exists many selected stops near BL, it is hard to select more stops because most of related region have been detected and there is no need to detect them again.

5.3. Boundary Precision Analysis. In this part, the precision of BL is shown and we compare the proposed mechanism with COBOM [12]. The object boundary precision is used

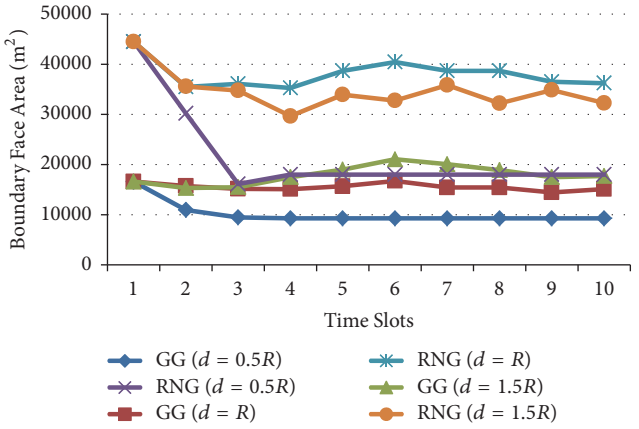


Figure 11: Comparison of the size of BA when different initializations and different d are applied.

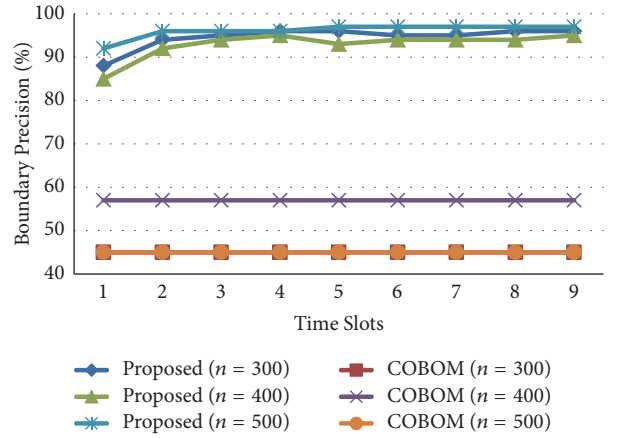


Figure 13: Comparison of the boundary precision for proposed mechanism and COBOM when n is set to 300, 400, and 500.

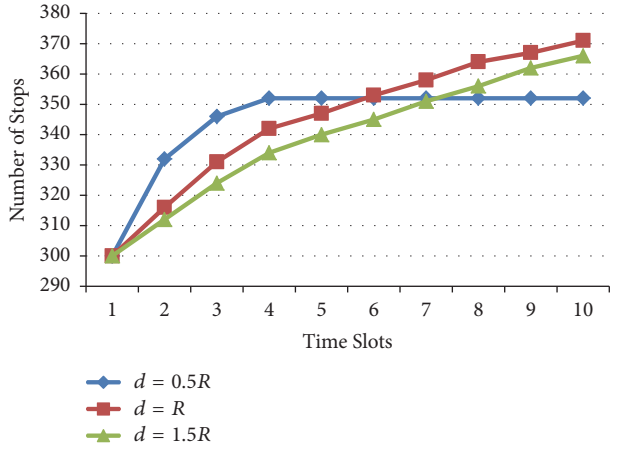


Figure 12: Comparison of the number of stop stations when different d are applied.

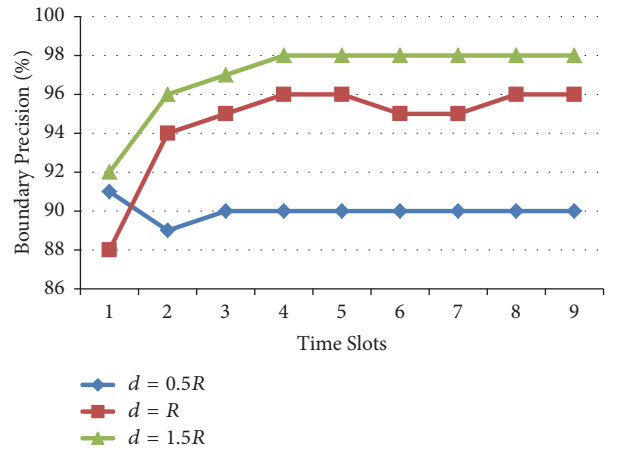


Figure 14: Comparison of the boundary precision when different d are applied.

to make comparison. The object boundary precision in this article is the proportion of boundary nodes which are close to the object boundary. If more boundary nodes are close to the real boundary, the boundary precision is higher.

Figure 13 shows the result of boundary precision under different density of node deployment. The proposed mechanism has a higher boundary precision than that of COBOM because the mechanism in this article can get the sensory data from sensing holes. Mobile sensors are deployed to move to the area without static sensors. COBOM only allows a small set of sensors to report their data; however, the network is sparse and a representative node may be far from the real boundary. The curves of COBOM have no changes. It is because the object boundary is stable. In each time slot, the status of boundary nodes is not changed. So the boundary precision is not changed once the boundary nodes are selected in COBOM.

Figure 14 shows the result of boundary precision with different distance between stops. We randomly deployed 300 sensor nodes in a 500 m × 500 m region. The distance d between two nearest stops ranges from $0.5R$ to $1.5R$. With the increment of time durations, the boundary precision keeps stable in all curves. And they are on a high level from 90% to 98%. In fact, a well coverage with stops could increase the accuracy of detection.

5.4. Mobile Sensor Performance Analysis. In this section, we analyze the performance of mobile sensors. After network initialization, we select some stop stations on the predicted BL. Assume that time constraint is T, that means we should collect all of the sensory data before T. We estimate a length L for traversing all those stop stations, and a predicted minimum number of mobile sensors is calculated based on the velocity of mobile sensors. Then this problem can be looked on as a multiple Traveling Salesman Problem, and ACO is applied to solve this problem; the objective function is to minimize the total length for all mobile sensors and to balance the length for each mobile sensor, thus balancing the time consumption.

The time constraint T is set to 50 s, and we have known that $V = 5$ m/s. According to T and V, we implement three different sizes of object about 40 m, 80 m, and 120 m. The

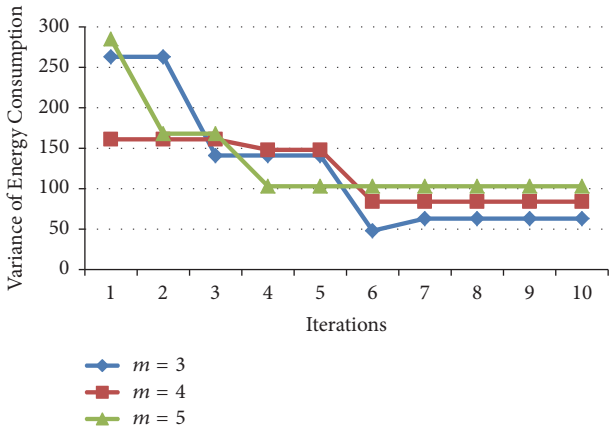


FIGURE 15: Comparison of the variance of energy consumption when m is set to 3, 4, and 5.

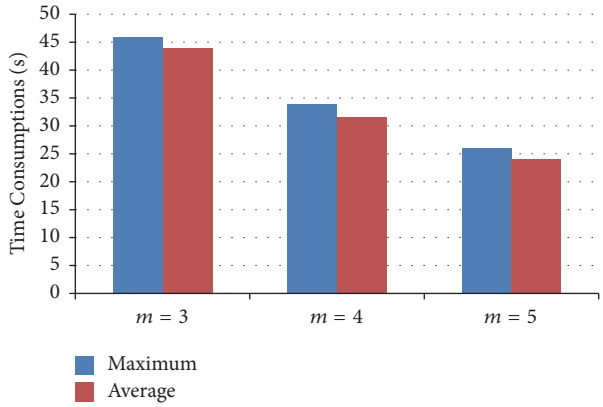


FIGURE 16: Comparison of the time when m is set to 3, 4, and 5.

number of mobile sensor m ranges from 1 to 5. Figures 15–20 show the results of implementations.

In Figures 15, 17, and 19, the changes of variance of energy consumption are shown. It is clear that the variance decreases and trends to a stable value. That means an optimistic scheduling for each mobile sensor is generated.

In Figures 16, 18, and 20, the maximum and average time consumption for mobile sensors are shown with different objects ranging from 40 m to 120 m. It is obvious that, according to the proposed scheme, there must be a suitable number of mobile sensors that could satisfy the time constraint T. With the increasing number of mobile sensors deployed in the network, the maximum and average time consumption would decrease.

6. Related Works

There has been long-term progress on object detection and tracking research in WSNs. Recently, fog computing framework has been more and more popular in IoT applications [26–29]. In [30], a mobile WSN is applied to intrusion detection, and fog computing is applied to improve the network performance. A three-tier model is proposed where

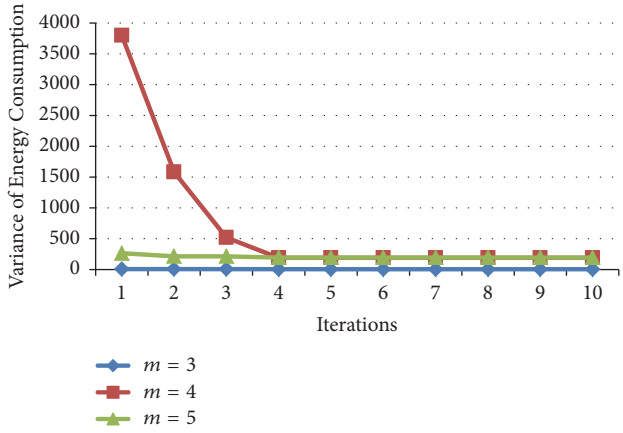


FIGURE 17: Comparison of the variance of energy consumption when m is set to 3, 4, and 5.

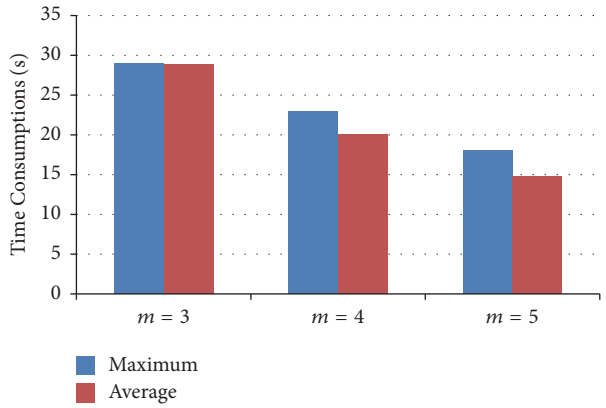


FIGURE 18: Comparison of the time when m is set to 3, 4, and 5.

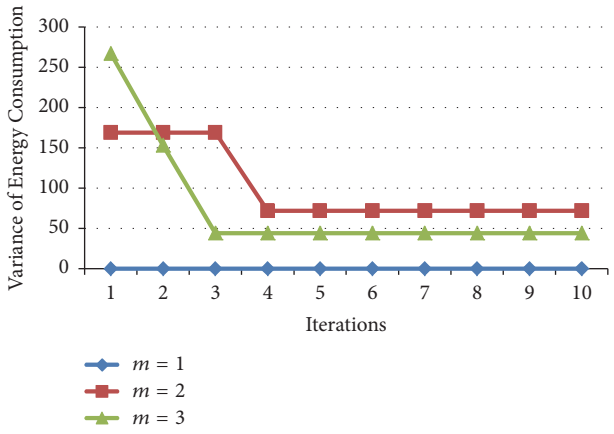


FIGURE 19: Comparison of the variance of energy consumption when m is set to 1, 2, and 3.

WSN is a bottom tier. Fog servers are located in the middle tier to manage the WSN and be connected to the cloud. Fog computing builds a flexible framework for gathering data. It is more close to the data source so that latency could be reduced.

In [31], a toxic gas boundary area detection scheme is proposed whose target is to detect the boundary area

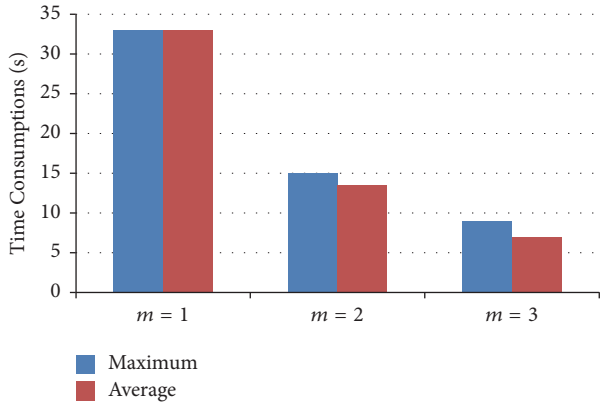


Figure 20: Comparison of the time when m is set to 1, 2, and 3.

rather than a single boundary line. Planarization algorithm is applied into the network region and it is divided into many small faces. Then based on inner-boundary node (IBN) and outer-boundary node (OBN), they identify a boundary area. IBN is a node where the reading of the toxic gas is higher than the alarm threshold. OBN is a node where the reading of the toxic gas is lower than the alarm threshold. The characteristic of boundary area is that it has both IBNs and OBNs on its vertexes. But it does not move forward to detect the holes inside the faces. In this article, we concentrate on detecting the sensing holes by mobile sensors according to a predicted boundary line and the shape of boundary area is more precise.

In [10], an energy-efficient algorithm for detection and monitoring object boundary called COBOM is proposed. They only choose a part of nodes to report to the sink node. If the current reading of a sensor is different from its previous reading, it broadcasts its reading and ID information. A node that receives the reading and information will store them in a BN-array. If its BN-array exists in different reading, it becomes a boundary node (BN). Only a part of BNs are selected to report message to the sink node, which are called representative nodes. Those representative nodes also suppress their neighbor nodes so that no significant information is discarded. In DEMOCO [32], if a node receives a message containing the same status, it ignores the message. If it receives a different message that is also different from at least one of its neighbor nodes, it is called a boundary node. Different back-off time is assigned to different boundary nodes, and if a boundary node is back early in short duration, it will suppress other nearby boundary nodes in sending messages. They are called representative nodes (RNs). Compared with COBOM, DEMOCO produces fewer BN and RN nodes and it performs better than COBOM.

Sleep scheduling mechanism is proposed to be an energy-efficient way in WSNs. In [11] an energy-efficiency scheme for monitoring a large-scale object is used. At first sensor nodes send message to their neighboring nodes. A node that receives a message including different status and has the shortest distance with the sender node is called boundary node (BN). Then master boundary nodes (MBNs) are selected from BNs based on the received event messages. MBNs are a the same concept as RN in COMBO and DEMOCO. Based on the above steps, it predicts next boundary by collaborating between MBNs. A boundary line is recognized by linked MBNs which is named the current boundary line. According to object diffusing, they draw a predicted next boundary. Sleeping sensor nodes in the next boundary zone waked up to send report. In this scheme, only the sensor nodes near the boundary area are activated to save energy. And next boundary is predicted to a precise object boundary detection.

The authors in [33] presented a sleep scheduling scheme for toxic gas monitoring. A sensor node inside the gas leakage area is selected as the zone head based on the largest residual energy by restricting flooding of the head selection message within the zone. CKN algorithm is applied to ensure global connectivity. Each awake node reports the sensing status and the grid centers covered by itself until the zone head. A awake node sends a wake-up message which contains its position, node ID, and the coordinates of the grid-points covered by itself to its one-hop sleep-neighbor. The sleep node is estimated and wakes up according to its neighbor nodes status.

In [9] a mobile element path planning for time-constrained data gathering scheme is proposed. Data produced by measuring needs to be delivered to predefined sink within a given time interval from the beginning of the measurement. Mobile elements travel alone a predefined path and collect data. Every node should be visited, and a node-disjoint scheduling scheme is presented for searching an approximately optimal path. In our previous work [34], a WSN region is divided into many small areas: each area should be visited by mobile sensors for gathering data. Multiple mobile sensors are deployed in network: each of them is responsible for a certain subregion, and heuristic algorithms are applied for routing.

In [35] ant colony optimization (ACO) is introduced to schedule mobile sensors in consumer home automation networks. The network is partitioned into several clusters and only one sensor node serves as a CH in each cluster. A node that has a relative higher residual energy is called CH. It forms a center of a cluster by broadcasting message to other nearby sensors and gathering their data. Authors in this work only applied one mobile sensor to traverse those CHs and collect data. The mobile sensor relieves excessive consumption of part nodes and prolongs the lifetime of the consumer home network. ACO is utilized to plan the mobility path for the mobile sensor. In our work, we apply more than one mobile sensor considering the timeliness.

7. Conclusion

In this article, we propose to detect object boundary region through applying mobile sensors. The network is divided by planar algorithms at first. An estimated object boundary is derived through applying the interpolation algorithm. To examine whether the boundary reflects that fact, candidate sensing locations are discovered and traversed by mobile nodes for gathering sensing data. The heuristic algorithm (i.e., ACO) is applied to generate optimal paths for mobile

sensors. Experimental results show that the proposed mechanism can get a precise object boundary region and can balance the energy consumption and time consumption for mobile sensors.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported partially by the National Natural Science Foundation of China (Grant no. 61772479) and by the Fundamental Research Funds for the Central Universities (China University of Geosciences (Beijing), China).

References

[1] L. Gao, T. H. Luan, S. Yu, W. Zhou, and B. Liu, "FogRoute: DTN-based data dissemination model in fog computing," *IEEE Internet of Things Journal*, vol. 4, no. 1, pp. 225–235, 2016.

[2] 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*, 2016.

[3] R. Lu, K. Heung, A. H. Lashkari, and A. A. Ghorbani, "A lightweight privacy-preserving data aggregation scheme for fog computing-enhanced IoT," *IEEE Access*, vol. 5, pp. 3302–3312, 2017.

[4] D. V. Manatakis and E. S. Manolakos, "Estimating the Spatiotemporal Evolution Characteristics of Diffusive Hazards Using Wireless Sensor Networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 9, pp. 2444–2458, 2015.

[5] T. Qiu, R. Qiao, and D. Wu, "EABS: an event-aware backpressure scheduling scheme for emergency internet of things," *IEEE Transactions on Mobile Computing*, 2017.

[6] W.-R. Chang, H.-T. Lin, and Z.-Z. Cheng, "CODA: a continuous object detection and tracking algorithm for wireless ad hoc sensor networks," in *Proceedings of the 5th IEEE Consumer Communications and Networking Conference (CCNC '08)*, pp. 168–174, January 2008.

[7] S. Park, H. Park, E. Lee, and S.-H. Kim, "Reliable and flexible detection of large-scale phenomena on wireless sensor networks," *IEEE Communications Letters*, vol. 16, no. 6, pp. 933–936, 2012.

[8] S. Imran and Y.-B. Ko, "A continuous object boundary detection and tracking scheme for failure-prone sensor networks," *Sensors*, vol. 17, no. 2, article 361, 2017.

[9] K. Almi'ani, A. Viglas, and L. Libman, "Mobile element path planning for time-constrained data gathering in wireless sensor networks," in *Proceedings of the 24th IEEE International Conference on Advanced Information Networking and Applications, AINA2010*, pp. 843–850, Australia, April 2010.

[10] C. Zhong and M. Worboys, "Energy-efficient continuous boundary monitoring in sensor networks," Tech. Rep., 2007.

[11] S. Park, S.-W. Hong, E. Lee, S.-H. Kim, and N. Crespi, "Large-scale mobile phenomena monitoring with energy-efficiency in wireless sensor networks," *Computer Networks*, vol. 81, pp. 116–135, 2015.

[12] X. Ji, H. Zha, J. J. Metzner, and G. Kesidis, "Dynamic cluster structure for object detection and tracking in wireless ad-hoc sensor networks," in *Proceedings of the 2004 IEEE International Conference on Communications*, pp. 3807–3811, fra, June 2004.

[13] H.-J. Lee, M. T. Soe, S. H. Chauhdary, S. Rhee, and M.-S. Park, "A data aggregation scheme for boundary detection and tracking of continuous objects in WSN," *Intelligent Automation and Soft Computing*, vol. 23, no. 1, pp. 135–147, 2017.

[14] Y. Zhang, X. Zhang, W. Fu, Z. Wang, and H. Liu, "HDRE: coverage hole detection with residual energy in wireless sensor networks," *Journal of Communications and Networks*, vol. 16, no. 5, pp. 493–501, 2014.

[15] F. Yan, A. Vergne, P. Martins, and L. Decreusefond, "Homology-based distributed coverage hole detection in wireless sensor networks," *IEEE/ACM Transactions on Networking*, vol. 23, no. 6, pp. 1705–1718, 2015.

[16] S. Das, I. Banerjee, and T. Samanta, "Sensor localization and obstacle boundary detection algorithm in WSN," in *Proceedings of the 3rd International Conference on Advances in Computing and Communications*, pp. 412–415, August 2013.

[17] K. M. Alam, J. Kamruzzaman, G. Karmakar, and M. Murshed, "Dynamic adjustment of sensing range for event coverage in wireless sensor networks," *Journal of Network and Computer Applications*, vol. 46, pp. 139–153, 2014.

[18] K. R. Gabriel and R. R. Sokal, "A new statistical approach to geographic variation analysis," *Systematic Zoology*, vol. 18, no. 3, pp. 259–278, 1969.

[19] G. T. Toussaint, "The relative neighbourhood graph of a finite planar set," *Pattern Recognition*, vol. 12, no. 4, pp. 261–268, 1980.

[20] T. Qiu, A. Zhao, F. Xia, W. Si, and D. O. Wu, "ROSE: Robustness Strategy for Scale-Free Wireless Sensor Networks," *IEEE/ACM Transactions on Networking*, vol. 25, no. 5, pp. 2944–2959, 2017.

[21] T. Qiu, R. Qiao, M. Han, A. K. Sangaiah, and I. Lee, "A Lifetime-Enhanced Data Collecting Scheme for the Internet of Things," *IEEE Communications Magazine*, vol. 55, no. 11, pp. 132–137, 2017.

[22] Z. Zhou, C. Du, L. Shu, G. Hancke, J. Niu, and H. Ning, "An Energy-Balanced Heuristic for Mobile Sink Scheduling in Hybrid WSNs," *IEEE Transactions on Industrial Informatics*, vol. 12, no. 1, pp. 28–40, 2016.

[23] S. Sharma, D. Puthal, S. Tazeen, M. Prasad, and A. Y. Zomaya, "MSGR: A Mode-Switched Grid-Based Sustainable Routing Protocol for Wireless Sensor Networks," *IEEE Access*, vol. 5, pp. 19864–19875, 2017.

[24] Y. Lin, J. Zhang, H. S.-H. Chung, W. H. Ip, Y. Li, and Y.-H. Shi, "An ant colony optimization approach for maximizing the lifetime of heterogeneous wireless sensor networks," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 42, no. 3, pp. 408–420, 2012.

[25] W. Liu, S. Li, F. Zhao, and A. Zhen, "An ant colony optimization algorithm for the multiple traveling salesmen problem," in *Proceedings of the 2009 4th IEEE Conference on Industrial Electronics and Applications, ICIEA 2009*, pp. 1533–1537, China, May 2009.

[26] A. M. Rahmani, T. N. Gia, B. Negash et al., "Exploiting smart e-Health gateways at the edge of healthcare Internet-of-Things: A fog computing approach," *Future Generation Computer Systems*, vol. 78, pp. 641–658, 2018.

[27] J. Liu, J. Li, L. Zhang et al., "Secure intelligent traffic light control using fog computing," *Future Generation Computer Systems*, vol. 78, pp. 817–824, 2018.

[28] B. R. Stojkoska, K. Trivodaliev, and D. Davcev, "Internet of things framework for home care systems," *Wireless Communications and Mobile Computing*, vol. 2017, Article ID 8323646, 2017.

[29] A. Yousefpour, G. Ishigaki, R. Gour, and J. P. Jue, "On Reducing iot Service Delay via Fog Offloading," *IEEE Internet of Things Journal*, 1 page.

[30] Q. Yaseen, F. Albalas, Y. Jararweh, and M. Al-Ayyoub, "A fog computing based system for selective forwarding detection in mobile wireless sensor networks," in *Proceedings of the 1st International Workshops on Foundations and Applications of Self-Systems, FAS-W 2016*, pp. 256–262, Germany, September 2016.

[31] L. Shu, M. Mukherjee, and X. Wu, "Toxic gas boundary area detection in large-scale petrochemical plants with industrial wireless sensor networks," *IEEE Communications Magazine*, vol. 54, no. 10, pp. 22–28, 2016.

[32] J.-H. Kim, K.-B. Kim, S. H. Chauhdary, W. Yang, and M.-S. Park, "DEMOCO: Energy-efficient detection and monitoring for continuous objects in wireless sensor networks," *IEICE Transactions on Communications*, vol. E91-B, no. 11, pp. 3648–3656, 2008.

[33] M. Mukherjee, L. Shu, L. Hu, G. P. Hancke, and C. Zhu, "Sleep Scheduling in Industrial Wireless Sensor Networks for Toxic Gas Monitoring," *IEEE Wireless Communications Magazine*, vol. 24, no. 4, pp. 106–112, 2017.

[34] Y. Zhang, Z. Zhou, D. Zhao, M. Barhamgi, and T. Rahman, "Graph-based mechanism for scheduling mobile sensors in time-sensitive WSNs applications," *IEEE Access*, vol. 5, pp. 1559–1569, 2017.

[35] J. Wang, J. Cao, B. Li, S. Lee, and R. S. Sherratt, "Bio-inspired ant colony optimization based clustering algorithm with mobile sinks for applications in consumer home automation networks," *IEEE Transactions on Consumer Electronics*, vol. 61, no. 4, pp. 438–444, 2015.

Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 7157192, 22 pages
https://doi.org/10.1155/2018/7157192

Review Article

Fog Computing: An Overview of Big IoT Data Analytics

Muhammad Rizwan Anawar,[1] **Shangguang Wang**,[1] **Muhammad Azam Zia**,[2] **Ahmer Khan Jadoon**,[1] **Umair Akram**,[3] **and Salman Raza**[1]

[1]*State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China*
[2]*Department of Computer Science, University of Agriculture Faisalabad, Faisalabad, Pakistan*
[3]*School of Economics and Management, Beijing University of Posts and Telecommunications, Beijing 100876, China*

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

Received 6 February 2018; Accepted 21 March 2018; Published 7 May 2018

Academic Editor: Xuyun Zhang

A huge amount of data, generated by Internet of Things (IoT), is growing up exponentially based on nonstop operational states. Those IoT devices are generating an avalanche of information that is disruptive for predictable data processing and analytics functionality, which is perfectly handled by the cloud before explosion growth of IoT. Fog computing structure confronts those disruptions, with powerful complement functionality of cloud framework, based on deployment of micro clouds (fog nodes) at proximity edge of data sources. Particularly big IoT data analytics by fog computing structure is on emerging phase and requires extensive research to produce more proficient knowledge and smart decisions. This survey summarizes the fog challenges and opportunities in the context of big IoT data analytics on fog networking. In addition, it emphasizes that the key characteristics in some proposed research works make the fog computing a suitable platform for new proliferating IoT devices, services, and applications. Most significant fog applications (e.g., health care monitoring, smart cities, connected vehicles, and smart grid) will be discussed here to create a well-organized green computing paradigm to support the next generation of IoT applications.

1. Introduction

Fog computing or fog networking, also known as fogging, is pushing frontiers of computing applications, data, and services away from centralized cloud to the logical stream of the network edge. Fog networking system works on to build the control, configuration, and management over the Internet backbone rather than the primarily control by network gateways and switches those which are embedded in the LTE network. We can illuminate the fog computing framework as highly virtualized computing infrastructure which provides hierarchical computing facilities with the help of edge server nodes. These fog nodes organize the wide applications and services to store and process the contents in close proximity of end users. Sometimes, fog computing used frequently and often interchangeably the term "edge computing." However, there is a little bit difference between those two concepts. Fog and edge computing both involve pushing the processing and intelligence capabilities down to the proximity where the information is originating. The main difference between both architectures is exactly where the computing and intelligence power is placed. In both structures data is sent by the same sources or physical assets, like pumps, relays, motors, sensors, and so on. All those devices perform a physical chore in this world such as electrical circuits, pumping water, switching, or sensing the task around them [1].

In past couple of years, the main idea was given to build mega data center architecture with the demand of centralized computing services called centralize cloud computing (CC) model (e.g., Google, Amazon, IBM, and Microsoft Azure). CC was moved all intelligently to the IoT but still a lot of processing power was done by the cloud server itself. Cloud was done with all data that creates some kind of files to put into some log files or in video files and others. So before fog computing architecture, moving information for data analytics, processing services, control systems, and data storage

into the centralized cloud had become a popular trend. With getting entrance into the new world, known as the world of technology, we have thousands of billion IoT devices which are connected to each other. With the passage of time IoT devices also have been proliferating with its explosive growth through the connection of physical things and operational components. However, if all those devices try to load whole computation at the cloud for just only functionality work, there would not be enough bandwidth to allow all these devices to communicate constantly with the cloud server [2]. Cloud servers themselves will get over loaded and there will be a big problem. Despite the power of the cloud structure, it is not applicable to facilitate operations that are time critical or cannot operate where Internet connectivity is poor. This has especially become true in time critical scenarios such as telemedicine and patient care, where milliseconds can have fatal consequences.

It is essential to make the concept of IoT clear. IoT is considered as a set of transformation period generated by the connected devices through over the world. So there is a question raised, that is, why we need to build that IoT? The answer is to dig up, access, and analyze the information that is previously available. All produced information in the environment formed analogues which cannot be a data. This information might be utilized in parking and might be placing a call at home. With connectivity of those things, those uniquely can be addressed and recognized with an IP address; the information can generate digital data, for example, communication of browsers, downloading of applications, and online transactions. The digitally translated data not only are more responsive but also opens up new analytical occasions in both technology and industrial fields.

At present, there is a mounting variety of powerful end users or devices like servers and smart access devices (e.g., smart phones, tablets, smart appliances in homes, cellular base stations, smart traffic direction polls along the roadside, connected smart vehicles, smart sensors, and controllers in smart power grid, smart building, and industrial control systems), which just are few names. Many more devices converge at industries specialization with the integration of information technology (IT) and operational technology (OT). Most of the use cases are established by industrial growth in particular sector (e.g., temperature sensors on a chemical vat and sensors that work at oil rigs stations). Thus, we can say that all these connecting things formulate the IoT. The ancestry of IoT goes back to several existing technologies like a machine to machine communication, RFID, and sensors.

According to Ovum and CISCO report, [3, 4] in March and June 2017, respectively, the main areas of investment are the deployment of IoT devices in industries (including manufacturing operations, transportation, smart grid technologies, smart buildings and increasingly, consumer IoT, and smart home automation). This is the main reason for raising the interest around the Industrial Internet of Things (IIoT) [3]. This situation is initiating a variety of new technologies and strategies to deal with, all the manufacturing and production related, rapidly rising amount of information and data at the core of IIoT. This is definitely the main idea which modernized the terms edge computing to fog computing by SISCO [5]. Fog applications might generate some real-time radio and network information that can offer a better analyzed experience to the user. This translated data (big IoT data) is not only much responsive but also opens up new extension of opportunities. Big data analytics from fog computing can help to glean insights and also enable devices to make smart and inelegance based decisions without human interaction.

Consequently, it is realistic and exciting to ask the question what should be done closer to the end users? Can a car turn into your key data storage? Can a smart home electrical device put together the different services that have been providing by other smart things, such as smart TV box, energy control boxes, and routers? What will happen if our smart phones give together the performance as radio network control functions without human interaction, those are executed by the LTE networks currently? Can smart devices at the edge become collectively enabled to support time-sensitive applications, like real-time data analysis at the edge, data mining of the transmitted formation, and many other industrial control functionalities? In the solution, fog computing exemplifies and also accelerates click to brick swings back from two directions: one is data plane and the other one is control plane [6].

In this manuscript, we have summed up the uniqueness that makes the fog compute towards the appropriate and significant platform that can deliver the quality of service (QoS) to significant connected IoT, particularly in future services and applications, such as connected vehicle, smart grid, and smart cities and in general wireless sensors and actuators networks. It is disruptive in several ways to handle all generated big IoT data or information which is explosively growing up. This survey will observe big data handling disruptions and concentrate on new aspects that IoT adds to big data from particularly distributed sources at the edge. We also will explain how big IoT data analytics is applicable to the industrial growth and can power the Internet of Things (IoT) for the industry [7].

Recently precise researches and achievements also would be discussed to meet above challenges in this manuscript. A transformational approach would be discussed, in Section 3.2.1, using the techniques of big data analytics based protected cluster management framework for optimized control plane at Software Define Network (SDN). Finally, ant colony algorithm is used to facilitate a big data analysis scheme for significantly improving the processing and accuracy of applications running in SDN. User mobility (e.g., from one node to another) makes QoS forecast and predicted values deviated from actual values in traditional cellular networks available. In Section 3.2.2, we will discuss an achievement in the context of service recommendation mathematical approach based on collective filtering which enables the QoS prediction for best user mobility. Smart connected vehicle is another required approach for fog enabled smart technology environment. A substantial accomplishment will inquire into Section 7.2.3, namely, virtual vehicle platform, a virtual vehicle coordination system based on the group coordination concept for sensing the environment information.

Specifically, virtual vehicle system proposes a discovery algorithm to find the optimal smart vehicle groups. Then, based on multigroup consent theory, they formulate a coordination algorithm to control virtual vehicle groups for multigroup synchronization among smart connected vehicles.

This work complements our whole work to present the prospective benefits of fog computing in terms of competency and low latency. This overview also focuses on enhanced programming models and applications for better and reliable deployment of fog computing structure. Some other architectural ideas for many upcoming systems and applications will be discussed such as cyber physical systems, IIoT structure, and embedded Artificial Intelligence (AI). Fog computing framework offers to explore structural designs, in this overview, which gives a particular concentration at large application domain of IoT situated at the edge of the fog computing structure deployment.

2. Fog Computing Overview

Data is increasingly generated by the proliferating IoT devices at the core network. Consequently, an essential structure can efficiently process all data at the close proximity of IoT, to be needed. Previous work such as micro data center [8, 9] about cloudlets [10] and at fog computing roll in IoT [2] has been initiated to the community based traditional CC. This concept will not always be efficient for data processing when data are going to be produced close to the edge of the network.

This section provides an overview of fog definitions and understanding about fog computing infrastructure with some related projects of previously established studies. It will also point out that why fog computing is going to be more efficient than CC, as well as from other terms for quality computing services.

2.1. Definitions of Fog. Fog computing presents a hierarchical distributed architecture with support of the integration of technological components and services might be in near future like smart cities, smart grid system, connected vehicles, and smart homes [11]. For the security of smart future, the deployment of fog structural networks at the edge is essential, to perform intelligent computations, big data analysis for better predictive processing, and identification of strange and unsafe events [12]. Thus, fog computing definition should be cleared at this emerging phase of fog computing framework. Definitions from [13, 14] give an extended vision about fog computing but cannot point out the unique similarity with the cloud. The fundamental definition is required to define and compare all functionalities of fog with preexisting structure, so here comes our definition.

Fog computing is an architectural development of computing with a resource pool where one or more ubiquitous and decentralized node(s) are enabled for potentially cooperating and communicating with each other or in group transmission exclusively at the extreme edge level, rather than backed by cloud processing. Fog nodes process tasks without third-party interference and collaboratively provide computational flexibility, better communication, storage capacity, and much more additional new mart services in a hierarchical environment for rising number of devices, clients, or end users in its close proximity.

Figure 1 is showing that fog is a complemented computing structure of cloud for providing reliability in computing services to make sure that QoS is at the edge of the core network. Fog computing is also an extension of cloud services at the edge of IoT devices to compete the challenges of traditional CC. Nodes at the edge are focused on sensing the raw data collection, command, and control of IoT devices. In this example, edge level nodes could be operated at millisecond or neon second to avoid contamination and ensure safety. Cloud level nodes, in the example, are focusing on data compression, filtering, analyses, and transformation. Each of them has the capability of operating and computing as a cloud model. These micro cloudlets will provide some edge level analytics required for critical real-time processing. The higher level nodes will focus on aggregating data and turning that data into knowledge. Mobile network facilitators are providing the services of fog similarities and they can offer fog computing services with IaaS, PaaS, or SaaS platform to the enterprise businesses through already providing services at their service network or even through cell towers [15].

2.2. Similar Terms. There are some matchings or similar terms of fog computing, called cloud computing, cloudlet architecture, mobile edge computing, and so on, which need to be explained here as follows.

Cloud Computing (CC). CC is comprised of software and applications running on the central server having a big data center and built-in local network [16]. Let us visualize the different types of clouds: (i) Private cloud is provisioned and used by a single organization. (ii) Community cloud provides the functionality of specific use for community of specific users. (iii) Public cloud is known for functionality of centralized computing for open use. (iv) Hybrid cloud is composition of two or more distinctive cloud architectures (e.g., private, community, or public). One solution work is identified for hybrid cloud [17] to provide a combination of existing cloud computing architecture to make enormous IT solutions. (v) Virtual Private cloud alternatively operates for addressing issues related to public and private clouds. A cloud or centralized node which is available openly for general public is called a public cloud. While when we use the term, private cloud, it refers to internal data centers of an organization(s) those are not available for general public use. Hence, cloud computing is a sum of SaaS and utility computing (IaaS plus PaaS) [18]. Make availability of CC services for IoT structure is not an easy task, caused by challenges of synchronization, standardization, balancing, reliability, management, and enhancement [19].

Cloudlets. Cloudlets are also called micro cloud data centers, like a small cloud computing architecture which is inherited features from centralized CC [10]. Cloudlet emphasizes providing services related to time-sensitive and limited bandwidth used applications. Cloudlets computing structure seems to be essential, if we have a sight at previous work,

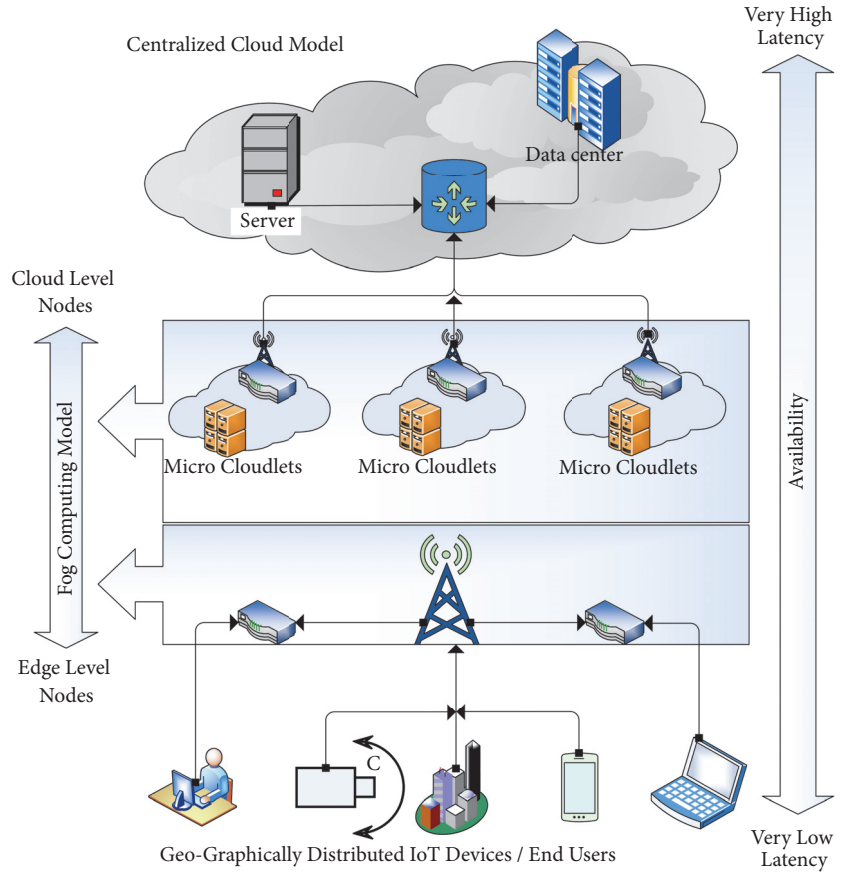


Figure 1: Fog computing structure view.

which showed mobile code offloading [8] and cost reduction structure of the central cloud [9]. Hence an additional difference between the enterprise and the cloud service data center environment can be explained tentatively.

Mobile Edge Computing (MEC). MEC is enabling technologies to make computational functionalities of CC for QoS at the edge of the network available. Mobile edge computing mentions the technologies to allowing computation at the edge. For example, micro cloud node can become an edge node between mobile devices and cloud, gateways could be the edge devices between home IoT and cloud, and a smart phone might be an edge device between body things and cloud [20]. MEC is primarily located at mobile network base stations [8] and sometimes called mobile cloud computing (MCC) architecture [21]. In MCC architecture, both data storage and data processing managed outside from mobile devices [21]. The reasoning of edge computing is that computation facilities should be deployed at the proximity where data is generated. From our point of view, mobile edge computing is substitutable with fog networking [6], but mobile edge computing has been focusing more on the things side, whereas fog computing is focusing more on the infrastructure side. Here, we are also considering that fog computing is a better arrangement in the form of edge computing, in the old sense. Edge computing drives the intelligence computing, communication capabilities, and processing power of an edge gateway or appliance directly into devices, such as programmable automation controllers [20].

2.3. Fog Interfaces with Cloud, IoT, and Other Fog Nodes. As defined before, fog computing expands the CC functionality with more elasticity to the edge level of the core network and shared same processing strategies and features (virtualization, multitendency, etc.) to make extendable nontrivial computation services. Some existing features of transformations make the fog computing more significant: (i) high-speed itinerant applications (e.g., smart connected vehicle and connected rail), (ii) low latency required surroundings applications, (iii) distributed control systems in large scale (e.g., smart grids, connected tracks, and STLS), and (iv) geologically distributed applications such as sensor networks to monitor the different environments.

Fog computing architecture allows processing, networking, and storage services to dynamically transfer at the fog node, cloud, and IoT continuum. However, the interfaces for fog to interact with the cloud, other fogs, and the things or users must facilitate flexibility and dynamic relocation of the computing, storage, and control functions among these different entities. It enabled well-situated end user assessment for fog computing services and also allowed capable and effectual QoS management.

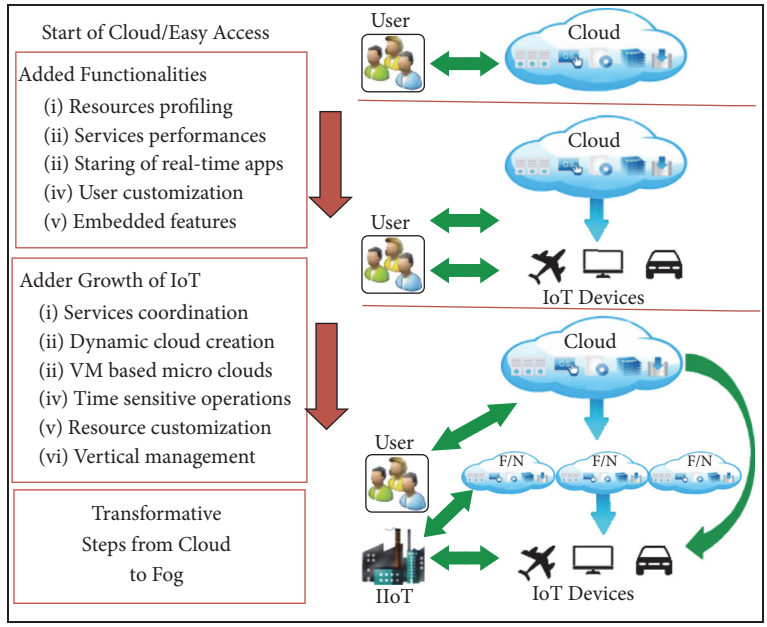


Figure 2: Transformative era form cloud to fog computing with interfaces every of them.

Fog to Cloud. From fog to cloud interface can be considered compulsory to support fog to cloud and cloud to fog collaboration which provides back-to-back services. It also supports functionalities, for example, (i) some functions at fog to be supervised or managed with the cloud computing ability, (ii) cloud and fog which can transfer data to each other for processing and comparing and for other required functionalities, (iii) cloud which can decide to distribute or schedule fog node(s) for allocation of the services on demand, (iv) cloud and fog both mutually which can differentiate for better management computing services, with each other, and (v) cloud which can make availability of its services through fog to the end users. It is essential to find out which information and services should be transversely passed at fog and cloud. Regularity and granularity of such data and information should decide how fog or cloud can response to that information or data.

Fog to Fog. Fog nodes must have pool resources functionality to support processing with each other. For example, all deployed fog nodes can share their data storage, computing, and processing capability tasks with prioritized node functionality system for one or several user applications. Multiple fog nodes might also act together with service for backups of each other.

Fog to IoT/End User. Fog computing provides its services to a widely distributed structure of IoT devices (e.g., smart devices and sensors) with the differential identification recognizing system. Fog to IoT interface or fog to user interface essentially needs to allow IoT access for fog services in user friendly environment and resource efficient and within secure ways.

Figure 2 explains about interfaces of fog with cloud and IoT both, through hierarchically distributed fog computing structure iteratively continuum. The right side of figure describes an emerging era of technology world from traditional cloud computing towards nearly deployed fog computing. It is also visualized that which type of interface is also included in different type of era (e.g., fog to cloud, fog to fog, and fog to IoT). In the left side of picture, it is pointed out that why we have a single and combined platform (fog computing) of these essential technologies. In the end, an important dispute or a question is raised that how all these interfaces and protocols to derive fog nodes functionality in existing or nearly distributed fog structure which will work together be designed. How the cost of required infrastructure could be reduced with providing high QoS to the end users or IoT.

3. Characteristics and Challenges

The main objectives of motivation from cloud to fog computing are extension of CC features to the edge level, which make CC services for increasing new variety of applications in IoT structure available. It also improves the QoS for them, for achieving low latency and high band width through better navigation services. A common uniqueness associated with fog computing is its hierarchical parallel processing operations at the edge of the core network [22, 23]. This implies that fog computing framework can become a nontrivial extended deployment of cloud computing. Hence, some important features are described in the following.

3.1. Characteristics of Fog. The major influences about the movement of a computational load from cloud to fog networking make an akin of CC at the proximity of edge. This procedure enables the development of new variety applications and services at IoT structure and defines the following characteristics of the fog computing.

(i) *Cognition*. Cognition is responsiveness to client centric objectives. Fog based data access and analytics give a better alert about customer requirements, best position handling for where to transmit, store, and control functions throughout cloud to the IoT continuum. Applications, due to close proximity, at end devices provide a better conscious and responsive reproduced customer requirement relation [6].

(ii) *Heterogeneity*. Fog computing is a virtualized structure so it offers computational, storage, and networking services between the main cloud and devices at the end. Its heterogeneity featured servers consist of hierarchical building blocks at distributed positions.

(iii) *Geographical Environment Distribution*. Fog computing environment has a widely circulated deployment in context to provide QoS for both mobiles and motionless end devices [24]. Fog network distributes geographically its nodes and sensors in the scenario of different phase environment, for example, temperature monitoring at chemical vat, weather monitoring sensors, STLS sensors, and health care monitoring system.

(iv) *Edge Location with Low Latency*. The coming out smart applications services are inadequate due to the lack of support at the proximity of devices with featured QoS at the edge of the core network. Video streaming in small TV support devices, monitoring sensors, live gaming applications, and many more are some examples of applications which required low latency services in its proximity [25].

(v) *Real-Time Interaction*. Real-time interaction is a variety and requirement of fog applications, like monitoring a critical process at oil rig with the fog edge devices or sensors, real-time transmission for traffic monitoring systems, electricity distribution monitoring system applications, and so on. Fog applications are having real-time processing capabilities for QoS rather than batch processing.

(vi) *Support for Mobility*. Mobility support is a vital fog computational advantage that can enable direct communication between mobile devices using SDN protocols (i.e., CISCO Locator/ID Separation Protocol) that decouples host identity from location identity with a dispersed indexing system [26].

(vii) *Large Scale Sensor Network*. Fog has a feature applicable when environment monitoring system, in near smart grid applications, inherently extends its monitoring systems caused by hierarchical computing and storage resource requirements.

(viii) *Widespread Wireless Access*. In this scenario wireless access protocols (WAP) and cellular mobile gateways can be classical examples as fog node proximity to the end users.

(ix) *Interoperable Technology*. Fog components must be able to work in interoperating environment to guarantee support for wide range of services like data streaming and real-time processing for best data analyses and predictive decisions.

These clarified characteristics enable new services and business models that can help to expand revenues, cost reduction, or speed up product rollouts in the industry and also have attractions for new investors in the context of fog structure deployment. Table 1 summarizes the differences of some key features between traditional cloud, deployed edge computing, and emerging fog networking structure functionalities and services that are provided to the end user or IoT devices.

3.2. Challenges and Opportunities of Fog. This section identifies and concentrates on some apparent issues in fog computing structure development. These following issues provided better understanding for the direction of future work.

3.2.1. Fog Networking (SDN and NFV). Fog network becomes heterogeneous, located at the edge of the network with extensions of CC functionalities. The responsibility of fog networking is to connect every required component at the node for maintaining and ensure QoS in core network connectivity and the provision of services upon all of those components. In the context of rising IoT used at large scale, this utilization might be not simple. Emerging technology, like SDN and NFV, should be planned to form elasticity and maintainability in such network environments. The deployment of SDN and NFV together can ease the achievement of increasing network scalability as well as cost reduction. It becomes very important in many use cases such as VM migrations, resource allocation, programmable interfaces, application aware control, and traffic monitoring [27].

To overcome the requirements of large scale network setups, the control plane is typically implemented as a distributed controller. Cluster management technology operates all types of proceedings and must continue a steady global network status, which significantly leads to big data in SDNs. Concurrently, the security of cluster technology becomes an open issue caused by the dynamic features of SDNs. To meet the above challenges, a big data analysis based secure cluster management framework for optimal control plane has been proposed recently [28]. An ant colony optimized method that uses big data analysis approach and the implementation system that optimizes the control plane are proposed. Ku et al. [29] have proposed a closest design in the context of SDN-based mobile edge cloud (fog nodes) architecture for QoS in mobile and vehicular ad hoc network (MANET and VANET). Table 2 also explains simulation results [29] that how a SDN-based network overtakes the other old-fashioned ad hoc networking protocols in packet delivery transfer ratio. The key feature of SDN-based approach responds faster. For example, SDN nodes update the SDN controller about the neighbor node's data information, and then, the SDN controller instantly detects the topology changes and also sends out the control messages as needed [29]. However, there are some questions raised like how to deal with retaining the connectivity graph of whole network in different distributed conditions and then node updating, how to assist

Table 1: Key featured difference between fog and cloud.

Features	FOG computing	Edge computing	Cloud computing
Availability of server nodes	Availability high range of servers	Less scalable than fog computing	Availability of few servers
Type of services	Distributed and localized limited and special for specific domain	Mostly uses in cellular mobile networks	Worldwide and global services
Location identification	Yes	Yes	No
Mobility features	Provided and fully supported	Provided and partially supported	Limited
Real-time interaction	Supported	Supported	Supported
Real-time response	Highest	Higher	Lower
Big data storage & duration	Short duration and targeted to specific area	Depends on the scenario of services and applications	Life time duration as its managing for big data
Big data analytic capacity and computation quality	Short time capacity with high level computation functionality	Short time capacity for prioritized computing facilities	Long time capacity only with categorization computing facilities
Working environment & positions	Streets, roadside, home, malls, field tracks (e.g., every Internet existing areas)	Deployed by the specific services provider in specific indoor areas	Indoors with massive components at cloud service provider owned place
Architectural design	Distributed	Distributed	Centralized
Number of users facilitated	Locally related fields (e.g., IIoT, STL devices)	Specific related fields (e.g., mobile users)	General Internet connected users
Major service provided	Cisco IOx, Intel	Cellular network companies	Google, Amazon, IBM, and Microsoft Azure

Table 2: SDN versus traditional ad hoc routing.

Node speed (m/s)	Packet delivery ratio of ad hoc networks versus SDN			
	SDN	OLSR	AODV	DSDV
5	90%	75%	73%	62%
10	83%	66%	72%	50%
15	75%	50%	60%	38%
20	51%	33%	52%	10%

with different controllers in fog network [30], and how to plan better connectivity through SDN system to encounter fog computing challenges such as mobility, latency, and scalability.

Alternatively, NFV changes the network functionality with virtual machine operational environment, but NFV is not intentionally approved in the framework of fog computing, until now. In the cellular core network the author [31] proposed functional placement problem on virtualized and SDN "rotten" gateway for minimizing the network transparency for achieving low latency and better utilization of bandwidth. Several types of research about NFV show how to propose better performance of virtualized software in the middle box platform [32]. NFV software has introduced ClickOS [33] as an addition in fog networking until the first concern for achievement QoS in network virtualization appliances. NFV helps to save both capital expenses (CAPEX) and operating expenses (OPEX). In context to address cost reduction challenges in dynamic fog computing resource allocation on behalf of delay sensitive mobile applications, NFV could enhance the fog architecture. Specifically, new elasticity is afforded by NFV in dynamic fog deployment rather than the current fixed location MEC practices. Yang et al. [34] formulate an optimized cost reduction problem for dynamic approach of near MEC (fog) services.

Table 3 is showing the deployment cost result about proceeding to assess the improvements provided by SCPA through flow level simulations within large network topologies. To this end, researchers [34] have compared the resultant MEC (fog nodes) operational cost of Heuristic+Reoptimization in contradiction of Heuristic and lower bound of best solutions (denoted by OPT_{LB}) under diverse network sizes, physical capacities, and latency requirements of NFV enabled nodes or servers [34].

3.2.2. Quality of Service (QoS). QoS discusses as an important issue in fog computing framework and also can be listed into four magnitudes, reliability, delay, connectivity, and capacity which are described as follows:

(i) *Reliability*. Reliability is one of key points for better and secure data transmission at core network. Madsen et al. [35] have reviewed the consistency requirement in the grid, clustering, and sensor networking towards a comprehensive research for reliable fog networking. Normally, reliability can be enhanced periodically through check points to resume after failure.

(ii) *Delay*. Latency sensitive applications are essential software architectural requirement in fog networking deployment for provision a real-time streaming and processing response. Spatiotemporal system

Table 3: Average cost for each network size over different latency requirement.

Number of nodes required (approx.)		
Heuristic	Heuristic and optimized	OPT_{LB}
20	20	15
35	25	20
50	40	30
60	45	40
80	60	45
90	70	50
95	75	65
100	80	70
105	85	75
130	100	80

architecture [36] proposed a time-sensitive event processing, based on fog to mitigate high latency data transmission. Another research work [37] has been described and experimented at application named RECEP, which exploits overlapping on data and cache imprecise results for reusing of computational resources and reduction requirement. It can extend scalability and minimizes the delay of mobile CEP systems.

(iii) *Connectivity*. Connectivity issue for fog networking framework must be solvable because fog network is made available opportunities of partitioning and clustering for cost reduction, data trimming, and expanding connectivity methods. Wang et al. [38] proposed a different approach from outdated service recommendation methods. They first have predicted QoS key teachers by reducing the impact of user locations mobility, instability in mobile networks, and instability of the similar services at diverse invoked times, although this approach expands service recommendation accuracy in MEC (fog).

(iv) *Capacity*. Capacity for QoS can be divided into two types, first is network bandwidth and second is storage capacity. those are very important aspects to enable and achieve high bandwidth and proficient storage operations. There are some similar works which need to be discussed. Research and experiment, at sensors operations in a network according to the deployment of MANET [39], present a big new challenge in the scenario of real-time response. Fog nodes may be required to process on information that is generating by chemical vat sensor to avoid disaster which is a critical real-time response requirement. Thus, processing will not start prior to finishing the aggregation of whole data, which absolutely adds delay. Due to dynamic data positions and large volume in fog computing, there are some requirements to redesign searching algorithms for running queries on dispersed content [40, 41]. It is not easy to redesign cache functionality at fog node to utilize sequential locality for saving maximum bandwidth and attain low latency; therefore two works have been proposed: the first utilizes the cache at end device [42] and the second recommends edge router cache utilization [43].

3.2.3. Computation Offloading. Computational offloading can mitigate resource constraint problem at fog networking and provide assistance in the performance of applications, storage capacity, and extended battery lifetime. Computation offloading can be separated into six types: schema, adaptation, granularity, communication, objectives, and distributed execution [44]. Despite an abundance of research work, computation offloading in the context of CC and cloudlets [8, 10, 45, 46] provides some directions to compete this challenge. We will review here a few of them. MAUI [8] has suggested a profile and code offloading methods for conclusions on adapting the change of network connectivity, latency, and bandwidth. Code analyzer [45] performs technically to migrate or merge the point in byte code. Thinkair [46] is driven on flexibility and scalability of the cloud to increase the processing and storage capacity of mobile edge cloud processing by parallelized method executions through multiple virtual machines (VM). COMET [47] leverages with a mutual distributed memory to an enhancement of smart phones and tablets communication with machines. The question remains that how to panelize applications for offloading and how to plane for dynamic offloading at fog structure, devices, network, and server nodes.

The availability of QoS in fog networking is a key feature during attacks or a failure of infrastructure that can be mitigated with VM migration technique. One of migration approaches is VM live migration, in which VM processes migrate from one physical node to another physical node [27]. VM live migration ensures the continuous hosted applications services. Therefore, VM migration can be a resource intensive approach. Network shared storage between two hosted nodes reduces the data transfer time to run the applications on moving VMs. However, a theoretical smart live migration approach is offered for VM migration [27]. This approach can estimate the downtime to determine and carry on the stop and copy stage during system failure on both hosted fog nodes. This also reduces the downtime and migration time to make sure of resources and QoS to the end users (see Figure 3). The expected downtime results will be compared with a previously defined downtime verge. This dynamic approach defines the next hosted fog node where to move into the stop and copy phase.

3.2.4. Resource Provisioning and Management. Resource and service availability are an extension of CC still attractive feature in fog computing environment. Key challenges that we faced here are bandwidth, distributed processing, storage, and latency. In a connected vehicle situation, we can take an example of an ambulance and programmable STLS to keep green traffic signal on. In this scenario, a pathway of ambulance remains open which is also warnings other vehicles to keep a clearance of road. MigCEP [48] has been

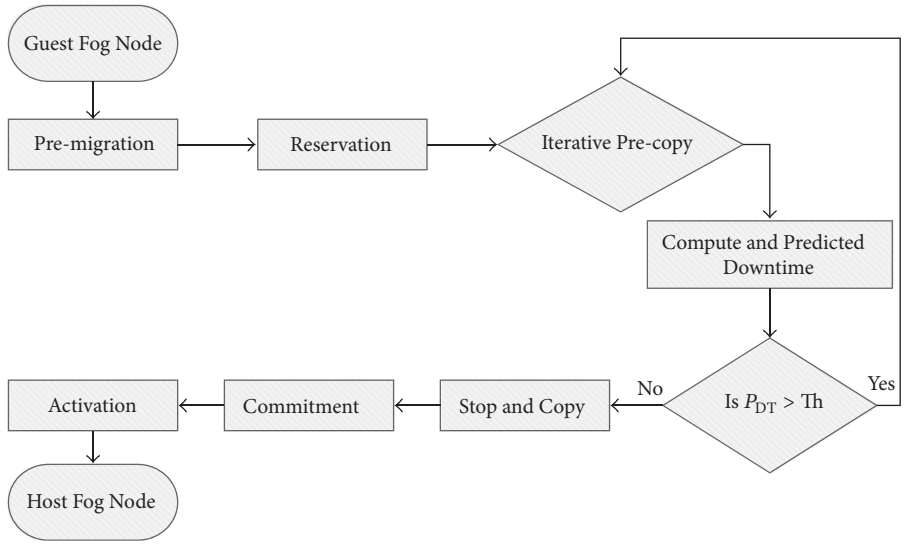


Figure 3: VM live migration between two physical nodes with shared storage.

proposed as an assignment and source sharing method for both cloud and fog computing structure. Resource sharing and coverage are critical situation for application processing in fog. A method has been described, which can centralize the flooding strategies dynamically to save energy in assorted networks [49]. Alternatively, Nishio et al. [50] planned a framework for varied resource sharing of mapping heterogeneous resources like availability of CPU, high bandwidth, and better storage on fog nodes structure. Resource sharing optimization challenges can be proposed by utilization of service-oriented functions. Fan also defined challenges and the solution [51] to measure the inclusive capability of information system to handle data including big data analytics, for effective information system as data storage, based on data analytics at the edge.

3.2.5. Security and Privacy. Security measurement in fog networking is directed by privacy, integrity, and availability. An idea is given, in which characteristically significant mechanism should be considered for designing and deployment of a fog structure [52]. While confidentiality and integrity are closely related to data privacy, availability entails the ability to remotely access resources offered by cloud servers and fog nodes when needed. Some challenges at fog inherited as cloud structure complemented deployment. Its various characteristics and deployment locations at the edge of the core network make it vulnerable. Prospective issues diminished with the deployment of fog computing [14] that are authentication, intrusion attack, security, and privacy. A designed work distributes the cloudlets as a mesh based intrusion recognition system for security of micro clouds (fog nodes) from interferences and network attacks [53]. Previously predicted ideas [13] about security issues that are currently related to a VM based environment might create probable security anxiety at fog devices. Zhanikeev [54] has recognized challenges linked with consistency requirements at hardware platform homogeneity for federation facility. Wang et al. [55] verified a man-in-the-middle attack, which can be averted with an authentic gateway prior to adding spiteful code into system. An overview is cleared separately about both security and privacy issues that are appropriate in currently and nearly use cases.

Security in Fog Computing. The hierarchical features of fog computing indicate some key issues of accessible huge number of IoTs by frontage fog devices. Security solutions that were planned for CC would not be directly applicable in fog structure. Working environment of fog computing may face security problems that do not exist in a usual cloud working structure. Authentication based on different gateways is identified as a major security issue of fog services [56]. Authorization in perspective of smart grids and machine to machine communication has been offered in recent work of Stojmenovic et al. [57], to alleviate some of the security issues like multicast authentication with public key infrastructure [58] and with use of intrusion detection system [59]. A decoy IT approach has been proposed [60] to withstand malware insiders by masquerade signals to avert threatening of sensitive information. Zuo et al. [61] suggested a ciphertext attack system in fog computing that presents a solution primarily to present the security model. A security model [22] also has been presented with reviewing possibility and nature of attacks. It is explained that the most important security issues should be categorized into whole network structure to facilitate and deliver QoS to distributed devices and communication components at fog network. Until that time, Denial of Service (DoS) and rogue gateway attacks were known as possible attacks on network transmission, while service data centers can expose to whole privilege escalation, physical damage, privacy leakages, and interruptions for QoS manipulation that has been identified as probable threats belonging to security [22].

Privacy in Fog Computing. Typically, public remote storage sensitive services and private cloud data centers could be compromised and leaked. Researchers mostly raised issues about the significant capability of legislation such as the PATRIOT Act at US based services provider [62–64]. Zhou et al. [65] have prolonged privacy protection of transited data for security of sensitive and important information. Their work ensures that anyone cannot access authorized user's private information. In contrast, fog computing is providing a hierarchical privacy hazard during deployment of fog components at the edge of the core network. Significant privacy hazards like data privacy, location privacy, and usage privacy deployed in fog nodes and application both [59]. Those nodes are much vulnerable to preventing of data thefts when compared with the centralized cloud. Dong et al. [66] have comprehensive research about sensor working in networks which are susceptible for content based privacy. He is proposing superfluous fog loops to defend the location identification of the content source node to make confusion for opponent. To alleviate spiteful eavesdropping during data transmission, it is offered [67] that a fog friendly structure through public key encryption will be converted in to irregular key updating for avoidance from high overhead. Lopez et al. [68] explained a quality based operation of secured middleware access for the better privacy of user data and prevented the service providers. Smart grid related privacy issues have been presented as safeguard schema proposal [69]. It also targeted the use of multidimensional data analytics approach based on homomorphism techniques in coding.

4. Fog Architectural Design and Implementations

We have disputed that location awareness applications are in danger due to the sense-process-actuate loop, as would be any other applications with feedback loops. In short, the cloud is too far from many IoT devices in the scenario of latency sensitive applications. Here, is small discussion listed about design and architecture implementation of fog computing framework.

4.1. Fog Frame Work and Architectures. Extension of computing, communication, and storage at the edge are described mathematically for simplified effective information system capacity as computation, data transmission, and storage [51]. To measure the comprehensive capability for a given information system, it can handle the involved data (including big data), for the effectiveness of information system as data capacity based processing for data analytics at the edge. Ha et al. [70] and Satyanarayanan et al. [71] both have designed comprehensively and obtained executed results in a real-time response through wearable smart devices assistance as Google Glass supported by cloudlet. CISCO related studies and information [26] explained an overview about a developing fog computing architecture with a three-layered approach of distributed AI services for better QoS at edge location of the network as well as distributed intelligence and predictable fog computing. Two systems can be discussed for the availability of resources near the edge of the network [72] that is explained for live sensor streaming analysis and cloudlets [10, 28] that are interactive for applications which are previously proposed and also will be discussed in next sections. A sequence in SDN [73, 74] is the basic structured property of a secured communications network in fog computing, having confidentiality, integrity, availability of information, authentication, and nonrepudiation. Programming and process as midlevel SDN structure for fog computing have been allowed [75–77], but these researches only allow and give a view about inserting routing logics, not basic application logics into network elements. Furthermore, SDN offers an expandable resource model for general principle applications. Implementation of a simple fog architecture with micro cloud nodes has been described, in between centralized cloud chains through end devices for health care smart sensor devices which are sensing and monitoring health care systems [78].

Existing work is paying attention to design and deployment of fog, such that, ParaDrop [79] and DECM [80] proposed fixing and solving extra energy consumption throughout the wireless data transportation by leveraging dynamic cloudlets model. Hong et al. refer to BOLO [81] which provides an essential abstraction to keep tracking for device performance and development of applications with intelligence processing pattern. It can help in fog computing for large scale and latency sensitive applications development at IoT structure. Bonomi et al. [2] previously and Chiang and Zhang [6] and Sarkar et al. [82] surveyed fog computing structural design which showed that fog computing is only the suitable platform for proposed applications in the context of fog computing framework, such as connected smart vehicles, smart grids, and smart cities. The importance of four layered fog computing architecture [77] has been described in detail which are lacking in current commercial CC models with smart pipe line monitoring examples through fiber optic sensor network. A research [35] has been presented in the field of signals and image processing which is proposing an analysis and showing how emerging of fog computing projects is challenged. It is clearly identified that an estimation and pricing model [83] for fog computing based micro datacenter might be deployed with the support of dynamic resources for IoT and end user devices.

Figure 4 is realizing hardware architectures as well as four-layer high level software architecture, deployed as a fog node. Hardware framework assorted physical components which contain servers, edge routers, access points, setup boxes, and scheduled storage devices with additional virtual image functionalities to support cost reduction and elasticity in fog computing architecture deployment. The platform is hosted by different OSs and software applications, thus having a wide range of software and hardware capabilities. A generic driver interface is provided by the abstraction layer of the software framework for managing the hardware such as storage, NIC, memory, and other types of hardware for seamlessly resourcing management and control [84].

With software architecture perception, fog computing nodes are highly virtualized with several VM APIs running

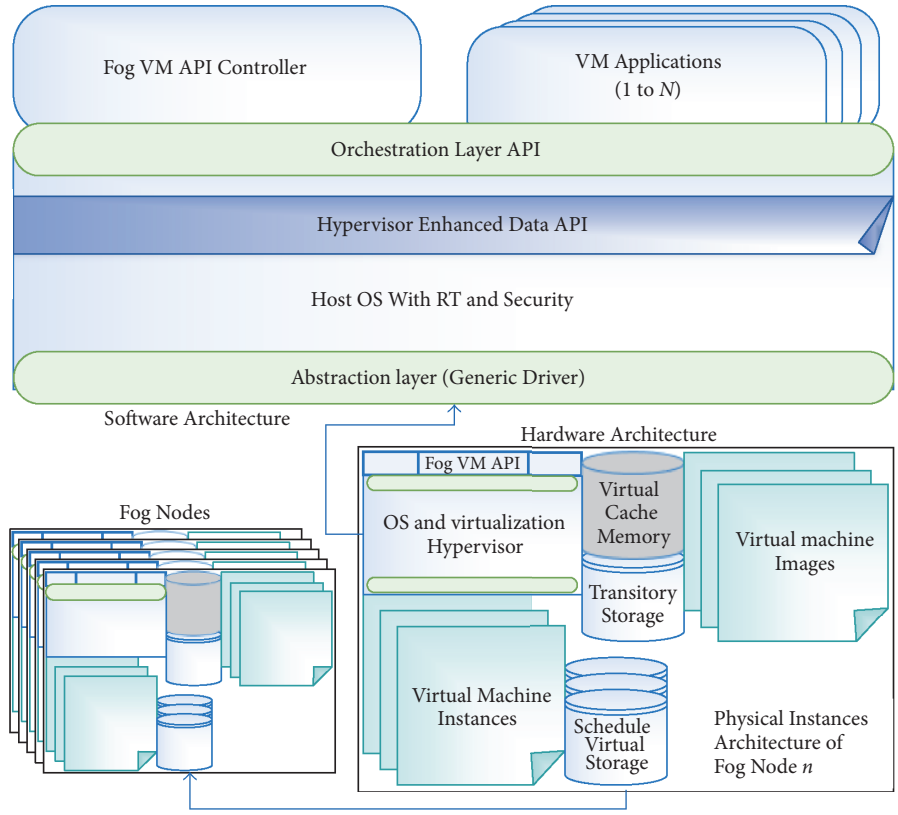


Figure 4: Fog computing software and hardware framework.

with support of highly proficient hypervisor enhanced applications. That hypervisor includes real-time enhancements and security to serve the needs of critical fog applications. This virtualized enhancement enables multiple OS to coexist on a single physical platform to make sure of efficient use of resources. The multitenancy application process shown on top ensures the segregation of different tenanted applications on the same node. They are carefully segregated from each other with the fog VM application controller and can communicate with functions via APIs only. The orchestration layer provides a policy based life cycle for managing fog services. This orchestration functionality provides a distributed approach as an underlying fog structure [84]. A hypervisor runs on top of all these APIs for providing a virtualized environment. The orchestration layer API achieves four basic targets, namely, mining and submission of data, analytics operations at submitted data, handling requests with scheduling, and preallocation of resources to implement at the predictive decision [85].

Certainly, the better deployment of fog based processing framework demands different significant algorithms and application work for dealing with consistency of the whole networks of cloud to the IoT at core network continuum.

4.2. Fog Proposed Application Taxonomy. Miscellaneous fog applications are recommended mostly as literature. In this section, we are reviewing a classification of those proposed application designs and implementation techniques. In the context of fog computing, previously it has been described that IoT smart devices might be used as fog nodes, and it might happen with your mobile phones or personal gateways [86]. Therefore, author [12] suggested a "Steiner" hierarchy idea based on caching schema, in which fog node servers primarily produced a Steiner hierarchy system to reduce whole path, load, and processing cost, for playing down the resource caching overheads. Extensive research works have been done [87, 88] at the use of fog computing services in existing health diagnosing and monitoring system, like that real-time fall detection which can imply mostly in health care real-time monitoring. Zhu et al. [26] also comprehensively organized fog computing for processing and transmitting video applications and smart services range in proxy assisted rate adaptation towards intellectual caching for on-demand video streaming. This feature improves the QoE virtually for in-hand interactive structural system of video surveillance cameras.

A vehicular ad hoc network (VANET) is designed with the complemented structure of SDN and fog computing to offer preprocessed low latency data transmitting [73]. Gazis et al. [89] have proposed a unique system as industrial perspective for organized fog computing framework to provide elasticity for individual operational processing requirements. Femtocloud by Habak et al. [90] has offered a dynamically self-configured mobile cloud with multicomponent by a clustered mobile devices structure to make availability of cloud services at the edge. The assessment recommended that

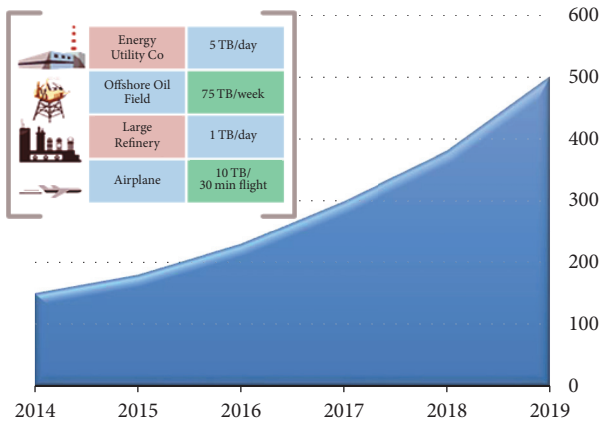


Figure 5: Data generated by worldwide connected users in billions.

this point of view offers a sensibly resourceful computational capability. In complex structures, the amount of data collection and analysis is increased exponentially, but the central cloud approach is also no longer sufficient.

Some extensions have been offered in the scenario of predefined smart meter structure for a flexible and cost efficient solution [91]. Tang et al. [77] described a hierarchically distributed fog computing structural design and application work which supports the integration immense number of mechanism and services in future of smart cities, for securing upcoming communities. This is essential to build a huge geospatial sensing system which will perform big data analytics and can identify inconsistent and harmful events within real-time most favorable response for better AI computing. Bonomi et al. [24] examined those disruptions and proposed hierarchical differentiate architecture which extends reliability and security towards the edge of the core network of fog computing. It also elaborated, deeply, a STL system and Wind Farm frameworks in the scenario of fog computing. Osanaiye et al. [27, 92], respectively, referred to fog computing as a substitute of cloud that can deliver location and requirement based services at mobile devices or end users (e.g., sample display application in shopping centers, shared parking, smart vehicular, and bus application services at fog networks) and also extended the reliability of migration with an abstract smart predefined VM live migration system. Osanaiye et al. [27] also mathematically approved that its VM live migration system with a linear regression approach from cloud to near fog structure also has been discussed in previous section (see Figure 5). Lee et al. [93] established the functionality of fog towards hybrid surroundings that can provide elastic and reliable fog computing services in the near targeted 5G network. The authors give fog computing development with intelligent vehicle smart grid to autonomous connected vehicles and vehicular fog network. The concept will help evolution to the Internet of Vehicles or the vehicular fog which will be the equivalent of existing Internet cloud for vehicles.

Separately smart transport system (STS) is used by Osanaiye et al. [27, 85] to diverse and circulate designed systems. Those extended systems frequently can monitor whole traffic and transmit all monitored data in real-time environment through smart devices and sensors, which preempt the whole traffic and provide safety to commuters. Some application works for fog structure, in the domain of healthcare high latency intolerant and augmented reality, have been proposed [87, 94, 95] that can be utilized for improvement of websites response through preprocessing and caching functionality. Saharan and Kumar [96] explained their work in four divergent domains of fog computing applications (e.g., STL system, wireless and actuator networks, smart grid, and connected vehicles). In the end, Table 4 is summarizing the comparison of some benefits and complications belonging to the deployment of fog computing structure.

5. Real-Time Use Cases

5.1. Video Streaming. Data transmission of applications related to video services seems to be well-organized at fog platform, caused by the capacity and elasticity of fog networking for provisioning low latency, mobility, and location identifications, with real-time data analysis. Related work [81] proved a video observation application with the requirement of three-level hierarchal structure to act upon motion detection through smart cameras and then did face recognition process with fog computation followed by identity data collection process at cloud instances. Magurawalage et al. [97] expressed aqua computing, which is the result of inspiration by water cycle that can execute in both fog and cloud processing environment. This proposed application design consists of clones at edge that will be served in video streaming circumstances, at the end user's device, as acted as a buffer. In support of on-demand video streaming [98] fog services are used, like enhanced communication at a virtual desktop structured system for provision real-time video data analytics for surveillance cameras. Other prospective benefits from systemized fog computing deployment are to improve video streaming, likely in artificial intelligent cache streaming. There is a similar research [99] that identified key demands for deployment of fog computing arrangement as a hierarchical structure of CC to support an intellectual network edge node.

5.2. Healthcare Monitoring Systems. Influential health observations or monitoring is the key target of application in biomedical data analytics for correctly predictable smart healthcare decisions in future. This achievement can be possible with the deployment of ubiquitous framework and systems and transmitting data in real-time response environment through fog nodes rather than transmitting at the cloud. Cao et al. [87] have proposed a real-time monitoring application, U-Fall, which can be separated into three major sections, front-end, back-end, and communication module, where both front-end and back-end make independently detections result. Therefore, a collaborative detection increases the accuracy and reduces the wrong alarms rate. Another important experiment is signifying the feature of U-Fall application that involuntarily detects the extensive fall throughout existing moderate strokes, during health monitoring. In the end obtained results suggested that

Table 4: Pros and cons of fog computing.

Pros	Cons
Improves the system response time	Trust and authentication anxiety
Conserves network bandwidth	WAN security concern
Systematic mobility support	Man in the middle and IP address bluffing security issues
Minimizes the latency at core network	Availability or cost for fog equipment/hardware
Superfluous data restriction support to send at cloud	Physical location might be takes away from any data benefit of the cloud anytime and anywhere
Improvement of security by keeping data close to the edge	Privacy issue by the distributed processing

a high sensitivity has been achieved. Utilizing fog as a smart gateway to give services for computing in some complicated methods like embedded data mining and prioritized storage has also been proved [95].

A case study about features of electrocardiogram mining plays a very important role in the analysis of cardiac diseases. In some tentative results of health care diagnosis and analysis monitoring, where fog computing attains more than 90% bandwidth usage efficiency with low latency in real-time response [87], FAST [100] has been proposed as an application method that is a distributed data analytics system in health monitoring with fog based sensors for monitoring and diminishing of strokes. Another application framework has proposed its services for monitoring special diseases with mature body sensor network at the nearest node at the edge [101]. However, fog deployed gateways are used mostly to improve the whole health care systems by improved QoS, like ECG feature extraction. Aazam and Huh [102] proposed a smart phone application work, named Emergency Help Alert Mobile Cloud, which operates in fog environment for preprocessing and offloading for real-time response reporting to an appropriate emergency department (e.g., ambulance) with the help of already stored contact details. These services can make a better help system to identify the incident locations as well as make possibilities to tracing the patients. Dubey et al. [103] have experimented the use of preanalyzed data for next data mining and analytics at unprocessed data collected by distributed sensors in smart health care monitoring. Ahmad et al. [104] presented fog structure as the central layer between centralized cloud and the end users (IoT) through a systemized framework. Therefore, a security solution, named Cloud Access Security Broker, has been introduced as a vital part of health care monitoring at fog nodes.

5.3. Gaming. The initiation of cloud computing has offered computer gaming without existing of more than one players and essential gaming hardware structure; therefore, online gaming facilitators and users are rapidly growing. Wang and Dey [105] described mobile gaming application framework based on the cloud servers, where centralized server works on all possessing load of the game. In this scenario the cloud server transmits only the gaming commands to the players, but in this system disruption is faced due to the high latency and limitation of the cloud structure. There must be a targeted QoS in fog computing to achieve and ensure high gaming QoE [106], but low latency and real-time requirements of latest gaming applications are inherently at risk. Lee et al. [107] have introduced an experiment that how to provide a quick response time with low latency in gaming with cloud processing that can improve the user experience. Consequently, a related model has been developed for better prediction or preassessments about real-time requirements for game inputs. Another structure with cloud services extension [108] has been proposed that can deploy more different geographically distinct devices equipped with intelligent physical resources. Those distributed devices seem to be fog structural nodes for better QoE cloud and also at edge cloud (fog node) gaming environment within the high recognition of massively multiplayer online gaming. A lightweight method [109] described the deployed super nodes for extension of video games processing near proximity of players or end users with low latency and high bandwidth QoS. In this experiment, a collected data encoding rate increases the playback consistency and end time driven by buffer forecast approach.

5.4. Smart Traffic Light (STL) System. Smart traffic light (STL) system interacts with several sensors of fog for monitoring instances for accidents deterrence, preservation of a steady traffic flow, collections of related data for analysis, and improvement of the STL system [2]. That analyzed information can be used in the prevention of possible accidents occurrence through preempt warning signals with connected vehicles. Stojmenovic and Wen [56] are evaluating a system, for sensing and monitoring with video surveillance cameras, the existence of an ambulance through its alert alarm, and emergency light to systematically change traffic lights to permit pass through the traffic for emergency vehicles. Bonomi et al. [24] have clearly identified three major goals of STL system: (i) steady traffic flow management; (ii) prevention from accidents; (iii) analysis at received data to analyze and improve the STL system. Accidental preemptions require a real-time response, while data retrieval and traffic flow control can be referred to lesser priority for real-time set of process. In this scenario, WAP and smart traffic light units can be deployed alongside the road for provision real-time transmitting data services, such as vehicle-to-vehicle, vehicle to fog nodes, and fog nodes to fog nodes.

6. Occasions for Big Data Analytics

Future of developed IoT revolution and technologies will be a key role in IT and OT over next two to four years. So here most related solutions take place that fog computing and big data are both of the most prominent technologies in the nearest future, since providing the groundwork for convergence with the help of mobility, consumerization, predictive analytics, and in-memory databases. Therefore, we will discuss IoT big data analytics challenges and opportunities of process plan through fog structure, rather than processing and storage at centralized cloud framework [110].

6.1. Big Data Opportunities. Few decades ago data was generated and accumulated by the worker or employees of a company. With the high growth of Internet, users are generating their own data on Facebook and Twitter, and these data become larger than older one. Now there has become a third level of this progression because now machines are accumulating data in the building of all over the cities with fully implanted sensors that are monitoring humidity, temperature, and electricity usage with smart meters. On the other hand the satellite around the earth is monitoring the earth 24 hours a day with taking pictures and accumulating data. However, these orders of magnitude are higher than user accumulated data. So it is a progression from employee generated data to the user's generated data then machine's generated data. Primarily intentions are very important for data analysis to achieve efficient operations and fault free and efficient cost running processes in industry [7].

6.2. Big Data Revolution. The big data revolution is gradually rising up and there is no choice to accumulate and arrange for the intellectual and intelligent results for better decision systematically (e.g., health care monitoring applications and industrial controlled IoTs). Smart devices like wearable exercise bands, driverless vehicles, and environmental monitoring sensors, in our lives, are gradually going to digitally connect. These connected devices are producing explosive data packets. According, the information reported by IDC and EMC, the whole digital universe is replication in every two years and will generate 40,000 exabytes data by the year 2020 [4]. There are two key challenges which might be faced, caused by that growth of data. The first is that businesses are ongoing to digital transformation for their services in order to expand a competitive benefit, while the continued explosion of IoT devices is identified as the second [7].

Figure 5 is providing an example of daily based generated data assessments by CISCO in 2017 from large scaled implanted IoT and IIoT with graphical representation. IoT devices and applications are organized at an overwhelming rate from a countless of global endpoints. According to the research studies [51], the number of IoT devices is predicted to get bigger from 1.2 billion presented in 2015 to 5.4 billion connected devices globally by 2020. This proliferated forecast means that explosively huge amount data will be generated and it should be analyzed for accumulation of the benefits and QoS user's applications. The term "big data," for this large volume of data, is used. According to big data that can be divided into high volume, variety, and velocity informational sources [4, 51], cost effective and innovative forms of information processing are required. So, what is the appropriate effective way to manage and process large and complex big data that enables better decision making, enhanced insight, and process automation? For rising of all these queries and challenges, fog computing becomes visible as the best solution.

6.3. IoT Big Data Analytics and Industrial Growth. Industry and companies rest over many difficult operating problems, such as from complex supply chain to motion applications, in equipment constraints to labor limitations, over every single day. For manufactures solving problems is nothing new. However with IoT big data analytics, comparative organizations can find new cost saving and revenue opportunities, instead of their flood of data becoming in overwhelming challenges. Analytics helps in operations to sense what important, understand what values can be generated, and act immediately to capture those values. Specifically manufacturers are using data analytics to predict depending equipment failures and improve quality and market responsiveness by coordinating extended and complex supply chains. Data analytics can better engage with their customers to develop new revenue streams and enhance product features and new product development. We can say that streaming of IoT big data presents one of the biggest opportunities [7]. Figure 6 is characterizing the benefits of big data analytics at fog computing and showing some predictive progress as a five-dimension emerging targets in near smart industrial development through user's smart devices growth.

6.4. The Future of Handling Big Data. What does a successful deployment of fog computing framework look like? Recently an accomplished computing and network framework to manage tremendously large amounts of big IoT data generated by end users or devices is described [111]. Fog computing is a framework which is an appreciative extension of the prenetworking system and cloud services. Fog and cloud computing operate together to generate IT solutions (e.g., increase network connectivity, processor capacity, security management, and analytics platforms). Previously it has been presented [86] that some experimental results claimed a sample of IoT framework with smart devices like mobile phones, home gateways, and so on. These devices can be operated for parallel data analytic but this procedure might be harmful in future application use. As a huge scale of data required to manage big data analytics, some organizations having the idea to the deployment of fog computing. Simultaneously, the security of cluster technology becomes an open issue caused by the dynamic features of SDNs. To overcome the requirements of large scale network setups, the control plane is typically implemented as a distributed controller. To meet the above challenges, a big data analysis based secure cluster management framework for optimal control plane has been proposed recently [28]. Respectively, cluster management technology operates all types of proceedings and must continue a steady global network status, which significantly leads to big data in SDNs. Concurrently, the

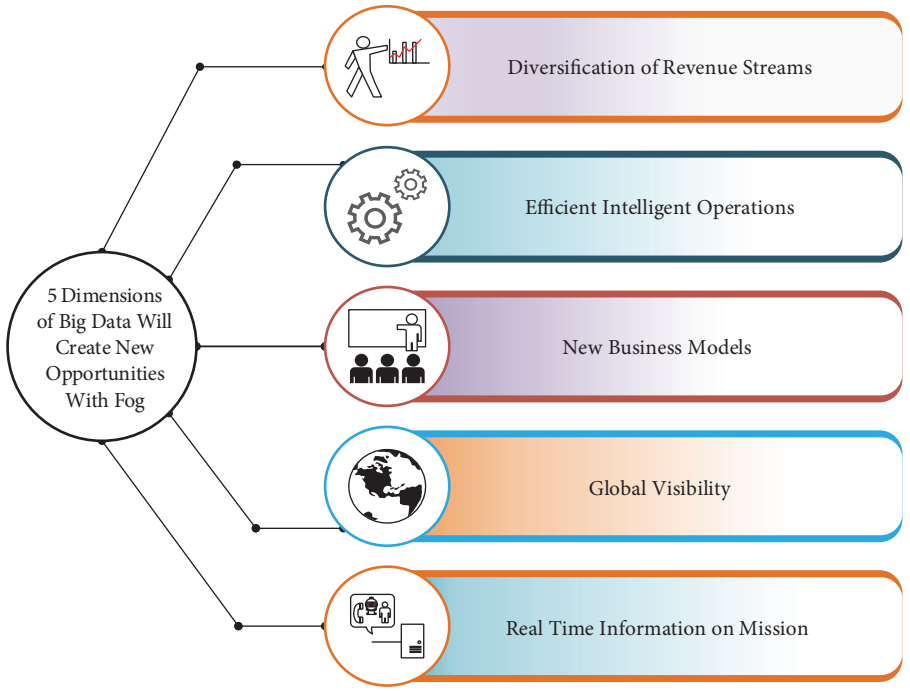


Figure 6: Industrial development by big data analytics expansion at fog computing structure.

security of cluster technology becomes an open issue caused by the dynamic features of SDNs. Jayaraman et al. [112] have proposed an application model which is an extensible, flexible, generic, and component based platform for real-time analytics. This application named CARDAP can be installed in complex dispersed mobile data analysis applications (e.g., sensing and monitoring environment in smart cities). CARDAP combines a number of resourceful data transmission strategies by real-time data stream mining in the context of data reducing and transmission.

From Figure 7 we see that the main feature of the CARDAP application is enabling the development of generated data analytics with connecting the featured components by using XML configuration files on a local device. The CARDAP has five key featured components. (1) Data capture component used the plugin and virtual sensor approach as an interface to handle the data stream. Collection of assorted data sources is enabled by the plugin and virtual sensors together. (2) Data analytics manager enables the access of users and developers for deploying specific data analysis algorithms. (3) Open mobile miner component applies a light weight clustering at analytic data to match high-speed data streaming. This technique also acquires best accuracy of data analytics based on existing resources. (4) Data Sink component allows applications to push processed data on any linked fog node or cloud. (5) The query and storage manager achieve the functionality of storing analytic data locally that could be used in critical real-time operation.

We believe that better deployment of fog framework will take place through a third-party service provider (e.g., ad hoc cellular mobile networks) which can deploy and manage fog nodes for enterprising work. Some of the ordinary enterprising difficulties to deploying a fog structure are similarly faced by organizations for private data infrastructure. That is why we have described a proposal which presents a framework for building mesh hybrid IT environments [51] in which data is produced by people, sensors, and devices at the edge that can be interconnected over low latency and high-speed connections securely. This blueprint is called Interconnection Oriented Architecture [1]. It presents all the benefits of fog framework as well as applicable for all types of non-IoT functionality to bring big data analytics to the edge of the core network. Businesses of all sizes will be utilizing some form of big data analytics to improve their industry and will be connected with digital services. Therefore, enterprises need to make sure that their IT infrastructure is operational to handle the growth of data and all operations for big IoT data analytics.

7. An Emerging Era of Fog

The discussion about fog computing features introduces and separates the fog computing framework from CC, for the new technology requirement of IoT. The corresponding system with centralized cloud, fog, is being prominent alongside the following two magnitudes:

(1) Accumulation of significant information and data storage at close proximity to end user, in spite of storage system that will be running only at cloud data center.

(2) Accumulation of significant amount of data for required processing at closer location of the end user or IoT devices, to reduce processing functionality at cloud framework. Such processing and organized functions can improve the following [113]:

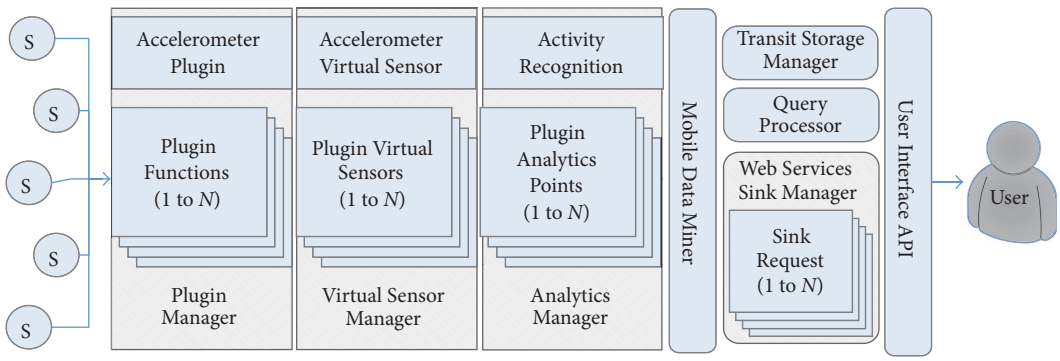


Figure 7: CARDAP system architecture.

(i) Better functionality for big IoT data analytics generated by end users or IoT devices.

(ii) Improved functionality for significant intelligent operational processes for proposed use cases in near future (e.g., smart cities, vehicles, and smart grids).

(iii) Enhanced management with elasticity for IoT as well as IIoT in the context of 4th industrial revaluation state towards the next industrial state and superior use of cloud computation with its complemented fog networking for big IoT data analytics [7].

7.1. Future Work Use Cases Study. Due to the previous work discussion, there are some questions raised such as how to put the modules back together, who does what, and at what timescale? Significant structural design of fog computing supports a diversity of applications belongs to the IoT and IIoT as well as with those things that are frequently discussed in 5G technology requirements and big data analysis [43]. Particularly, fog computing architecture is featured with data plane and control plane both; each of them is rapidly rising the number of examples in protocol layers to the application layer from the physical layer.

(i) *Fog Data Plane*. Fog data plane works at (i) bandwidth control and content caching at the edge, (ii) pooling of user's idle computing/bandwidth/resources/storage and local contents, (iii) micro data centers and cloudlets at the edge, and (iv) device-to-device mutual data transmission.

(ii) *Fog Control Plane*. Fog control plane is working through (i) sensing inference of network states, (ii) client level controlled cloud storage, (iii) content management over the top, (vi) fog computation driven RAN, (v) management of sessions and load of signals at the edge, (iv) IoT big data analytics and real-time data streaming, and (vii) client based HetNets control.

Fog data plane has been more broadly studied [10]. Here, we emphasize a few meticulous cases that demonstrate the challenges and potential of fog control plane. We will also see that fog computing operates across the hierarchical and distributed devices, accessing at an edge, and observes the associated work between fog and cloud, which are defined in following cases [6]:

(i) Load balancing LTE states (in commercial deployment) with some complement of passive extents (e.g., RSRQ), throughput, dynamic analytics, and data mining might infer the number of resource blocks used state [114].

(ii) Smart data pricing and deployment of OTT network (commercially) as fog computing for enhancement directly to the IoT structure and removing the reliance on the whole network [115].

(iii) Client side HetNets control (in 3GPP Standards) with existence of diverse network components. There is a key attribute which exists in cellular networks [116], and in this case, fog to cloud structure allows real-time response relationship through its interface, while time taking parameters, like having RAT stability characteristic data, can pass from the cloud.

(iv) Hierarchical user control cloud storage (in beta trial), by the use of cloud based cheap storage through control of client for privacy at fog nodes, we can achieve the best of both frameworks (cloud and fog) for described challenges based on new IoT structure [6]. For example, dispersal of bytes of a file across multiple public clouds by file portioning on the fog state can be secured and provide privacy for significant information, even given encryption key would be leaked [117].

(v) Real-time processing in AI (in beta trial) is a consideration of virtual reality in which most of the information recovery and processing targets may be forwarded to the related devices, such as some on associated phones, some on smart home storage, and many others and then the rest to the cloud. Fog architecture may process for those devices in parallel, with intelligent location hierarchy [118].

(vi) The future of big data analytics (in disruptive business models) is the best performance for a successful venture. IT strategy should be customized to the needs of the specific organization based on best data analysis. By using appropriate data analysis and

prioritization techniques, critical data (which needs real-time treatment) can be recognized and processed at the fog node for provision QoS to the end users [111].

Many more cases can be discussed in context of emerging era of fog, for example, borrowing bandwidth from closest other idle devices or end users when multiple devices processing to the same node [119], bandwidth management at home gateway [120, 121], distributed beamforming, where fog existence has some physical layer, such as with utilization of multiuser MIMO which is described during laboratory demonstration for improving throughput and reliability [122]. These all can be done entirely at the client side through fog computing and helps meet the challenges of big IoT data analytics.

7.2. Nearly Targeted Use Cases

7.2.1. Smart Grids. Currently proposed work in the scenario of smart grids can be associated caused by the energy demands. Nowadays we have a rate of advancement that the energy power is produced and controlled by habitual method technology. There are also some requirements to impose reducing gas emission controlled curtail and climate change [123]. Cyber Physical Energy System [124] is proposed for improving performance and reliability, for efficient production consumption of power from grids through the better smart management of demand and supply, throughout fog intelligence framework. A model application of this system has been implemented with fog platform to support monitoring, automotive response, and scalability. Another paper work [125] described an important innovative energy supervision implemented with fog structure. This can monitor the energy management in tow domains of prototypes HEM and HVAC. Both prototypes will provide services in-house energy management and micro-grid-level energy management, respectively. Abdelwahab et al. [126] have explained cloud processing based remote sensing advancement to determine and aggregate smart grid operational data for QoS at smart power grid through fog computing. Cloud computing characteristics that are providing services to user and suppliers directly already have been proved for inefficient bandwidth approach.

7.2.2. Smart Cities. A smart city is another one of the key goals for IoT applications with fog based structure that has the range from automated traffic control to smart energy management buildings and homes. The objective is to overcome challenges associated with rapid urban growth. Yi et al. [127] recognized fog computing nodes at closed edge location which is a unique solution for near smart city projects as well as having extensions of all functionality of cloud structure in an integrated structure to enable and connect explosively increased smart applications with flexible resources. Kitchin [128] is describing a smart city model that has full control of ubiquitous processing for an economy that is driven by modernism and creative policies. IoT structure and applications frameworks for making a city to smart city will require high computational and storage capacities, and will face interoperability challenges. A smart city is described [68] as community system at the edge that has optimized energy utilization and can provide a better quality of life through the smart life of citizens. Byers and Wetterwald [129] identified complication that is associated with centralized cloud relating management of road traffic control, parking system, and environmental monitoring as a smart city use case. A comprehensive research which describes a strong relation between fog and big IoT data analytics [77] is a hierarchical fog networking structure to support a huge number of automotive components and services for smart cities in future work.

7.2.3. Smart Connected Vehicles. With the cumulative number of embedded sensor vehicles and the acceptance of various interfaces for communication, vehicles have become a sensing environment platform for many applications, for instance, road safety, traffic and weather monitoring, driver behavior, and urban surveillance analysis. Additionally, smart connecting vehicles could be synchronized to sense large scale data efficiently. Such direction requires massive computing, power, and communication resources. Wang et al. [130] recently have proposed a virtual vehicle coordination system based on the concept of group consent for sensing the environment information. Specifically, they propose a discovery algorithm to find the optimum virtual vehicle groups. Next, in the scenario of multigroup consent theory, they formulate a coordination algorithm to control virtual vehicle groups to attain multigroup harmonization by regulating the communication connectivity among virtual vehicles. Finally, the exact environment information is attained through the harmonization of multiple virtual vehicle groups.

Connected vehicles are a subsection of the smart city where vehicles will work as fog nodes at the edge to sense situation in driving, urban obstruction, and innovation of footage. In this case an application work [131] assists the safety and above-mentioned requirements. Vehicular cloud attributes are significantly compared with the traditional cloud structure. A vehicular fog computing has been proposed [132] where a connected vehicle can act as a fog node, and this functionality can engage the association of many end user devices. A vehicular fog as a fog computing structure in vehicles and the core network surroundings for improvement in automotive driving has been extended [93]. Lee et al. describe VANET as MANET and will use the vehicles as an end node for smart phones. An architecture is planned by merging VANET with in SDN [73] to provide services for future VANET anxiety. During crowded vehicles in parking, some solutions [133] are given for deficient management of shared parking. Simulations consequences have been indicating effectiveness and consistency in monitoring empty parking slot.

8. Conclusion and Future Work

We have to close up fog computing overview, as distributed and hierarchical framework having scheduled storage, virtual machine images (embedded virtual cache and transient

storage), and network resources deployment with the complement of CC functionality at the edge of IoT. This work represented some key characteristics with enduring general ideas about possible challenges as like how fog computing can expand CC services at the edge. Hence, it is clarifying some use cases that provoked the necessity of fog computing especially about the real-time data analysis importance for IIoT, typically in health care, STLS, and smart grid. Our work emphasizes fog's consequences and disruption in main three aspects, IoT, big data analytics, and storage. With concentration at the future potential of the emerging era of fog, this survey provided some nearly imminent use cases which recount some near time research directions and distributed application work for flexibility in QoS of fog computing framework. Fog computing having some dependency of existing key technology can be categorized, namely, computing (e.g., computation offloading and latency management), communication (e.g., SDN, NFV, and 5G), and storage (precache system and storage expansion). Fog networking could be more intelligent with the deployment of existing key technologies. SDN and NFV together can increase the network scalability as well as cost reduction and live VM migration (computation offloading) can meet the resource constraint challenges on edge devices.

In spite of many opportunities of fog computing for business continuity, proficiency, cost reductions, and so on, there are also many challenges faced at industrial data management phase. A lot of research works have been considering IIoT applications and most of them have been connected with fog computing, surely in primarily developing markets such as manufacturing or oil and gas. Yet, most of research works still in waiting or uncertain towards the progress and development of smart cities might be with combinations of smart building, STLS and sensor monitoring networks that are already organized. Cloud provides services for wide area connectivity, global coordination, heavy duty computation, and massive storage capacity, while fog will facilitate user-centric service, edge resource pooling, rapid innovation, and real-time processing [6]. This transformative epoch is an interesting era to start thoroughly discovering what fog may look like and which differences will be passed to the world of virtual computing in the next 10 years.

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 Natural Science Foundation of China (no. 61472047).

References

[1] I-SCOOP, "The growing role of fog and edge computing in IoT," https://www.i-scoop.eu/internet-of-things-guide/#The_growing_role_of_fog_and_edge_computing_in_IoT.

[2] 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. 13–15, Finland, August 2012.

[3] I-SCOOP, "Internet of Things Trends for 2017 and Beyond," https://www.i-scoop.eu/internet-of-things-guide/#Internet_of_Things_trends_for_2017_and_beyond.

[4] J. Gantz and D. Reinsel, "The digital universe in 2020: Big data, bigger digital shadows, and biggest growth in the far east," *IDC iView: IDC Analyze the future*, vol. 2007, pp. 1–16, 2012.

[5] S. Hoque, M. S. D. Brito, A. Willner, O. Keil, and T. Magedanz, "Towards Container Orchestration in Fog Computing Infrastructures," in *Proceedings of the 41st IEEE Annual Computer Software and Applications Conference Workshops, COMPSAC 2017*, pp. 294–299, Italy, July 2017.

[6] M. Chiang and T. Zhang, "Fog and IoT: an overview of research opportunities," *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 854–864, 2016.

[7] S. Yin and O. Kaynak, "Big data for modern industry: challenges and trends," *Proceedings of the IEEE*, vol. 103, no. 2, pp. 143–146, 2015.

[8] E. Cuervoy, A. Balasubramanian, D.-K. Cho et al., "MAUI: making smartphones last longer with code offload," in *Proceedings of the 8th Annual International Conference on Mobile Systems, Applications and Services (MobiSys '10)*, pp. 49–62, New York, NY, USA, June 2010.

[9] A. Greenberg, J. Hamilton, D. A. Maltz, and P. Patel, "The cost of a cloud: research problems in data center networks," *ACM SIGCOMM Computer Communication Review*, vol. 39, no. 1, pp. 68–73, 2008.

[10] 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.

[11] K. Lee, D. Kim, D. Ha, U. Rajput, and H. Oh, "On security and privacy issues of fog computing supported Internet of Things environment," in *Proceedings of the International Conference on the Network of the Future, NOF 2015*, Canada, October 2015.

[12] J. Su, F. Lin, X. Zhou, and X. Lu, "Steiner tree based optimal resource caching scheme in fog computing," *China Communications*, vol. 12, no. 8, Article ID 7224698, pp. 161–168, 2015.

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

[14] 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.

[15] D. C. Verma and P. Verma, *Techniques for Surviving Mobile Data Explosion*, John Wiley & Sons, 2014.

[16] A. Botta, W. de Donato, V. Persico, and A. Pescapé, "Integration of cloud computing and internet of things: a survey," *Future Generation Computer Systems*, vol. 56, pp. 684–700, 2016.

[17] B. Rochwerger, D. Breitgand, E. Levy et al., "The Reservoir model and architecture for open federated cloud computing," *IBM Journal of Research and Development*, vol. 53, no. 4, pp. 4:1–4:11, 2009.

[18] M. Armbrust, A. Fox, R. Griffith et al., "A view of cloud computing," *Communications of the ACM*, vol. 53, no. 4, pp. 50–58, 2010.

[19] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of things: a survey on enabling technologies, protocols, and applications," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015.

[20] 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.

[21] 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.

[22] 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*, 2016.

[23] S. K. Datta, C. Bonnet, and J. Haerri, "Fog Computing architecture to enable consumer centric Internet of Things services," in *Proceedings of the IEEE International Symposium on Consumer Electronics, ISCE 2015*, Spain, June 2015.

[24] F. Bonomi, R. Milito, P. Natarajan, and J. Zhu, "Fog computing: A platform for internet of things and analytics," *Studies in Computational Intelligence*, vol. 546, pp. 169–186, 2014.

[25] P. More, "Review of implementing fog computing," *International Journal of Research in Engineering and Technology*, vol. 04, no. 06, pp. 335–338, 2015.

[26] J. Zhu, D. S. Chan, M. S. Prabhu, P. Natarajan, H. Hu, and F. Bonomi, "Improving web sites performance using edge servers in fog computing architecture," in *Proceedings of the IEEE 7th International Symposium on Service-Oriented System Engineering (SOSE '13)*, pp. 320–323, IEEE, San Francisco, Calif, USA, March 2013.

[27] O. Osanaiye, S. Chen, Z. Yan, R. Lu, K.-K. R. Choo, and M. Dlodlo, "From Cloud to Fog Computing: A Review and a Conceptual Live VM Migration Framework," *IEEE Access*, vol. 5, pp. 8284–8300, 2017.

[28] J. Wu, M. Dong, K. Ota, J. Li, and Z. Guan, "Big Data Analysis-Based Secure Cluster Management for Optimized Control Plane in Software-Defined Networks," *IEEE Transactions on Network and Service Management*, vol. 15, no. 1, pp. 27–38, 2018.

[29] I. Ku, Y. Lu, and M. Gerla, "Software-defined mobile cloud: architecture, services and use cases," in *Proceedings of the 10th International Wireless Communications and Mobile Computing Conference (IWCMC '14)*, Nicosia, Cyprus, August 2014.

[30] B. Heller, R. Sherwood, and N. McKeown, "The controller placement problem," in *Proceedings of the 1st ACM International Workshop on Hot Topics in Software Defined Networks, HotSDN 2012*, pp. 7–12, Finland, August 2012.

[31] A. Basta, W. Kellerer, M. Hoffmann, H. J. Morper, and K. Hoffmann, "Applying NFV and SDN to LTE mobile core gateways; the functions placement problem," in *Proceedings of the 4th ACM Workshop on All Things Cellular: Operations, Applications, and Challenges, AllThingsCellular 2014*, pp. 33–38, usa, August 2014.

[32] J. Hwang, K. K. Ramakrishnan, and T. Wood, "NetVM: high performance and flexible networking using virtualization on commodity platforms," *IEEE Transactions on Network and Service Management*, vol. 12, no. 1, pp. 34–47, 2015.

[33] J. Martins, M. Ahmed, C. Raiciu, V. Olteanu, M. Honda, R. Bifulco et al., "ClickOS and the art of network function virtualization," in *Proceedings of the USENIX Conference on Networked Systems Design and Implementation*, pp. 459–473, 2014.

[34] B. Yang, W. K. Chai, Z. Xu, K. V. Katsaros, and G. Pavlou, "Cost-Efficient NFV-Enabled Mobile Edge-Cloud for Low Latency Mobile Applications," *IEEE Transactions on Network and Service Management*, vol. 15, no. 1, pp. 475–488, 2018.

[35] H. Madsen, G. Albeanu, B. Burtschy, and F. Popentiu-Vladicescu, "Reliability in the utility computing era: Towards reliable fog computing," in *Proceedings of the 2013 20th International Conference on Systems, Signals and Image Processing, IWSSIP 2013*, pp. 43–46, Romania, July 2013.

[36] K. Hong, D. Lillethun, U. Ramachandran, B. Ottenwälder, and B. Koldehofe, "Opportunistic spatio-temporal event processing for mobile situation awareness," in *Proceedings of the 7th ACM International Conference on Distributed Event-Based Systems, DEBS 2013*, pp. 195–206, USA, July 2013.

[37] B. Ottenwälder, B. Koldehofe, K. Rothermel, K. Hong, and U. Ramachandran, "RECEP: Selection-based reuse for distributed complex event processing," in *Proceedings of the 8th ACM International Conference on Distributed Event-Based Systems, DEBS 2014*, pp. 59–70, India, May 2014.

[38] 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*, 2017.

[39] B. Sheng, Q. Li, and W. Mao, "Data storage placement in sensor networks," in *Proceedings of the 7th ACM International Symposium on Mobile Ad Hoc Networking and Computing, MOBIHOC 2006*, pp. 344–355, ita, May 2006.

[40] H. Wang, C. C. Tan, and Q. Li, "Snoogle: A Search Engine for the Physical World," in *Proceedings of the IEEE INFOCOM 2008 - IEEE Conference on Computer Communications*, pp. 1382–1390, Phoenix, AZ, USA, April 2008.

[41] H. D. Wang, C. C. Tan, and Q. Li, "Snoogle: a search engine for pervasive environments," *IEEE Transactions on Parallel and Distributed Systems*, vol. 21, no. 8, pp. 1188–1202, 2010.

[42] Y. Zhang, C. Tan, and Q. Li, "CacheKeeper: A system-wide web caching service for smartphones," in *Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing, UbiComp 2013*, pp. 265–274, Switzerland, September 2013.

[43] X. Wang, M. Chen, T. Taleb, A. Ksentini, and V. C. M. Leung, "Cache in the air: exploiting content caching and delivery techniques for 5G systems," *IEEE Communications Magazine*, vol. 52, no. 2, pp. 131–139, 2014.

[44] N. I. Md Enzai and M. Tang, "A taxonomy of computation offloading in mobile cloud computing," in *Proceedings of the 2nd IEEE International Conference on Mobile Cloud Computing, Services, and Engineering*, pp. 19–28, Oxford, UK, April 2014.

[45] 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.

[46] 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.

[47] M. S. Gordon, D. A. Jamshidi, S. A. Mahlke, Z. M. Mao, and X. Chen, "COMET: Code Offload by Migrating Execution Transparently," in *OSDI*, pp. 93–106, Code Offload by Migrating Execution Transparently, COMET, 2012.

[48] B. Ottenwälder, B. Koldehofe, K. Rothermel, and U. Ramachandran, "MigCEP: Operator migration for mobility driven distributed complex event processing," in *Proceedings of the 7th*

ACM International Conference on Distributed Event-Based Systems, DEBS 2013, pp. 183–194, USA, July 2013.

[49] W. Liu, T. Nishio, R. Shinkuma, and T. Takahashi, "Adaptive resource discovery in mobile cloud computing," *Computer Communications*, vol. 50, pp. 119–129, 2014.

[50] T. Nishio, R. Shinkuma, T. Takahashi, and N. B. Mandayam, "Service-oriented heterogeneous resource sharing for optimizing service latency in mobile cloud," in *Proceedings of the 1st International Workshop on Mobile Cloud Computing and Networking, MobileCloud 2013*, pp. 19–26, ind, July 2013.

[51] P. Fan, "Coping with the big data: Convergence of communications, computing and storage," *China Communications*, vol. 13, no. 9, Article ID 7582312, pp. 203–207, 2016.

[52] O. Osanaiye, K.-K. R. Choo, and M. Dlodlo, "Distributed denial of service (DDoS) resilience in cloud: review and conceptual cloud DDoS mitigation framework," *Journal of Network and Computer Applications*, vol. 67, pp. 147–165, 2016.

[53] Y. Shi, S. Abhilash, and K. Hwang, "Cloudlet mesh for securing mobile clouds from intrusions and network attacks," in *Proceedings of the 3rd IEEE International Conference on Mobile Cloud Computing, Services, and Engineering, MobileCloud 2015*, pp. 109–118, San Francisco, Calif, USA, April 2015.

[54] M. Zhanikeev, "A cloud visitation platform to facilitate cloud federation and fog computing," *The Computer Journal*, vol. 48, no. 5, Article ID 7111861, pp. 80–83, 2015.

[55] Y. Wang, T. Uehara, and R. Sasaki, "Fog computing: Issues and challenges in security and forensics," in *Proceedings of the 39th IEEE Annual Computer Software and Applications Conference Workshops, COMPSACW 2015*, pp. 53–59, Taichung, Taiwan, July 2015.

[56] I. Stojmenovic and S. Wen, "The fog computing paradigm: scenarios and security issues," in *Proceedings of the Federated Conference on Computer Science and Information Systems (FedCSIS '14)*, pp. 1–8, IEEE, Warsaw, Poland, September 2014.

[57] I. Stojmenovic, S. Wen, X. Huang, and H. Luan, "An overview of Fog computing and its security issues," *Concurrency Computation*, vol. 28, no. 10, pp. 2991–3005, 2016.

[58] Y. W. Law, M. Palaniswami, G. Kounga, and A. Lo, "WAKE: Key management scheme for wide-area measurement systems in smart grid," *IEEE Communications Magazine*, vol. 51, no. 1, pp. 34–41, 2013.

[59] S. Yi, Z. Qin, and Q. Li, "Security and privacy issues of fog computing: A survey," in *Proceedings of the Springer Conference on Wireless Algorithms, Systems*, pp. 685–695, 2015.

[60] S. J. Stolfo, M. B. Salem, and A. D. Keromytis, "Fog computing: Mitigating insider data theft attacks in the cloud," in *Proceedings of the 1st IEEE Security and Privacy Workshops, SPW 2012*, pp. 125–128, San Francisco, Calif, USA, May 2012.

[61] C. Zuo, J. Shao, G. Wei, M. Xie, and M. Ji, "CCA-secure ABE with outsourced decryption for fog computing," *Future Generation Computer Systems*, vol. 78, pp. 730–738, 2018.

[62] K.-K. R. Choo and R. Sarre, "Balancing privacy with legitimate surveillance and lawful data access," *IEEE Cloud Computing*, vol. 2, no. 4, pp. 8–13, 2015.

[63] K.-K. R. Choo, "Legal Issues in the Cloud," *IEEE Cloud Computing*, vol. 1, no. 1, pp. 94–96, 2014.

[64] K.-K. R. Choo, "Challenges in dealing with politically exposed persons," *Trends and Issues in Crime and Criminal Justice*, p. 1, 2010.

[65] M. Zhou, R. Zhang, W. Xie, W. Qian, and A. Zhou, "Security and privacy in cloud computing: a survey," in *Proceedings of the 6th International Conference on Semantics, Knowledge and Grid (SKG '10)*, pp. 105–112, Beijing, China, November 2010.

[66] M. Dong, K. Ota, and A. Liu, "Preserving source-location privacy through redundant fog loop for wireless sensor networks," in *Proceedings of the 15th IEEE International Conference on Computer and Information Technology, CIT 2015, 14th IEEE International Conference on Ubiquitous Computing and Communications, IUCC 2015, 13th IEEE International Conference on Dependable, Autonomic and Secure Computing, DASC 2015 and 13th IEEE International Conference on Pervasive Intelligence and Computing, PICom 2015*, pp. 1835–1842, UK, October 2015.

[67] S. Kulkarni, S. Saha, and R. Hockenbury, "Preserving privacy in sensor-fog networks," in *Proceedings of the 2014 9th International Conference for Internet Technology and Secured Transactions, ICITST 2014*, pp. 96–99, UK, December 2014.

[68] P. G. Lopez, A. Montresor, D. Epema, A. Datta, T. Higashino, A. Iamnitchi et al., "Edge-centric computing: Vision and challenges," *ACM SIGCOMM Computer Communication Review*, vol. 45, pp. 37–42, 2015.

[69] R. Lu, X. Liang, X. Li, X. Lin, and X. Shen, "EPPA: an efficient and privacy-preserving aggregation scheme for secure smart grid communications," *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 9, pp. 1621–1632, 2012.

[70] K. Ha, Z. Chen, W. Hu, W. Richter, P. Pillai, and M. Satyanarayanan, "Towards wearable cognitive assistance," in *Proceedings of the 12th Annual International Conference on Mobile Systems, Applications, and Services, MobiSys 2014*, pp. 68–81, USA, June 2014.

[71] M. Satyanarayanan, Z. Chen, K. Ha, W. Hu, W. Richter, and P. Pillai, "Cloudlets: At the leading edge of mobile-cloud convergence," in *Proceedings of the 2014 6th International Conference on Mobile Computing, Applications and Services, MobiCASE 2014*, pp. 1–9, USA, November 2014.

[72] M. Swan, "Sensor mania! The internet of things, wearable computing, objective metrics, and the quantified self 2.0," *Journal of Sensor and Actuator Networks*, vol. 1, no. 3, pp. 217–253, 2012.

[73] N. B. Truong, G. M. Lee, and Y. Ghamri-Doudane, "Software defined networking-based vehicular Adhoc Network with Fog Computing," in *Proceedings of the 14th IFIP/IEEE International Symposium on Integrated Network Management (IM '15)*, pp. 1202–1207, Ottawa, Canada, May 2015.

[74] T. Chen, M. Matinmikko, X. Chen, X. Zhou, and P. Ahokangas, "Software defined mobile networks: concept, survey, and research directions," *IEEE Communications Magazine*, vol. 53, no. 11, pp. 126–133, 2015.

[75] D. Kreutz, F. M. V. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: a comprehensive survey," *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14–76, 2015.

[76] C. Douligeris and D. N. Serpanos, *Network Security: Current Status and Future Directions*, John Wiley & Sons, 2007.

[77] B. Tang, Z. Chen, G. Hefferman, T. Wei, H. He, and Q. Yang, "A hierarchical distributed fog computing architecture for big data analysis in smart cities," in *Proceedings of the ASE BigData SocialInformatics*, p. 28, 2015.

[78] Y. Shi, G. Ding, H. Wang, H. Eduardo Roman, and S. Lu, "The fog computing service for healthcare," in *Proceedings of the 2nd International Symposium on Future Information and Communication Technologies for Ubiquitous HealthCare, Ubi-HealthTech 2015*, pp. 70–74, China, May 2015.

[79] D. Willis, A. Dasgupta, and S. Banerjee, "ParaDrop: A multi-tenant platform to dynamically install third party services on wireless gateways," in *Proceedings of the 9th ACM MobiCom Workshop on Mobility in the Evolving Internet Architecture, MobiArch 2014*, pp. 43–48, USA, September 2014.

[80] K. Gai, M. Qiu, H. Zhao, L. Tao, and Z. Zong, "Dynamic energy-aware cloudlet-based mobile cloud computing model for green computing," *Journal of Network and Computer Applications*, vol. 59, pp. 46–54, 2016.

[81] K. Hong, D. Lillethun, U. Ramachandran, B. Ottenwälder, and B. Koldehofe, "Mobile fog: a programming model for large-scale applications on the internet of things," in *Proceedings of the 2nd ACM SIGCOMM Workshop on Mobile Cloud Computing (MCC '13)*, pp. 15–20, Hong Kong, China, August 2013.

[82] S. Sarkar, S. Chatterjee, and S. Misra, "Assessment of the Suitability of Fog Computing in the Context of Internet of Things," *IEEE Transactions on Cloud Computing*, vol. 6, no. 1, pp. 46–59, 2015.

[83] M. Aazam and E.-N. Huh, "Fog computing micro datacenter based dynamic resource estimation and pricing model for IoT," in *Proceedings of the IEEE 29th International Conference on Advanced Information Networking and Applications (AINA '15)*, pp. 687–694, IEEE, Gwangiu, South Korea, March 2015.

[84] C. F. C. Solutions, *Unleash the power of the Internet of Things*, Cisco Systems Inc., 2015.

[85] C. Dsouza, G.-J. Ahn, and M. Taguinod, "Policy-driven security management for fog computing: preliminary framework and a case study," in *Proceedings of the 15th IEEE International Conference on Information Reuse and Integration (IEEE IRI '14)*, pp. 16–23, August 2014.

[86] A. Mukherjee, H. S. Paul, S. Dey, and A. Banerjee, "ANGELS for distributed analytics in IoT," in *Proceedings of the 2014 IEEE World Forum on Internet of Things, WF-IoT 2014*, pp. 565–570, Republic of Korea, March 2014.

[87] Y. Cao, P. Hou, D. Brown, J. Wang, and S. Chen, "Distributed analytics and edge intelligence: Pervasive health monitoring at the era of fog computing," in *Proceedings of the ACM Workshop on Mobile Big Data, Mobidata 2015*, pp. 43–48, China.

[88] T. H. Laine, C. Lee, and H. Suk, "Mobile gateway for ubiquitous health care system using ZigBee and Bluetooth," in *Proceedings of the 8th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing, IMIS 2014*, pp. 139–145, UK, July 2014.

[89] V. Gazis, A. Leonardi, K. Mathioudakis, K. Sasloglou, P. Kikiras, and R. Sudhaakar, "Components of fog computing in an industrial internet of things context," in *Proceedings of the 12th Annual IEEE International Conference on Sensing, Communication, and Networking - Workshops, SECON Workshops 2015*, pp. 37–42, USA, June 2015.

[90] K. Habak, M. Ammar, K. A. Harras, and E. Zegura, "Femto clouds: leveraging mobile devices to provide cloud service at the edge," in *Proceedings of the 8th IEEE International Conference on Cloud Computing (CLOUD '15)*, pp. 9–16, July 2015.

[91] Y. Yan and W. Su, "A fog computing solution for advanced metering infrastructure," in *Proceedings of the 2016 IEEE/PES Transmission and Distribution Conference and Exposition, T and D 2016*, USA, May 2016.

[92] T. H. Luan, L. Gao, Z. Li, Y. Xiang, G. Wei, and L. Sun, "Fog computing: Focusing on mobile users at the edge," https://arxiv.org/abs/1502.01815.

[93] E.-K. Lee, M. Gerla, G. Pau, U. Lee, and J.-H. Lim, "Internet of Vehicles: From intelligent grid to autonomous cars and vehicular fogs," *International Journal of Distributed Sensor Networks*, vol. 12, no. 9, 2016.

[94] A. V. Dastjerdi, H. Gupta, R. N. Calheiros, S. K. Ghosh, and R. Buyya, "Fog computing: Principles, architectures, and applications," https://arxiv.org/abs/1601.02752.

[95] T. N. Gia, M. Jiang, A.-M. Rahmani, T. Westerlund, P. Liljeberg, and H. Tenhunen, "Fog computing in healthcare Internet of Things: A case study on ECG feature extraction," in *Proceedings of the 15th IEEE International Conference on Computer and Information Technology, CIT 2015, 14th IEEE International Conference on Ubiquitous Computing and Communications, IUCC 2015, 13th IEEE International Conference on Dependable, Autonomic and Secure Computing, DASC 2015 and 13th IEEE International Conference on Pervasive Intelligence and Computing, PICom 2015*, pp. 356–363, UK, October 2015.

[96] K. P. Saharan and A. Kumar, "Fog in Comparison to Cloud: A Survey," *International Journal of Computer Applications*, vol. 122, no. 3, pp. 10–12, 2015.

[97] C. S. Magurawalage, K. Yang, and K. Wang, "Aqua computing: Coupling computing and communications," https://arxiv.org/abs/1510.07250.

[98] X. Zhu, D. S. Chan, H. Hu, M. S. Prabhu, E. Ganesan, and F. Bonomi, "Improving video performance with edge servers in the fog computing architecture," *Intel Technology Journal*, vol. 19, 2015.

[99] J. Foerster, D. Ott, O. Oyman, Y. Liao, S. Somayazulu, X. Zhu et al., "Towards realizing video aware wireless networks," *Intel Technology Journal*, vol. 19, 2015.

[100] Y. Cao, S. Chen, P. Hou, and D. Brown, "FAST: A fog computing assisted distributed analytics system to monitor fall for stroke mitigation," in *Proceedings of the 10th IEEE International Conference on Networking, Architecture and Storage, NAS 2015*, pp. 2–11, USA, August 2015.

[101] T. N. Gia, M. Jiang, A.-M. Rahmani, T. Westerlund, K. Mankodiya, P. Liljeberg et al., "Fog computing in body sensor networks: An energy efficient approach," in *Proceedings of the Inernatinal Conference onIEEE International Body Sensor Networks*, pp. 1–7, 2015.

[102] M. Aazam and E.-N. Huh, "E-HAMC: Leveraging Fog computing for emergency alert service," in *Proceedings of the 13th IEEE International Conference on Pervasive Computing and Communication, PerCom Workshops 2015*, pp. 518–523, USA, March 2015.

[103] H. Dubey, J. Yang, N. Constant, A. M. Amiri, Q. Yang, and K. Makodiya, "Fog data: Enhancing telehealth big data through fog computing," in *Proceedings of the ACM conference on ASE BigData & SocialInformatics 2015*, p. 14, 2015.

[104] M. Ahmad, M. B. Amin, S. Hussain, B. H. Kang, T. Cheong, and S. Lee, "Health Fog: a novel framework for health and wellness applications," *The Journal of Supercomputing*, vol. 72, no. 10, pp. 3677–3695, 2016.

[105] S. Wang and S. Dey, "Cloud mobile gaming: Modeling and measuring user experience in mobile wireless networks," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 16, no. 1, pp. 10–21, 2012.

[106] Z. Zhao, K. Hwang, and J. Villeta, "Game cloud design with virtualized CPU/GPU servers and initial performance results," in *Proceedings of the 2012 3rd ACM Workshop on Scientific Cloud Computing, ScienceCloud '12*, pp. 23–30, Netherlands, June 2012.

[107] Y.-T. Lee, K.-T. Chen, Han-I Su, and C.-L. Lei, "Are all games equally cloud-gaming-friendly? An electromyographic

approach," in *Proceedings of the 2012 11th Annual Workshop on Network and Systems Support for Games, NetGames 2012*, Italy, November 2012.

[108] S. Choy, B. Wong, G. Simon, and C. Rosenberg, "Edgecloud: A new hybrid platform for on-demand gaming," Tech. Rep., University of Waterloo, Edgecloud, 2012.

[109] Y. Lin and H. Shen, "Cloud fog: Towards high quality of experience in cloud gaming," in *Proceedings of the 44th International Conference on Parallel Processing, ICPP 2015*, pp. 500–509, chn, September 2015.

[110] J. Bughin, M. Chui, and J. Manyika, "Clouds, big data, and smart assets: Ten tech-enabled business trends to watch," *McKinsey Quarterly*, vol. 56, pp. 75–86, 2010.

[111] S. K. Sharma and X. Wang, "Live Data Analytics with Collaborative Edge and Cloud Processing in Wireless IoT Networks," *IEEE Access*, vol. 5, pp. 4621–4635, 2017.

[112] P. P. Jayaraman, J. B. Gomes, H. L. Nguyen, Z. S. Abdallah, S. Krishnaswamy, and A. Zaslavsky, "CARDAP: A Scalable Energy-Efficient Context Aware Distributed Mobile Data Analytics Platform for the Fog," in *Advances in Databases and Information Systems*, vol. 8716 of *Lecture Notes in Computer Science*, pp. 192–206, Springer International Publishing, Cham, 2014.

[113] I-SCOOP, "From Cloud Computing to Fog Computing," https://www.i-scoop.eu/internet-of-things-guide/#From_the_cloud_to_the_fog_in_IoT.

[114] A. Chakraborty, V. Navda, V. N. Padmanabhan, and R. Ramjee, "Coordinating cellular background transfers using loadsense," in *Proceedings of the 19th Annual International Conference on Mobile Computing and Networking, MobiCom 2013*, pp. 63–74, usa, October 2013.

[115] S. Ha, S. Sen, C. Joe-Wong, Y. Im, and M. Chiang, "TUBE," *Computer Communication Review*, vol. 42, no. 4, pp. 247–258, 2012.

[116] E. Aryafar, A. Keshavarz-Haddad, M. Wang, and M. Chiang, "RAT selection games in HetNets," in *Proceedings of the 32nd IEEE International Conference on Computer Communications (INFOCOM '13)*, pp. 998–1006, Turin, Italy, April 2013.

[117] J. Y. Chung, C. Joe-Wong, S. Ha, J. W.-K. Hong, and M. Chiang, "CYRUS: Towards client-defined cloud storage," in *Proceedings of the 10th European Conference on Computer Systems, EuroSys 2015*, France, April 2015.

[118] L. Canzian and M. V. Der Schaar, "Real-time stream mining: Online knowledge extraction using classifier networks," *IEEE Network*, vol. 29, no. 5, pp. 10–16, 2015.

[119] X. Chen, B. Proulx, X. Gong, and J. Zhang, "Social trust and social reciprocity based cooperative D2D communications," in *Proceedings of the 14th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc '13)*, pp. 187–196, ACM, Bangalore, India, August 2013.

[120] F. M. F. Wong, C. Joe-Wong, S. Ha, Z. Liu, and M. Chiang, "Mind your own bandwidth: An edge solution to peak-hour broadband congestion," https://arxiv.org/abs/1312.7844.

[121] F. M. F. Wong, C. Joe-Wong, S. Ha, Z. Liu, and M. Chiang, "Improving user QoE for residential broadband: Adaptive traffic management at the network edge," in *Proceedings of the 23rd IEEE International Symposium on Quality of Service, IWQoS 2015*, pp. 105–114, USA, June 2015.

[122] Y. Du, E. Aryafar, J. Camp, and M. Chiang, "IBeam: Intelligent client-side multi-user beamforming in wireless networks," in *Proceedings of the 33rd IEEE Conference on Computer Communications, IEEE INFOCOM 2014*, pp. 817–825, Canada, May 2014.

[123] S. Galli, A. Scaglione, and Z. Wang, "For the grid and through the grid: the role of power line communications in the smart grid," *Proceedings of the IEEE*, vol. 99, no. 6, pp. 998–1027, 2011.

[124] K. Vatanparvar and M. A. A. Faruque, "Demo abstract: Energy management as a service over fog computing platform," in *Proceedings of the 6th ACM/IEEE International Conference on Cyber-Physical Systems, ICCPS 2015*, pp. 248-249, USA, April 2015.

[125] M. A. Al Faruque and K. Vatanparvar, "Energy management-as-a-service over fog computing platform," *IEEE Internet of Things Journal*, vol. 3, no. 2, pp. 161–169, 2016.

[126] S. Abdelwahab, B. Hamdaoui, M. Guizani, and A. Rayes, "Enabling smart cloud services through remote sensing: An internet of everything enabler," *IEEE Internet of Things Journal*, vol. 1, no. 3, pp. 276–288, 2014.

[127] 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.

[128] R. Kitchin, "The real-time city? Big data and smart urbanism," *GeoJournal*, vol. 79, no. 1, pp. 1–14, 2014.

[129] C. C. Byers and P. Wetterwald, "Fog Computing Distributing Data and Intelligence for Resiliency and Scale Necessary for IoT: the Internet Of Things (ubiquity symposium)," *Ubiquity*, vol. 2015, no. 4, pp. 1–12, 2015.

[130] S. Wang, Q. Zhao, N. Zhang, T. Lei, and F. Yang, *Virtual Vehicle Coordination for Vehicle as An Ambient Sensing Platform*, IEEE Access, 2018.

[131] N. Lu, N. Cheng, N. Zhang, X. Shen, and J. W. Mark, "Connected vehicles: solutions and challenges," *IEEE Internet of Things Journal*, vol. 1, no. 4, pp. 289–299, 2014.

[132] 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.

[133] O. T. T. Kim, N. Dang Tri, V. D. Nguyen, N. H. Tran, and C. S. Hong, "A shared parking model in vehicular network using fog and cloud environment," in *Proceedings of the 17th Asia-Pacific Network Operations and Management Symposium, APNOMS 2015*, pp. 321–326, Republic of Korea, August 2015.

Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 5018053, 18 pages
https://doi.org/10.1155/2018/5018053

Research Article

Predicting Short-Term Electricity Demand by Combining the Advantages of ARMA and XGBoost in Fog Computing Environment

Chuanbin Li, Xiaosen Zheng, Zikun Yang, and Li Kuang

School of Software, Central South University, Changsha 410075, China

Correspondence should be addressed to Li Kuang; kuangli@csu.edu.cn

Received 26 January 2018; Accepted 25 March 2018; Published 6 May 2018

Academic Editor: Xuyun Zhang

With the rapid development of IoT, the disadvantages of Cloud framework have been exposed, such as high latency, network congestion, and low reliability. Therefore, the Fog Computing framework has emerged, with an extended Fog Layer between the Cloud and terminals. In order to address the real-time prediction on electricity demand, we propose an approach based on XGBoost and ARMA in Fog Computing environment. By taking the advantages of Fog Computing framework, we first propose a prototype-based clustering algorithm to divide enterprise users into several categories based on their total electricity consumption; we then propose a model selection approach by analyzing users' historical records of electricity consumption and identifying the most important features. Generally speaking, if the historical records pass the test of stationarity and white noise, ARMA is used to model the user's electricity consumption in time sequence; otherwise, if the historical records do not pass the test, and some discrete features are the most important, such as weather and whether it is weekend, XGBoost will be used. The experiment results show that our proposed approach by combining the advantage of ARMA and XGBoost is more accurate than the classical models.

1. Introduction

In recent years, with the rise of Cloud Computing [1, 2], more and more computing and storage processing are taking place in Cloud, and the vast employment of Cloud inevitably leads to high latency, network congestion, and low reliability. At the same time, with the wide adoption of IoT services, a variety of household appliances and sensors will be connected to the Internet and produce a large amount of data [3–5]. It has been estimated that the number of devices connected by IoT will reach 50 billion to 100 billion by 2020, which means there will be more and more data without the control of existing techniques on data processing and analysis, privacy leaks may be caused, and the quality of service will be decreased [6, 7]. In this regard, the rapid development of IoT has deepened the dilemma of Cloud Computing. The emergence of Fog Computing makes up for these shortcomings but also brings new opportunities and challenges to the transformation and upgrading of traditional industries. Electricity system, which aims at providing enterprises with safe, reliable, and high-quality electric power, has become an indispensable part in the construction of national economy and people's life, so it is affected at first. Under the current technical conditions, it is still not possible to achieve large-scale storage of electric energy; therefore, it is required to generate electricity according to the system load at any time, or else the quality of electricity supply and usage may be affected, and even the safety and stability of the system may be endangered. It has become an urgent and important research issue to improve the accuracy of electricity demand prediction in Fog Computing framework.

In the field of electricity demand prediction, scholars have carried out extensive research. In the early stage, scholars basically followed the technology in the field of economic prediction, focusing on the rule of the load sequence in the form of time series itself. The prediction model is established by analyzing the qualitative relationship between the historical load and related factors, and the parameters are estimated

according to historical data. However, time series based approach requires historical data with high accuracy, insensitive to the factors such as weather and holidays. Actually, it is very difficult to express the nonlinear relation between the input and output by using a clear mathematical equation since the electricity data are nonlinear, time-varying, and uncertain. In order to further improve the accuracy of electricity demand prediction, artificial intelligence methods have been applied since the 1990s, such as neural network, expert system, and wavelet analysis. However, the existing methods usually can just be applied to a limited scenario and only effective for simple electricity systems with a small quantity of factors.

The main deficiencies of existing work include the following: (1) electricity data samples have been proven to have anomalies but few of the existing solutions detect and deal with the abnormal data. (2) When classifying users and their data, the number of clusters needs to be specified in advance, and the density distribution on electricity consumption is usually ignored in clustering users. (3) In the process of modeling, the temporal features of data are not fully excavated, and the interaction among the features is not fully considered. (4) Most of the solutions just adopt a sole model, which cannot give full play to the advantage of each model.

In order to overcome the shortcomings of existing solutions, in this paper, we propose a short-term electrical-demand prediction approach by combining the advantages of XGBoost and ARMA in Fog Computing framework. Firstly, sensors collect the real-time data of electricity consumption, and then the Fog Nodes would classify enterprise users into different groups according to the amount of electricity consumption and perform anomaly and outlier data detection and procession for each group. Secondly, a model selection process would be performed, that is, based on a series of tests including stationarity, white noise and Pearson correlation coefficient of data, as well as the observed electricity consumption rule; we will decide whether to use time series model or decision tree based model for the modeling of each enterprise group. Finally, the prediction values of each enterprise user are combined to obtain the final result. The accuracy of the proposed approach is verified to be 20% higher than classical models by experiments.

The rest of the paper is organized as follows: Section 2 reviews the related work on predicting short-term electricity demand. Section 3 describes the framework of the proposed solution, as well as the details of the involved key techniques. Section 4 introduces the experiment and analyzes the results. Section 5 summarizes our work and provides the future research plan.

2. Related Work

Because of the nonlinear, time-varying, and uncertainty characteristics of electricity data, it is difficult to accurately grasp the related factors and the rules of electricity consumption change. How to effectively improve the accuracy of electricity demand prediction has become a major challenge to researchers [8]. At present, the methods used for short-term electricity demand prediction mainly include time series [9–11], Regression Analysis [12, 13], Support Vector Regression [14–16], Neural Network [17–20], Bayes [21], Fuzzy Theory [20, 22], and Wavelet Echo State Network [23]. Each kind of method has its own applicable scenario, and no model can achieve desired satisfying result alone.

In order to improve the accuracy of prediction, the current research works mainly focus on three directions. The first one is to explore the optimization of single model. Zhu et al. [24] propose to predict daily load roughly by ARMA first, then obtain the difference sequences that are noncyclical and strongly influenced by the weather, and finally propose an improved ARIMA prediction model with strong adaptability to weather. Factors which influence the electricity consumption are recognized and the mapping relation between key factors and electricity consumption are mined [25, 26]. Ghelardoni et al. [27] use the empirical mode decomposition method to divide the time series into two parts, describing the trend and the local oscillation of energy consumption values, respectively, and then use them to train the support vector regression model. Che et al. [28] use the human knowledge to construct fuzzy membership functions for each similar subgroup and then build an adaptive fuzzy comprehensive model based on self-organizing mapping, support vector machine, and fuzzy reasoning for prediction. An electricity load model is established based on improved particle swarm optimization algorithm and genetic algorithm [29, 30].

The second direction is to improve the accuracy of prediction by integrating different models. Haque et al. propose a hybrid intelligent algorithm based on wavelet transform and fuzzy adaptive resonance theory [31]. In [32–34], the wavelet decomposition is used to project the load sequence decomposition onto different scales, different models are used to predict the different components, and finally the final result is obtained by reconstructing the components. Pindoriya et al. [35] propose an adaptive wavelet neural network (AWNN) for short-term price prediction in the electricity market. Pany and Ghoshal [36] propose a local linear wavelet neural network (LLWNN) model instead of wavelet neural network for the electricity price prediction. Che and Wang [37] propose a hybrid model that combines the unique advantages of SVR and ARIMA models in both nonlinear and linear modeling.

The third direction is to explore composite models for prediction. The weighted average of all the results by various algorithm is usually used, and there are two kinds of ways to determine the weights. The first kind is to improve the fitting accuracy of historical electricity consumption by minimizing the fitting error. The main methods include monotone iterative algorithm [38], evolutionary programming [39], and quadratic programming [40]. Wang et al. [41] propose using adaptive practical swarm optimization algorithm to optimize the weight of the integrated model. The second kind is to determine the weights by evaluating the algorithm's score. Elliott and Timmermann [42] introduce the concept of the loss function, quantify the negative impact caused by different prediction errors, then take the minimum loss expectation as the goal, and perform optimization to get the weights. Yao et al. [43, 44] employ analytic hierarchy process (AHP) in multiobjective decision analysis to get the relative merits of each algorithm in fitting accuracy, model adaptability,

Table 1: Examples of the enterprise electricity consumption dataset.

record_date	user_id	power_consumption
2015/1/1	1	1135
2015/1/2	1	570
2015/1/3	1	3418
2015/1/4	1	3968

and result reliability, the judgment matrices are obtained, and then the weights are combined by calculating the main eigenvectors of each matrix. Petridis et al. study the use of probability models to determine the weight of each model and combine the values of each algorithm to obtain the final result [45]. Without enough quantitative theoretical basis, the weights of such models only reflect the advantages and disadvantages of the algorithms.

In summary, there have been many research works on the prediction of short-term electricity demand, and exploring composite models for prediction is the main trend. However, existing methods still have limitations. In the context of the rapid development of smart electricity grid, in this paper, we propose a short-term electricity demand prediction method based on XGBoost and ARMA.

3. Predicting Short-Term Electricity Demand

3.1. Problem Definition. Given a dataset $R = \{r_1, r_2, r_3 \cdots r_n\}$, r_i is the ith electricity consumption record for a certain enterprise user, and r_i can expressed as a 3-tuple, that is, $r_i = \langle$record_date, user_id, power_consumption$\rangle$, where record_date represents the date time, user_id is the ID of the enterprise user, and power_consumption represents the electricity consumption amount of the enterprise user on that day.

We use the dataset from Tianchi, which contains the historical records of 1454 enterprises in Yangzhong High-Tech Industrial Development Zone of Jiangshu Province from 2015-01-01 to 2016-11-30, for the following illustration and experiments. Examples of the dataset are shown in Table 1.

Given the dataset R and a month time in future, we aim to predict the total amount of the electricity demand on the desired month in this region on the basis of historical records.

3.2. Framework of the Proposed Approach

3.2.1. Smart Electricity System in Fog Computing. Fog Computing framework has the advantages of low latency, saving core bandwidth, and high reliability [46, 47]. Fog Nodes are located lower in the network topology and thus they have less network latency and more reactivity. As an intermediate between Cloud and terminals, Fog Layer can filter and aggregate enterprise messages and only send the necessary messages to Cloud, thus reducing the pressure on the core network. In order to serve enterprises in different regions, the same services will be deployed on the Fog Nodes in each region. Once the services in a certain area are abnormal, the requests can be quickly forwarded to other same services nearby, which makes the framework highly reliable.

As an extension of Cloud Computing, the framework preprocesses enterprise data and makes real-time decision and provides temporary storage to enhance the users' experience. In the current electricity systems, the number of hops from enterprise terminal to Cloud is generally 3 to 4 or even more, so the system will have to face the network delay when making real-time decisions. Figure 1 shows the framework of a smart electricity system in Fog Computing. Electricity meters collect data as sensors, and, for some enterprises with major changes of data and high real-time requirement, we can make real-time decisions in Fog Nodes to meet the need of real-time electrical-demand prediction; otherwise, the data can be buffered at Fog Nodes, compressed to save network bandwidth, and then transmitted to the Cloud.

3.2.2. Process of Electricity Demand Prediction Approach. Figure 2 shows the process of electricity demand prediction approach based on multimodel fusion algorithm, which includes five main steps.

(1) Data Preprocessing. After data are collected, the missing values will be filled and their form will be unified.

(2) Enterprise Users Clustering. The size of electricity demand for each enterprise user is measured by counting the total amount of its historical electricity consumption. Then enterprises users will be divided into different groups by clustering them according to their sizes of electricity demand.

(3) Model Selection. We then aim to determine an appropriate training model for each group of enterprise users. First, the rules of electricity consumption for different groups of enterprise users are analyzed for prejudgement of model selection. If the electricity consumption changes with time, showing a periodical change or an obvious rising/falling trend, we consider to model the users' data by XGBoost. If the electricity consumption shows an irregular change over time or fluctuates around a certain constant and the fluctuation range is limited, we consider to model the users' data by time series model.

Second, a series of tests would be performed to verify the prejudgement and finally determine the selected model for each group of users. Feature correlation analysis and feature importance scores would be performed before selecting XGBoost modeling. Stationary and white noise test would be performed before selecting ARMA modeling. If neither XGBoost nor ARMA is appropriate, mean model will be used.

(4) Model Building. After model is selected for a given group of users, data cleaning is performed first, including the detection and processing of anomalies and outliers. For XGBoost modeling, anomalies and outliers will be deleted first, the influence of time factor and temperature on the prediction will be considered emphatically, Pearson correlation coefficient will be used to identify redundant features, and appropriate features will be selected to build the model by combining the feature importance outputted by pretraining. For AMRA modeling, anomalies and outliers will be modified by average value, and parameters p and q will be

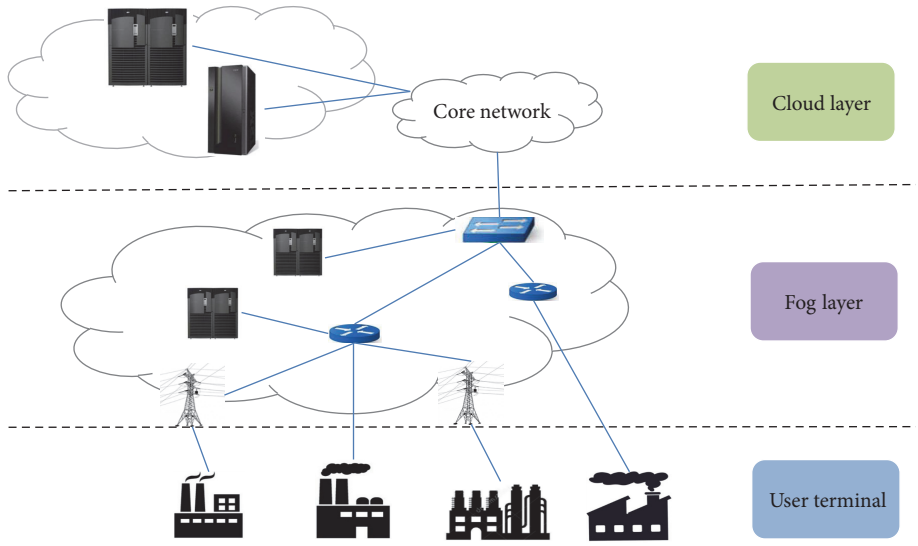


Figure 1: Smart electricity system in Fog Computing.

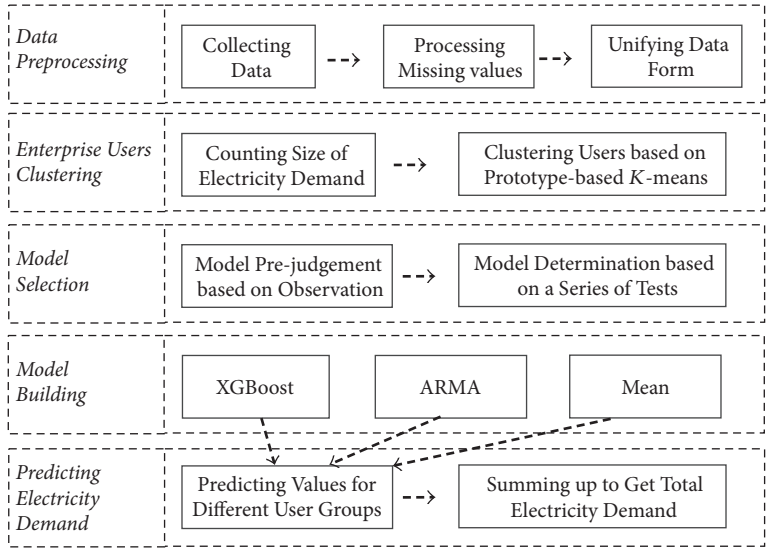


Figure 2: Process of electricity demand prediction approach.

optimized based on the minimum amount of information principle of BIC.

(5) Predicting Electricity Demand. After modeling for different enterprise groups, the prediction values of each group will be summed up to get the final daily electricity demand in the desired month of this region.

3.3. Key Techniques. In this section, we will elaborate on the detailed key techniques in the five steps of the electricity demand prediction approach.

3.3.1. Data Preprocessing

(1) Collecting External Weather Data. Taking into account the impact of temperature on electricity consumption, we first collect external weather data from http://lishi.tianqi.com. Samples are shown in Table 2.

(2) Processing Missing Value. We find that there are some missing values in Tianchi dataset, so we then fill the missing values with the mean value of the three days before and after the date with missing value. The detailed calculation is shown as follows:

$$c_{ij} = \frac{\left(c_{i,j-3} + c_{i,j-2} + c_{i,j-1} + c_{i,j} + 1, c_{i,j+2}, c_{i,j+3}\right)}{6}, \quad (1)$$

in which c_{ij} represents user i's missing record on the jth day of the year.

(3) Unifying Data Form. The electricity consumption records are reorganized to facilitate the follow processing. Each

Table 2: Samples of weather information.

Record_date	Highest temperature	Lowest temperature	Conditions	Wind direction	Wind force
2016/10/12	21	16	Overcast	West	breezy
2016/10/13	20	16	Partly Cloudy	Calm	light breeze
2016/10/14	19	17	Mostly Cloudy	West	light breeze
2016/10/15	20	16	Showers	East	breezy
2016/10/16	23	17	Overcast	NorthWest	light breeze

column represents records of an enterprise user, and there are totally 1454 users. Each row represents the records on a certain day, and the dates are sorted in ascending order. Each grid represents the electricity consumption of a specific user on a certain day, which can be expressed as $r_{ij} = \{\text{Record\_date}_j, \text{Power}_i, c_{ij}\}$. The unified data sheet is shown as Table 3.

3.3.2. Clustering Users by Prototype-Based K-Means Algorithm. Clustering analysis can find locally strongly related object groups. Outlier detection can detect objects that are significantly different from most objects by detecting the discreteness of the data. Based on the characteristics of the two techniques, in this paper, we use a prototype-based clustering method to detect the degree of data discreteness, so as to learn the distribution of the data and then determine the range of k first, and next use K-means algorithm to cluster enterprise users in order to achieve enterprise groups division.

The principle of the prototype-based clustering method is to cluster all the objects first and then evaluate the degree that the objects belong to the clusters according to the distance. In traditional method, the distance between the object and the cluster center is used to measure the degree that the object belongs to the cluster. In this paper, we consider the density of data distribution and adopts the relative distance. Based on this idea, we design the clustering algorithm for enterprise groups division as shown in Algorithm 1.

We will take the Tianchi dataset as an example to illustrate the process of prototype-based K-means clustering algorithm:

First, we calculate the historical electricity consumption of each enterprise from 01/01/2015 to 10/31/2016, and cluster all the samples by using prototype-based algorithm. We take the relative distance to measure the discrete degree of samples and choose the threshold λ as 20, so the sample with relative distance greater than 20 is deemed as an outlier. Figure 3 shows the discrete points with relative distances.

Each of the points in Figure 3 is marked with a pair, which indicates the enterprise ID and its relative distance to the cluster center. There are 7 points with higher dispersion that are 1416, 175, 174, 90, 129, 1262, 1307, and 1310 from high to low. According to Figure 3, the data are generally distributing in four distance segments. User 1416 is located at the top of the figure, whose relative distance is greater than 400, much higher than the other points, and it should be classified to the first category. At the middle of the figure, 175 and 174, whose relative distances are between about 150 and 200, should be classified to the second category. The users with ID 90, 129, 1262, 1307, and 1310, whose relative distances are between 100 to 20, should be the third category. And the remaining 1447 enterprise users, whose relative distances are below 20, should be the fourth category. So we set $k = 4$ when using K-means algorithm. And the final result of users clustering is shown as Table 4.

3.3.3. Model Selection. Model selection is the basis of next step and it is especially important. Figure 4 shows the process and rules of model selection, which mainly includes four parts.

(1) Data Preparing. The data now mainly include the enterprise user ID, its group ID, the history electricity consumption, and the weather data from 2015/1/1 to 2016/11/30.

(2) Model Prejudgement Based on Periodicity/Trend and Nonlinear/Weak Stationary Analyzing

(2a) Periodicity/Trend Analyzing. Periodicity analyzing is to find out whether electricity consumption of a user will change periodically with time. If a user's electricity consumption shows regular fluctuations, the increasing speed is the similar at both sides of the wave peak, and the appearance of peak is strongly related to time period, that is, $\text{Power}(n) = \text{Power}(n + T)$, where T is the time span; then we can make a prejudgement that XGBoost model would be appropriate with this character.

Trend analyzing is to find out whether change pattern of electricity consumption will show some trend. If a user's electricity consumption shows a tendency to rise or fall, or the appearance of peak changes with time and shows a relation of positive and negative correlation, then we can also consider using XGBoost model for this character.

(2b) Nonlinear/Weak Stationary Analyzing. If the user's electricity consumption presents irregular change or mutation or no obvious change, it shows that the sequence of electricity consumption is nonlinear. If the user's electricity consumption shows a fluctuation around a certain constant and the fluctuation is approximately in the same range, it can be seen that the mean and variance of the data are all constant, which indicates that the data has no obvious trend. If the mean value has nothing to do with the change of time, and the influence among the sequence variables is almost the same after delaying k periods, it reflects the weak stationary of data. So we can make a prejudgement that time series modeling like

Table 3: Unified data sheet.

Record_date	Power 1	Power 2	Power 3	Power 4	Power 5	Power i	Power 1454
2015/1/1	1135	24	385	206	156	⋯	976
2015/1/2	570	22	475	5134	368	⋯	520
2015/1/3	3418	18	526	6784	359	⋯	536
2015/1/4	3968	119	1	6475	467	⋯	488
2015/1/5	3986	108	535	6592	433	⋯	406
2015/1/6	4082	109	663	6742	452	⋯	554
2015/1/7	4172	86	606	6963	467	⋯	803
2015/1/8	4022	106	735	6301	495	⋯	975
2015/1/9	4025	103	698	6693	542	⋯	796

Input: Historical electricity consumption records R
Output: the group division of enterprise users ClusterSet(Power$_i$)
(1) /*Calculate the historical total electricity consumption of each enterprise*/
INPUT: R = {Record_date$_j$, user$_i$, c_{ij}}
FOR i = 1 To length(user$_i$)
DO
SumPower$_i$ = 0
FOR j = 1 To length(Record_date$_j$)
DO
SumPower$_i$ = SumPower$_i$ + c_{ij};
END
END
OUTPUT: {SumPower$_i$}
(2) /*Select the clustering algorithm (K-means) to cluster all the objects and then choose k = 1 to cluster all the samples into a cluster and find the centroid of the cluster*/
INPUT: {SumPower$_i$}
DO
Select K = 1
ClusterCenter = Kmeans (SumPower$_i$)
END
OUTPUT: ClusterCenter
(3) /*Calculate the relative distance*/
INPUT: ClusterCenter, {SumPower$_i$}
FOR i = 1 To length(user$_i$)
DO
AbsoluteDistance$_i$ = SumPower$_i$ − ClusterCenter
MedianDistance$_i$ = median(AbsoluteDistance$_i$)
RelativeDistance$_i$ = AbsoluteDistance$_i$/MedianDistance$_i$
END
OUTPUT: {RelativeDistance$_i$}
(4) Make discrete point relative distance error figure and roughly determine the range of k accord to the relative distance
(5) /*Use K-means algorithm to do clustering for enterprises, and get the groups of enterprises according to the result of clustering*/
INPUT: {⟨user$_i$, SumPower$_i$⟩}, k
DO
Select K = k
ClusterSet(user$_i$) = Kmeans(user$_i$)
END
OUTPUT: ClusterSet(user$_i$)

Algorithm 1: Prototype-based K-means clustering algorithm

Table 4: Clustering result.

Cluster ID	Enterprise user ID	Semantic meaning
1	1416	400 < relative distance, large enterprise
2	174, 175	150 < relative distance < 200, Second large enterprise
3	90, 129, 1262, 1307, 1310	20 < relative distance < 100, medium enterprise
4	Other enterprises except 1416, 174, 175, 90, 129, 1262, 1307, 1310	0 < relative distance < 20, small enterprise

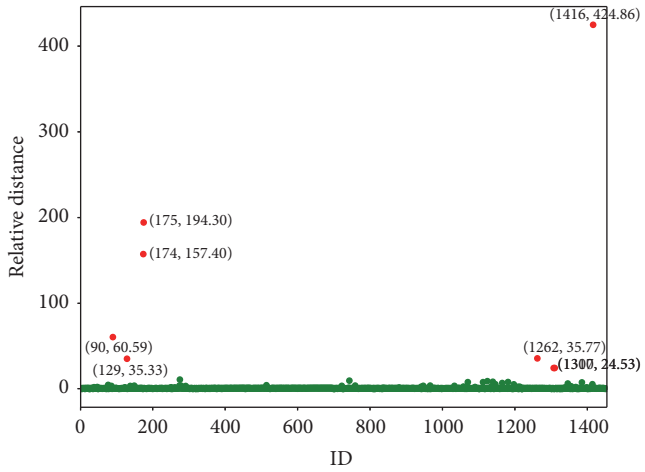


Figure 3: Discrete points with relative distances.

ARMA would be appropriate for such kind of data with weak stationary.

(3) Model Verification for XGBoost. If XGBoost is prejudged as an appropriate choice for a user, a series of tests including Pearson correlation coefficient and feature importance analyzing would be performed before finally deciding to use XGBoost.

(3a) Feature Generating. Feature is the information extracted from the data that is useful in prediction. Feature extraction is mainly based on the existing background knowledge, so that the feature can play a better role in the machine learning algorithm. Based on thorough analysis of the features, we mainly extract the time and weather features as the input features.

(3b) Correlation Analyzing. We then use Pearson correlation coefficient test to analyze the correlation between the features and the prediction value. We use the results of feature generation and electricity consumption data for testing. If the correlation between the prediction target and some discrete features, such as weather, holidays, workday/nonworkday, is strong, it is better to use XGBoost for modeling.

(3c) Feature Importance Analyzing. Feature importance analysis refers to analyzing the importance relationship between each feature and the target value and studying the influence of each feature on the change of the target. By outputting the importance of all features through the pretraining of XGBoost model, if the factors which cannot be characterized in time series, such as weather, vacation, and workday/nonworkday, have high scores, it can be determined to use XGBoost model.

If both tests have low scores, it indicates that the user's data are not suitable for XGBoost model, and we can turn to test whether they are suitable for the time series (ARMA) model.

(4) Model Verification for ARMA. If ARMA is prejudged as an appropriate choice for a user, or if XGBoost is not suitable, we then perform a series of tests including Pearson correlation coefficient, stationarity testing, and white noise testing to verify whether ARMA is suitable.

(4a) Correlation Analyzing. Same as in (3b), we test the correlation between features and the target value by using Pearson's correlation coefficient. If the correlation between the target and discrete features is low, we can consider to use the time series (ARMA) model preliminarily.

(4b) Stationarity Testing. Stationarity is the prerequisite for time series modeling. If the data is stationary, the fitting curve obtained through the sample time series can still inertially continue for a period of time in the future. If the data is not stationary, it indicates that the shape of the sample fitting curve does not have the characteristic of "inertia" continuation; that is, the curve fitting based on the sample time series to be obtained in the future will be different from the current sample fitting curve. To test the stationarity of data, we use the unit root for inspection. If the electricity consumption sequence data has unit root, the data is stationary.

(4c) White Noise Testing. White noise sequence is a stationary random sequence without any information. If the sequence is white noise, it indicates that there is no relationship between the values of sequence, and it is a purely random sequence. The autocorrelation coefficient is equal to zero, that is, $\lambda(k) = 0, k \neq 0$. If the white noise test is passed, it shows that the sequence is a non-white noise sequence.

If both stationarity and white noise testing are passed, we can determine that ARMA model is suitable for modeling the user's data.

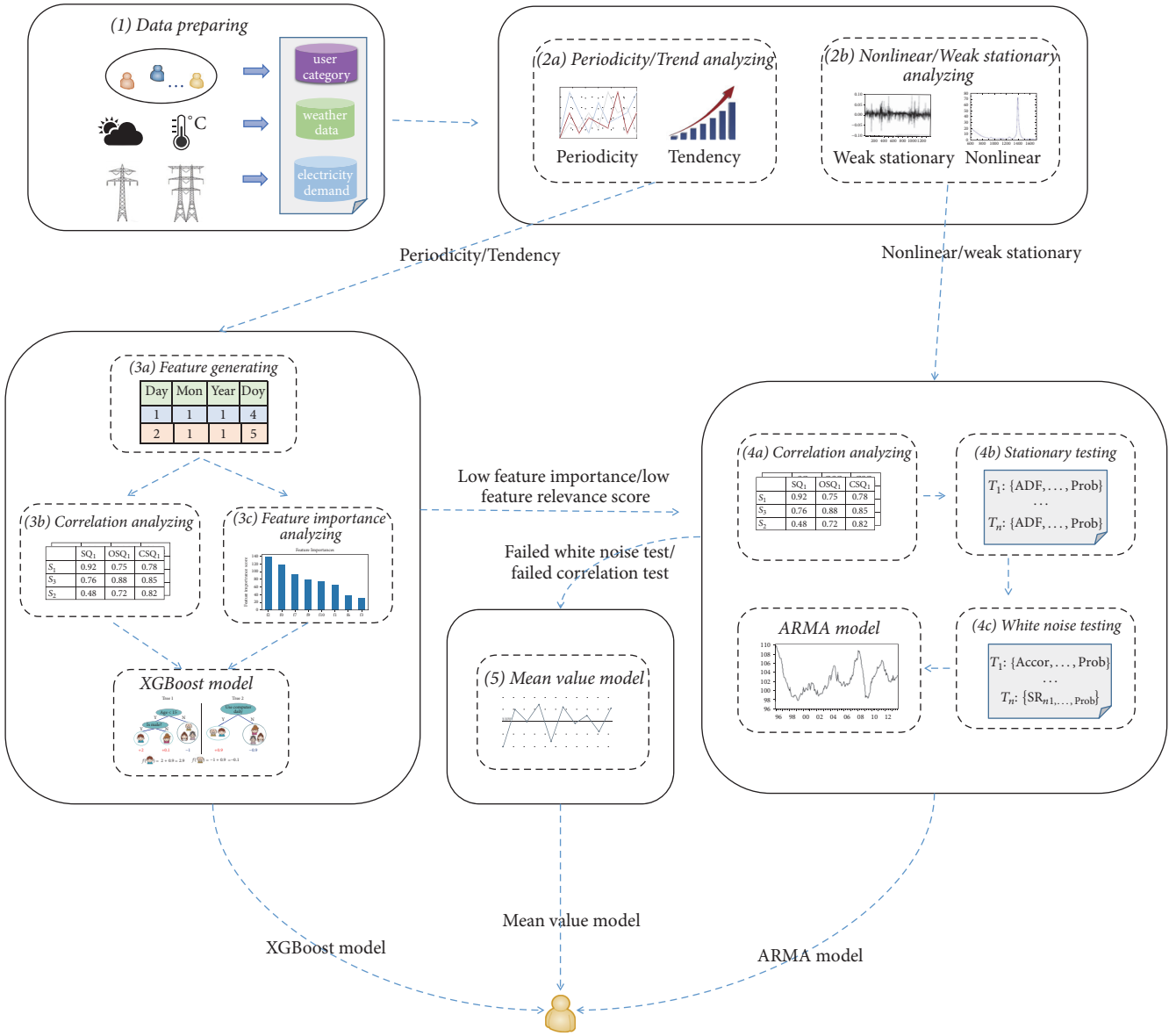


Figure 4: Framework of model selection.

(5) Mean Value Model. If neither XGBoost nor ARMA is suitable for modeling, we then use the mean model as a final choice. The mean model takes the mean of the historical electricity consumption data as the prediction value.

We then show the process of model selection by two examples: one is a small enterprise and another one is the enterprise with ID 1307. Figure 5 shows the daily electricity consumption amount of the 1307th enterprise from 1/1/2015 to 10/31/2016, in which the x-axis is represented as the xth day in the period.

From Figure 5, we can find that the curve of the 1307th enterprise fluctuates on the mean value line from 2015 to 2016 and the fluctuation amplitudes are almost the same, which is consistent with characteristics of weak stationarity and nonlinear. So it can be prejudged to employ ARMA for modeling.

Figure 6 shows the data of a small enterprise.

From Figure 6, it can be seen that the curve presents a cyclical fluctuation with the change of time. Through analysis, we can find that curve presents a "W" shape, it is symmetric on the 1/1/2016, and every small peak fluctuates in the unit of week. The data wave within each week is like a convex line, showing a high middle and low sides. It can be judged that the electricity consumption of small enterprises is greatly influenced by the year and week. So it can be prejudged to use XGBoost for modeling.

From the analysis above, we can find out that most enterprises are highly correlated with time features, so we can

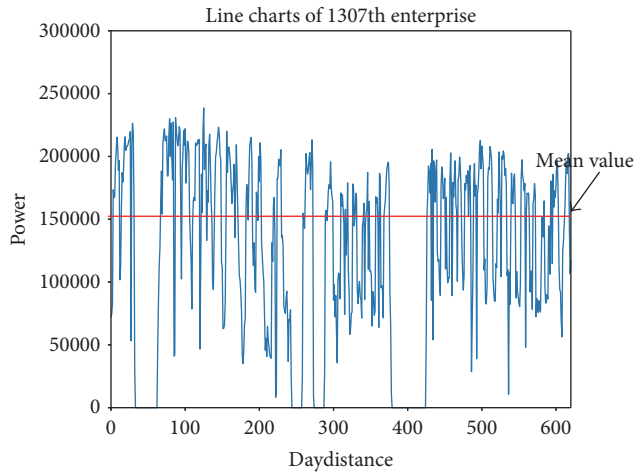


Figure 5: Daily electricity consumption amount of the 1307th enterprise.

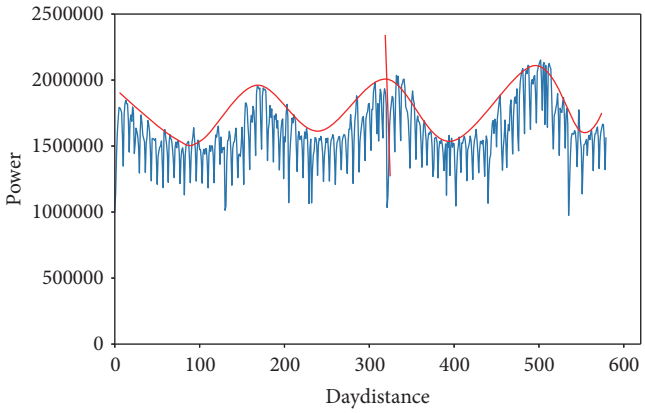


Figure 6: Daily electricity consumption amount of a small enterprise.

extract temporal features [48] as attributes for feature construction. Sahay et al. [49] introduced the influence of temperature to electricity consumption, so we also consider the effect of temperature. Then we use Pearson correlation coefficient to test the correlation between electricity consumption and features for enterprises from each group.

The main factors which may cause changes in electricity consumption are specifically shown in Table 5.

Figure 7 is the Pearson correlation coefficient test result of 1307th enterprise.

The correlation coefficient matrix is a symmetric matrix. The correlation between the feature and the target can be regarded as the importance of the feature, which is more important if it is closer to 1 or −1. From Figure 7, we can find that the scores of time features for the 1307th enterprise are relatively low, so it can be inferred that its electricity consumption characteristics are irrelevant to the time features. Next we perform stationarity test and white noise test on the data of the 1307th enterprise.

Table 6 shows the test results of the 1307th enterprise.

From Table 6, after smooth processing, we can find that the p value of the unit root test statistic (0.0089) of the series is significantly less than 0.01, so the original hypothesis is strictly rejected, which judges that the series is a stationary sequence. The p value of white noise test is significantly less than 0.01, so we strictly reject the original hypothesis, which judges that the logarithmic processed series is a stationary non-white noise sequence. Combined with the prejudgement and the result of the three tests, we can determine to choose ARMA for modeling the data of the 1307th enterprise.

From Figure 8, we can find that the scores of time features for the small enterprise are relatively high, so it can be inferred that the electricity consumption characteristics of small enterprises are highly correlated with the time features. In particular, the feature "holidays" has the strongest correlation, while time series model cannot make full use of temperature and holiday features. So time series modeling is not suitable for small enterprises.

We then calculate feature importance scores for the small enterprise by pretraining in XGBoost, and the result is shown in Figure 9.

After training, the XGBoost sorts the importance of features from high to low and the result is doy, daydis, dow, maxt, mint, dom, holiday, and woy (f2, f7, f0, f9, f10, f1, f6, and f3). We can find out that small enterprises are influenced by doy, dow, daydis, maxt, and mint greatly, which is consistent with previous analysis.

Combining the prejudgement with the test of Pearson correlation coefficient and feature importance scores, we can determine that XGBoost modeling is more suitable for small enterprises.

3.3.4. Abnormal Data Processing. Data quality is crucial for the performance of models. A large number of abnormal data in the original data may lead to the deviation of the result, so it is necessary to clean the data. Missing value processing has been done in the data preprocessing, and then, in this part, we mainly perform the detection and processing of abnormal data.

In order to detect abnormal data, outlier detection is usually used to find out the values which obviously deviate from most of the samples. For some enterprises, we use pretraining modeling to mark the sudden points which deviate too much from the fitted curve as the outlier. Prototype-based clustering outlier detection method is used to detect the outliers which are deviating from the centroid obviously. This outlier detection algorithm also adopts the relative distance which is similar to Algorithm 1. And there is a little difference; that is, the outlier detection algorithm based on prototype clustering filters the outliers by using the appropriate threshold value a in the 4th step and outputs the detected outliers.

Figure 10 shows the original daily electricity consumption amount of the 175th enterprise.

It can be seen from Figure 10 that the electricity consumption of the 175th enterprise has a large difference between the first year and the second year, so segment detection will be used. We take year as a unit to carry out segment detection. Figure 11 shows the results of outlier detection based on prototype-based clustering.

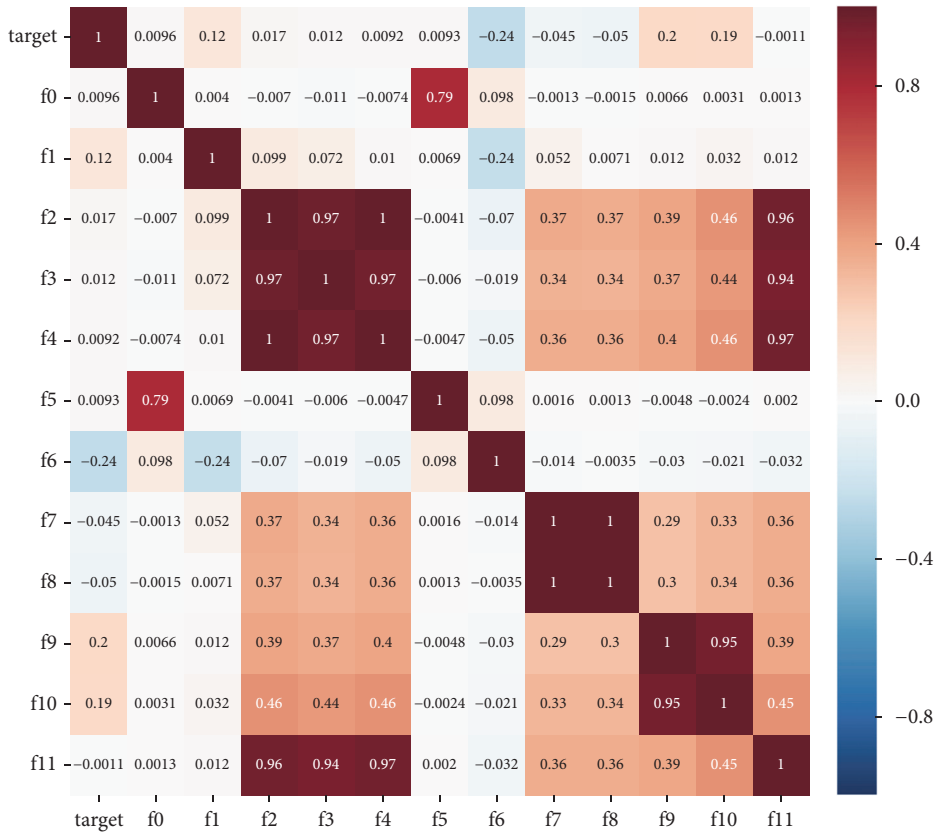


Figure 7: Pearson correlation coefficient test result of the 1307th enterprise.

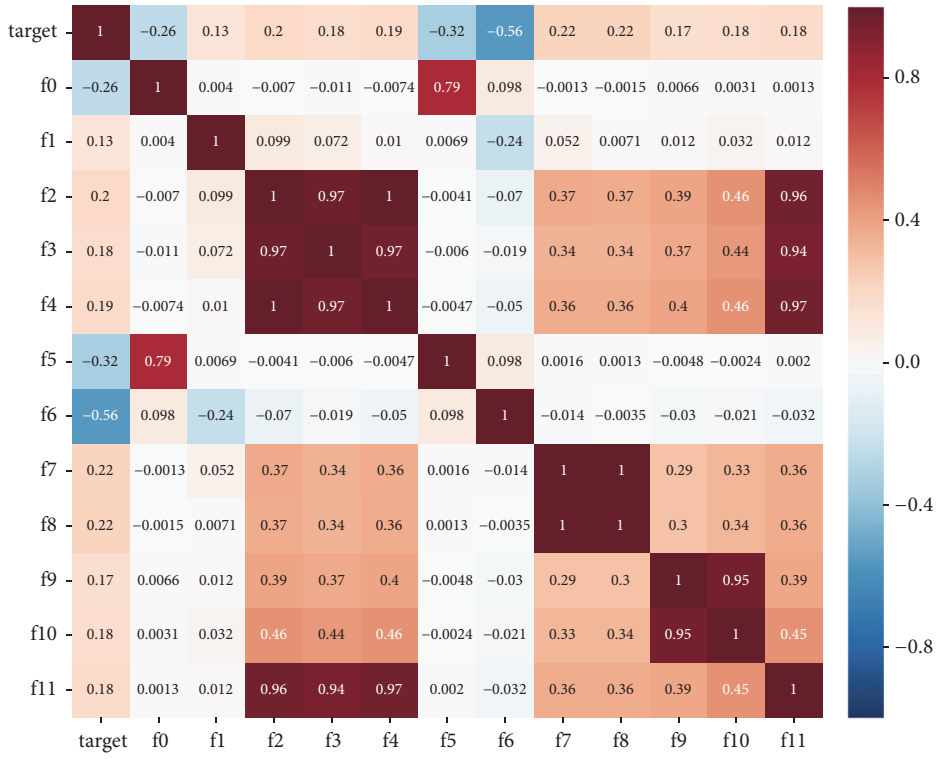


Figure 8: Pearson correlation coefficient test result of a small enterprise.

Table 5: Features.

Feature number	Feature name	Feature Description
F0	dow	Day of the week, integer, between 0 and 6
F1	dom	Day of the month, integer, between 1 and 31
F2	doy	Day of the year, integer, between 0 and 365
F3	woy	Week of the year, integer, between 1 and 52
F4	moy	Month of the year, integer, between 1 and 12
F5	weekend	Whether the day is a weekend, 0 for weekdays, and 1 for weekends
F6	holiday	Whether the day is a holiday, 0 is a holiday, 1 is not a holiday
F7	daydis	The distance between this day and the first day
F8	mondis	The distance between this month and the first month
F9	maxt	The highest temperature
F10	mint	The lowest temperature
F11	season	Season of the year, integer, between 1 and 4

Table 6: The test results of the 1307th enterprise.

The ADF test result	0.0089
The white noise test result	$1.3282e-142$

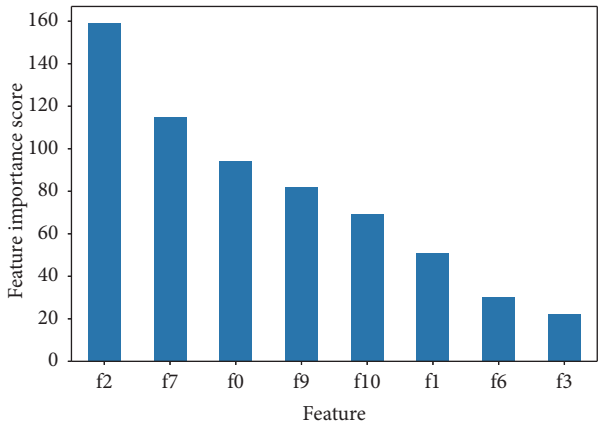


Figure 9: The importance score of each feature for the small enterprise user.

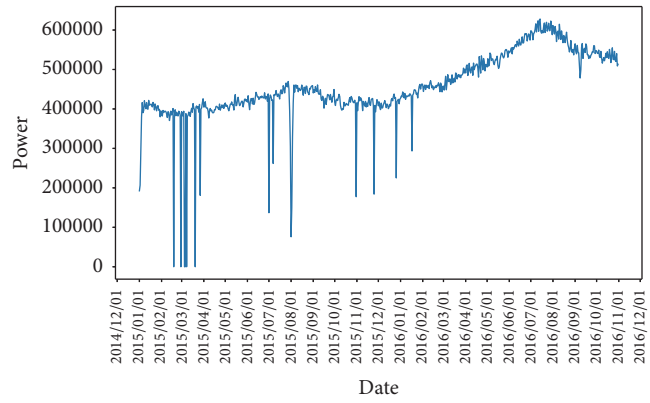


Figure 10: Electricity consumption amount of the 175th enterprise.

In order to facilitate data detection, data is numbered according to the date, which is the distance from 1/1/2015 to that day. For the first year, we use threshold $a_1 = 5$; that is, the points with relative distance larger than 5 are deemed as outliers. For the second year, we use threshold $a_2 = 2.6$; that is, the points with relative distance larger than 2.6 are detected as outliers. From Figure 11, we can see that the red dots are significantly deviated from the data centroid, which are abnormal data, and green spots are normal data.

For abnormal data, we adopt different strategies for different models. If XGBoost is used, the abnormal value will be deleted directly; otherwise, if ARMA or mean value model is used, the abnormal value will be modified by the average value of three days before and after the abnormal value day.

For the 175th enterprise, since it is determined to use XGBoost model, we delete the abnormal values directly. Figure 12 shows the daily electricity consumption of the 175th enterprise after data cleaning. From Figure 12, we can see that the curve becomes smooth after data cleaning. It is obvious that the electricity consumption pattern changes softly in the first year. And in the second year, there is an obvious small peak, showing the increase of electricity consumption.

3.3.5. Building ARMA Model. The basic idea of ARMA is, according to a stationary time series, which may be differential or logarithmic, processed to be a stationary series if necessary, a model is built to describe the stochastic process, and then the best prediction value of future time would be obtained by the built model and observed time series values.

The modeling process of ARMA is shown as Figure 13. It mainly consist of four steps.

(1) The square root test (ADF) is used to test the stationarity of the series. If the series is stationary, the white noise test will be performed. Otherwise, differential or logarithmic operation will be used to make it as a stationary series.

(2) The white noise test is performed. If the series is a stationary random series that has no information to extract, we quit the process. If the series passes the white noise test, which shows the series is a stationary non-white noise series, it can be modeled by the ARMA.

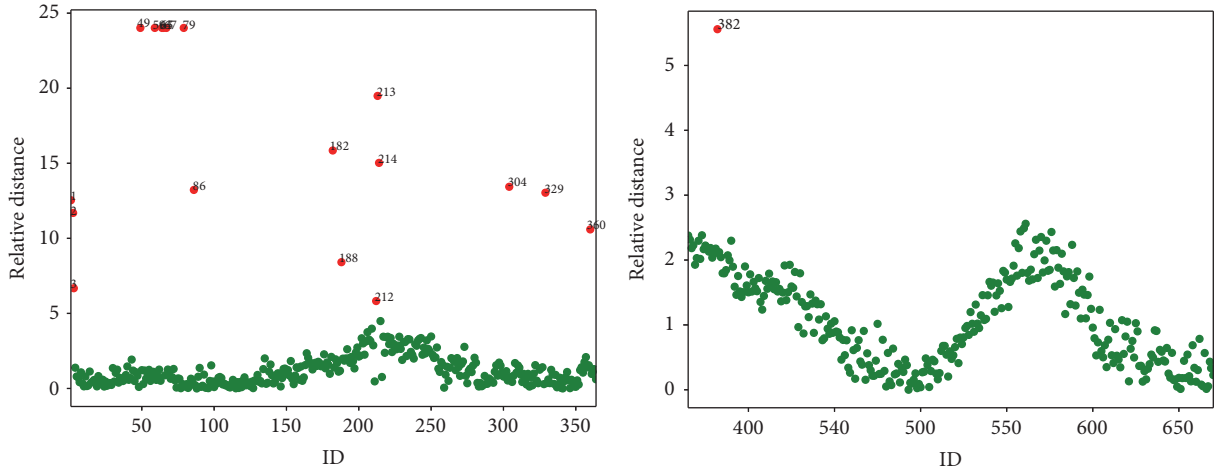


Figure 11: Outlier detection result (threshold $a_1 = 5$, $a_2 = 2.6$).

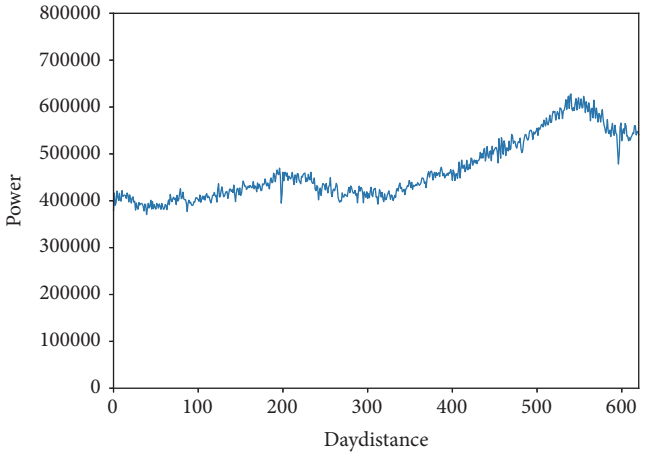


Figure 12: Daily electricity consumption of the 175th enterprise after data cleaning.

(3) Using parameter optimization, we determine p, q based on the minimum amount of information principle of BIC.

(4) Predict the electricity demand using the built AMRA model.

As we mentioned before, data of the 1307th enterprise is suitable for ARMA model. In data processing, logarithmic processing is done. The logarithmic processing can make the data smooth and make the data more stationary without changing the trend of the data. According to the result of ADF test, we judge that the series of 1307th enterprise is a stationary non-white noise series, and then we use parameter optimization to determine p, q based on the minimum amount of information principle of BIC.

The fitting result of the 1307th enterprise by ARMA is shown in Figure 14. The blue curve represents its actual electricity consumption. The red one represents the fitting line of ARMA (2, 0). From Figure 14, we can see that the prediction values of ARMA model are basically consistent, and the fitting performance is good.

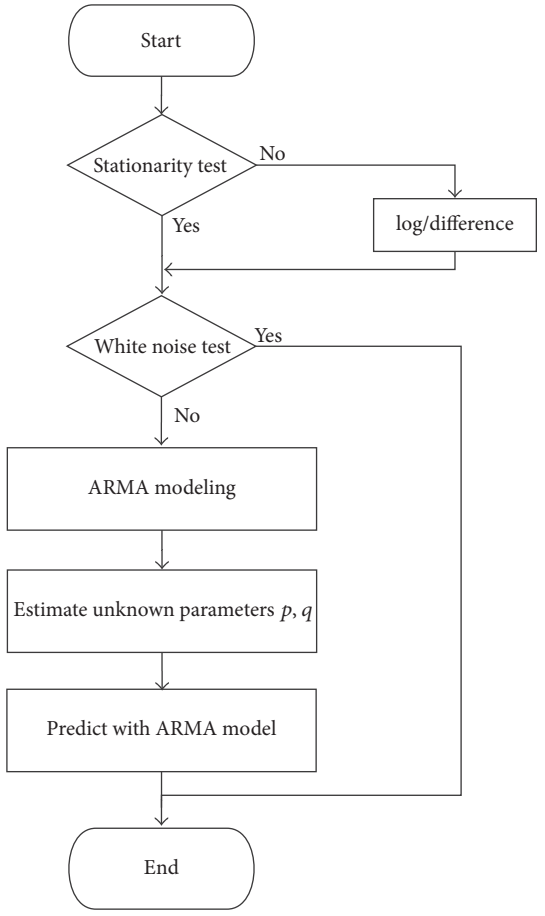


Figure 13: Modeling process of ARMA.

3.3.6. Building XGBoost Model. An integrated process of building XGBoost model is shown as Figure 15. It mainly consists of four steps.

(1) Feature correlation testing: the correlation test is a statistical test on whether the variables are related and the degree

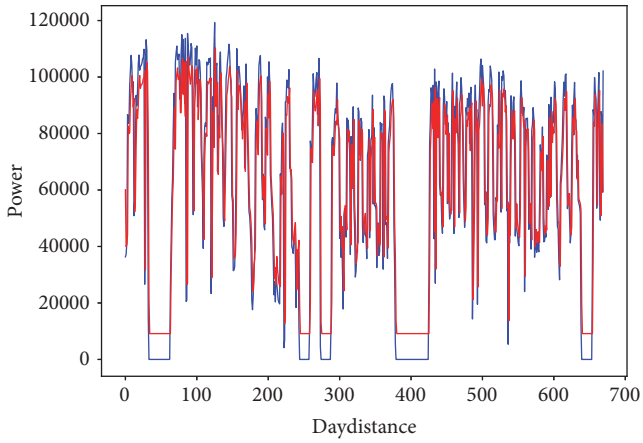


Figure 14: The fitting curve of 1307th enterprise.

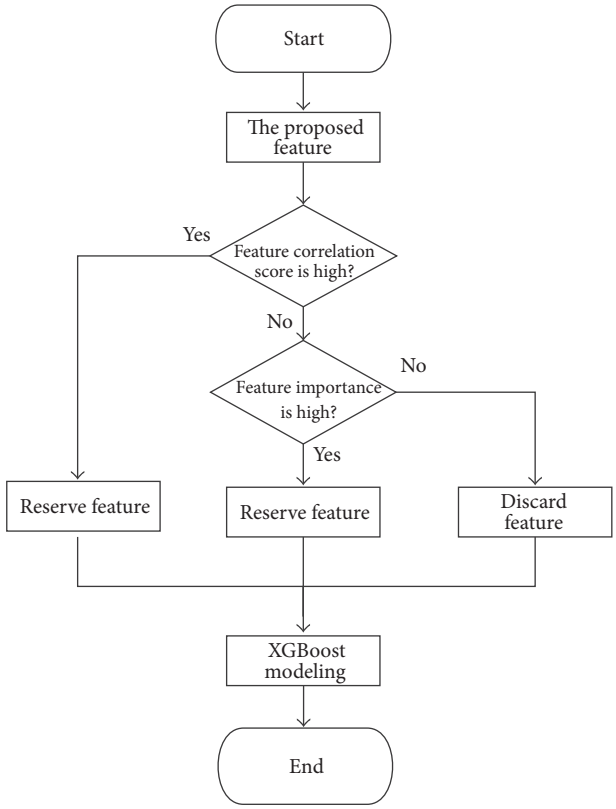


Figure 15: XGBoost algorithm modeling process.

of correlation. We use the Pearson correlation coefficient to measure the correlation between the features. If the correlation between two features is relative high, it indicates that the linear correlation between them exists, and there must be feature redundancy.

(2) Feature importance testing: features are important to the model, but too many features can cause redundancy and overfitting. Therefore, we need to filter features. According to the scores of feature importance, the higher the score is, the more important the features are, and the features with lower scores can be discarded.

(3) Modeling training: after processing the features, we can build the model. Choose XGBoost to train the model and use the 5-fold cross-validation method to verify the model during the training process.

(4) Predict the electricity demand using the built XGBoost model.

Take the data of a small business enterprise as an example to illustrate the process of XGBoost modeling. The data have already been cleaned. For the features listed in Table 5, we should filter features first.

Figure 16 shows the results of Pearson correlation coefficient test of features for the small enterprise. The score of correlation coefficient matrix can be regarded as the similarity between features, which are better if lower. If the correlation of two features is very high, it means that one of them is redundant. From Figure 16, we can see that doy, woy, and moy (f2, f3 and f4); daydis and mondis (f7 and f8); maxt and mint (f9 and f10); doy, woy, and moy (f2, f3, f4); and season (f11) are highly correlated, which means there is feature redundancy. At the same time, combined with the scores of feature importance which is pretraining output of XGBoost model in Figure 9, the features retained in the end are dow, dom, doy, woy, holiday, daydis, maxt, and mint.

Figure 17 shows the fitting curve of the small enterprise after 5-fold cross-validation, in which the blue line represents the actual values and the red represents the fitting curve of the XGBoost model. It can be seen that the fitting curve of the XGBoost model is basically consistent with the actual curve.

4. Experiments

We use the electricity consumption data of 1454 enterprises in Yangzhong High-Tech Industrial Development Zone of Jiangshu Province from 2015-01-01 to 2016-11-30 for experiments. The data between 2015-01-01 and 2016-10-31 are used as training set, and the data between 2016-11-01 and 2016-11-30 are used as test set to verify the model. The experiments mainly include two parts: the parameter optimization, which can be referred to in Section 4.3, and effectiveness verification of the proposed model, which can be referred to in Sections 4.4–4.6. Before the detailed result analysis, we will introduce evaluation indicators and classical models for comparison in Sections 4.1 and 4.2, respectively.

4.1. Evaluation Indicators

(1) MAE. We use MAE for one of the indicators. MAE refers to the mean absolute error between the predicted values and real ones. The formula is shown as follows:

$$\text{MAE} = \frac{\sum_{i=1}^{n} |p_i - q_i|}{n}, \tag{2}$$

where $\{p_1, p_2, \ldots, p_n\}$ refers to the prediction values and $\{q_1, q_2, \ldots, q_n\}$ refers to the real ones. The smaller MAE value is, the more accurate the model is.

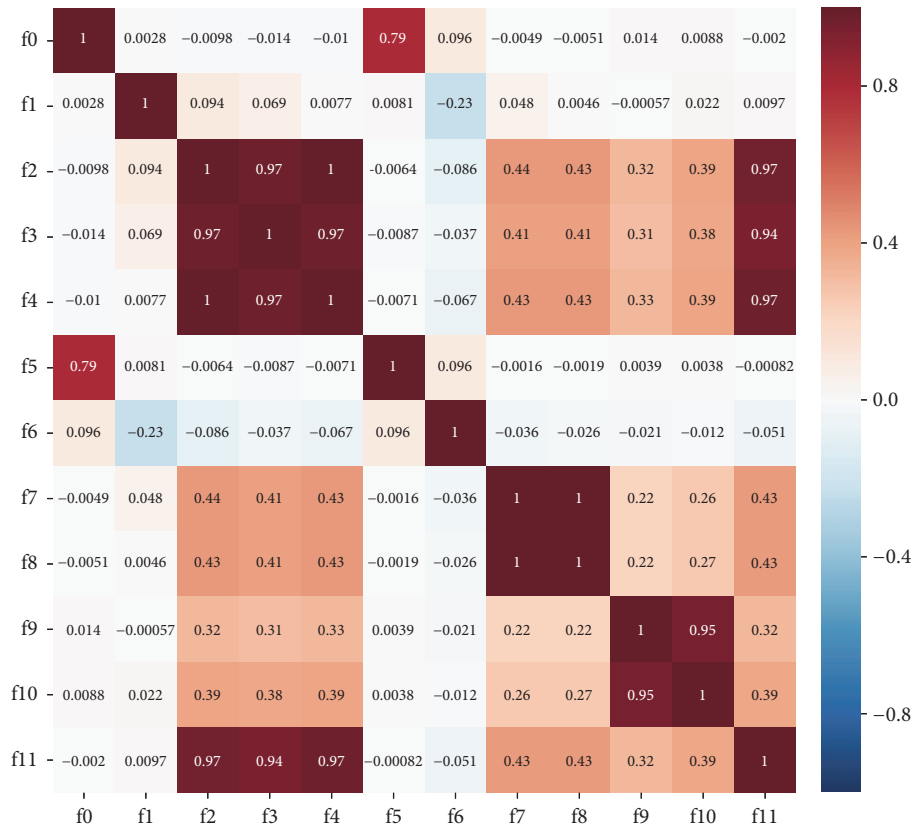


Figure 16: The results of the Pearson correlation coefficient test.

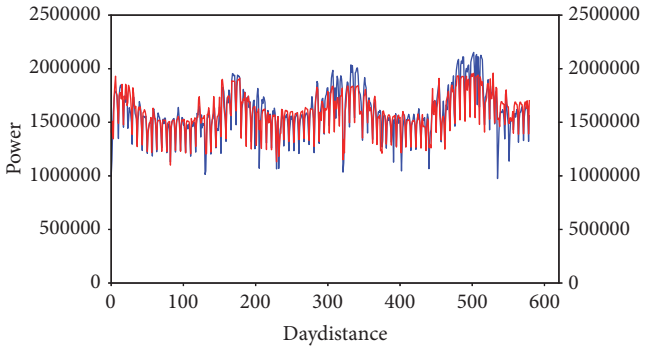


Figure 17: The fitting curve of small enterprises.

(2) Score. In order to measure the average deviation between the prediction values and real one, we use Score as the second indicator, and the detailed computation is shown as the following:

$$\text{Score} = \frac{1}{n}\sum_{k=1}^{n} e^{-3(|pi-qi|/qi)}. \quad (3)$$

Score is a function to calculate relative error. The bigger the Score value is, the more accurate the model is.

4.2. Models for Comparison. We choose the following four classical algorithms for comparison.

(1) ARMA. This algorithm regards the data sequence which is the electricity consumption with time changes as a random sequence and uses a specific mathematics model to describe the sequence.

(2) GBDT Model. Features are first extracted from original data and then selected by Pearson correlation coefficient and feature importance scores. The scores are obtained by pretraining of GBDT. Finally, the prediction value is obtained by training and modeling with GBDT.

(3) Random Forest Model. Features are first extracted from original data and then selected by Pearson correlation coefficient and feature importance scores. The scores are obtained by pretraining of Random Forest. Finally, the prediction value is obtained by training and modeling with Random Forest.

(4) XGBoost Model. Features are first extracted from original data and then selected by Pearson correlation coefficient and feature importance scores. The scores are obtained by pretraining of XGBoost. Finally, the prediction value is obtained by training and modeling with XGBoost.

4.3. Parameter Optimization

Depth of Tree in XGBoost Model. The depth is the primary parameter for XGBoost model, so we first work on optimizing the depth in XGBoost model.

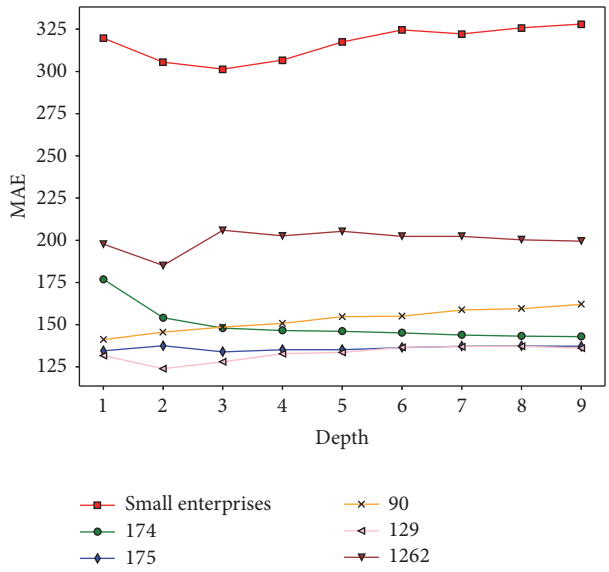


Figure 18: MAE scores with tree depth of XGBoost model changes.

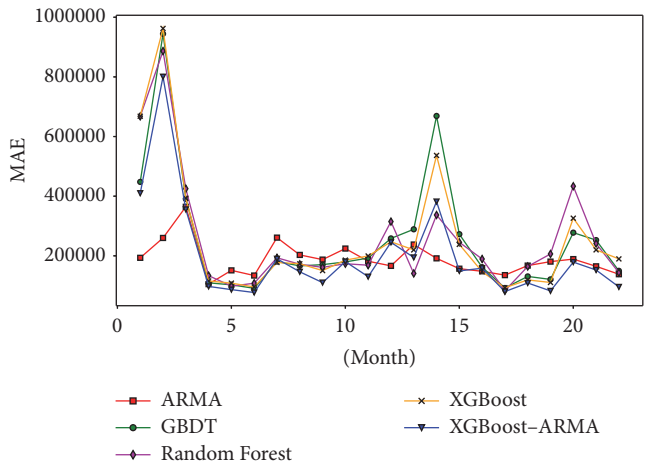


Figure 19: MAE values of five models in each month.

Figure 18 shows the change of MAE based on XGBoost model when depth has different values. The horizontal coordinate refers to the value of depth, the vertical coordinate refers to MAE value, and curves with different colors represent different enterprises. In general, MAE becomes smaller when depth increases. But when depth is large enough, MAE will not change any more. There are problems that will cause overfitting when depth is too large, and an overly fine classification would enlarge calculation. According to Figure 18, for small enterprise, when depth is 3, the MAE is the smallest; that is, the performance is the best. And similarly, the best depths for the enterprises whose ID are 174, 175, 90, 129, and 1262 are 3, 3, 1, 2, and 2, respectively.

(p, q) in ARMA Model. For ARMA model, the most important parameters are p and q. Table 7 is the BIC information for the 1307th enterprise when p, q in ARMA (p, q) has different values.

According to the smallest amount of information principle, the best ARMA parameter is found within all the pairs of p, q. From Table 7, the pair $(1, 0)$ has the smallest amount of information for the 1307th enterprise, so the parameter pair $(1, 0)$ suits better for the 1307th enterprise.

4.4. Verifying the Rationality of User Clustering. In our proposed approach, we propose to cluster users first. Therefore, in this section, we aim to verify the rationality of the step. We compare the MAE and Score values for the two cases, with and without user clustering, under five models, which are ARMA, XGBoost, GBDT, Random Forest, and our proposed XGB-ARMA. The result is shown in Table 8.

From Table 8, by comparing MAE and Score values vertically, the performances of the five models except XGBoost have improved when clustering enterprise users. It proves the rationality of the step. By comparing MAE and Score values horizontally, the performance of XGB-ARMA is the best since it has the smallest MAE and the highest Score.

4.5. Comparison of Different Models. In this section, we aim to make a detailed comparison between our proposed model and 4 classical models. Figure 19 shows how the MAE value changes with the month when we use ARMA, GBDT, Random Forest, XGBoost, or XGB-ARMA separately from January 2015 to October 2016. In Figure 19, the x-axis represents the month; y-axis represents the MAE value. Curves with different colors represent different models.

It can be seen from Figure 19 that the MAE values of XGB-ARMA are the lowest in most of the 22 months, and the MAE values of XGB-ARMA gradually decrease with time, indicating that the model is more and more stationary with the increase of time. On the other hand, it can be seen that the MAE values of various models around February (near the Spring Festival) are relatively high, indicating that the models are disturbed by the Spring Festival. Overall, XGB-ARMA outperforms other models, further demonstrating the effectiveness of the model.

4.6. Results on Test Set. In this section, we use the prediction results on test set to verify the reliability of our proposed model. Figure 20 shows the fitting curve based on XGB-ARMA Model in Nov. 2016.

In Figure 20, x-axis represents each day in Nov. 2016, while y-axis represents the power amount. Red curve represents the fitting result; blue curve represents real values. From Figure 20, we can see that fitting curve is smooth and has good generalization. It has similar tendency with the real values. The blue curve has sudden drops on 26th, 27th, and 28th, since the 1416th enterprise that takes up the 1/4 electricity consumption stopped working on the three days.

According to statistical analysis, the MAE of prediction based on XGB-ARMA in Nov. 2016 is 171641.423967, and the Score is 92.61. It proves that the model has good fitting performance. From the results, we can conclude that different models have different strengths and weaknesses when

Table 7: BIC information for the 1307th enterprise.

p	q					
	0	1	2	3	4	5
0	16004.1	15557.81	15391.2	15289.85	15253.54	15230.46
1	15161.09	15164.31	15170.7	15176.47	15182.02	15188.47
2	15164.25	15170.76	15177.1	15182.34	15187.98	15194.47
3	15170.76	15176.62	15182.36	15188.78	15189.81	15196.37
4	15176.3	15182.33	15188.81	15194.94	NaN	15200.08
5	15182.11	15187.76	NaN	15195.52	15206.96	15209.5

NaN: an infinite number, that is, not a number.

Table 8: MAE and Score values of models with or without user clustering.

Model without user clustering	ARMA	XGBoost	GBDT	Random forest	XGB-ARMA
MAE	188274.29	191378.12	257591.77	277195.20	194278.54
Score	0.8652	0.8678	0.8258	0.8158	0.8616
Model with user clustering	ARMA	XGBoost	GBDT	Random forest	XGB-ARMA
MAE	145605.80	245134.24	176533.17	187297.21	124609.62
Score	0.8954	0.8318	0.8755	0.8674	0.9085

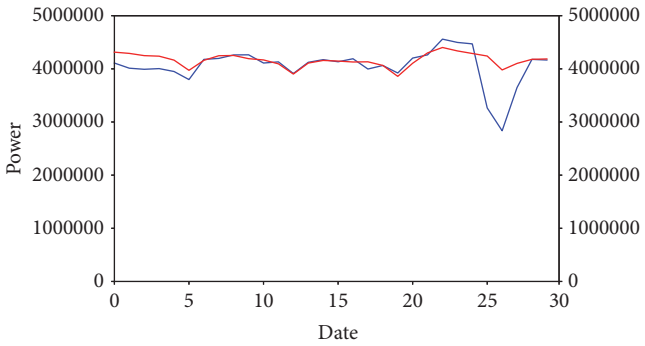


Figure 20: Fitting curve based on XGB-ARMA Model in Nov. 2016.

explaining data from different angles. Some works utilize single model for prediction and therefore abandon better chance, because for some enterprises there may have better models. Different enterprises have different electricity usage patterns. It is better to choose different models based on their own characteristic rather than adopt single model. XGB-ARMA model combines the advantages of ARMA model and XGBoost model, so it can capture the changing rules of electricity consumption for different enterprises more comprehensively blessed with strengths of different models.

5. Conclusion

In this paper, we propose a XGB-ARMA model to predict short-term electricity demand by combining the advantages of XGBoost and ARMA in Fog Computing framework. It can fully utilize the storage and computing ability of Fog Nodes and achieve the mass flow and low latency requirements of smart electricity system by data preprocessing, local computing, and real-time decision. The main contributions of this paper mainly include the following.

(1) We propose clustering enterprise users based on prototype-based K-means algorithm first, and the clustering result shows the density distribution on electricity consumption and clear semantic meaning. It is consistent with the Pareto Principle; that is, 20% of enterprise users consumes 80% of electricity energy.

(2) We propose choosing different models for different users according to the characteristic of their historical electricity consumption. A rigid model selection process is proposed, which includes model prejudgement and model determination. The prejudgement is achieved by analyzing the periodicity/trend and nonlinear/weak stationary of the historical curve, while the model determination is achieved by a series of tests, including correlation test, feature importance scores, stationary, and white noise test.

(3) Before the model building, we propose a processing strategy of abnormal data for different models. In addition, we construct a rich feature set by extending the single column of date time, such as weather, weekend, and holidays.

Future work includes the following: first, we aim to introduce local economic and population flow data to explore the influence of other factors on electricity consumption; second, we would like to explore a new method of enterprise users clustering which can classify users according to data distribution and different premodeling results; third, we would like to employ visualization techniques [50] in the presentation of our solution.

Data Availability

The data used to support the findings of this study are provided by Tianchi under license and so cannot be made freely

available. Access to these data will be considered by the author upon request, with permission of Tianchi.

Conflicts of Interest

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

Acknowledgments

The research is supported by National Natural Science Foundation of China (no. 61772560), Natural Science Foundation of Hunan Province (no. 2016JJ3154), Scientific Research Project for Professors in Central South University, China (no. 904010001), and Innovation Project for Graduate Students in Central South University (no. 1053320181628).

References

[1] 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.

[2] 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.

[3] I. Stojmenovic and S. Wen, "The fog computing paradigm: scenarios and security issues," in *Proceedings of the Federated Conference on Computer Science and Information Systems (FedCSIS '14)*, pp. 1–8, IEEE, Warsaw, Poland, September 2014.

[4] F. Bonomi, R. Milito, P. Natarajan, and J. Zhu, "Fog computing: A platform for internet of things and analytics," *Studies in Computational Intelligence*, vol. 546, pp. 169–186, 2014.

[5] R. Mahmud, R. Kotagiri, and R. Buyya, "Fog Computing: A Taxonomy, Survey and Future Directions," in *Internet of Everything*, Internet of Things, pp. 103–130, Springer Singapore, Singapore, 2018.

[6] L. Kuang, Y. Wang, P. Ma et al., "An Improved Privacy-Preserving Framework for Location-Based Services Based on Double Cloaking Regions with Supplementary Information Constraints," *Security and Communication Networks*, vol. 2017, pp. 1–15, 2017.

[7] L. Kuang, Z. Liao, W. Feng, H. He, and B. Zhang, "Multimedia services quality prediction based on the association mining between context and QoS properties," *Signal Processing*, vol. 120, pp. 767–776, 2016.

[8] Y. J. Wang, X. Z. Ji, and J. M. Shi, "Scenario analysis and application research on big data in smart power distribution and consumption systems," *Proceedings of the CSEE*, vol. 35, no. 8, pp. 1829–1836, 2015.

[9] A. E. Clements, A. S. Hurn, and Z. Li, "Forecasting day-ahead electricity load using a multiple equation time series approach," *European Journal of Operational Research*, vol. 251, no. 6, pp. 522–530, 2016.

[10] M. Bessec, J. Fouquau, and S. Meritet, "Forecasting electricity spot prices using time-series models with a double temporal segmentation," *Applied Economics*, vol. 48, no. 5, pp. 361–378, 2016.

[11] F. Yasmeen and M. Sharif, "Functional Time series (FTS) Forecasting of Electricity Consumption in Pakistan," *International Journal of Computer Applications*, vol. 124, no. 7, pp. 15–19, 2015.

[12] V. Bianco, O. Manca, and S. Nardini, "Electricity consumption forecasting in Italy using linear regression models," *Energy*, vol. 34, no. 9, pp. 1413–1421, 2009.

[13] F. Zhang, H. Wang, T. Han et al., "Short-term load forecasting based on partial least-squares regression," *Power System Technology*, vol. 3, pp. 36–40, 2003.

[14] W.-C. Hong, Y. Dong, W. Y. Zhang, L.-Y. Chen, and B. K. Panigrahi, "Cyclic electric load forecasting by seasonal SVR with chaotic genetic algorithm," *International Journal of Electrical Power & Energy Systems*, vol. 44, no. 1, pp. 604–614, 2013.

[15] K. Kavaklioglu, "Modeling and prediction of Turkey's electricity consumption using support vector regression," *Applied Energy*, vol. 88, no. 1, pp. 368–375, 2011.

[16] C.-C. Hsu, C.-H. Wu, S.-C. Chen, and K.-L. Peng, "Dynamically optimizing parameters in support vector regression: An application of electricity load forecasting," in *Proceedings of the 39th Annual Hawaii International Conference on System Sciences, HICSS'06*, p. 30, USA, January 2006.

[17] H. S. Hippert, E. C. Pedreira, and C. R. Souza, "Neural networks for short-term load forecasting: a review and evaluation," *IEEE Transactions on Power Systems*, vol. 16, no. 1, pp. 44–55, 2001.

[18] J. P. S. Catalão, H. M. I. Pousinho, and V. M. F. Mendes, "Hybrid wavelet-PSO-ANFIS approach for short-term electricity prices forecasting," *IEEE Transactions on Power Systems*, vol. 26, no. 1, pp. 137–144, 2011.

[19] Y. T. Chae, R. Horesh, Y. Hwang, and Y. M. Lee, "Artificial neural network model for forecasting sub-hourly electricity usage in commercial buildings," *Energy and Buildings*, vol. 111, pp. 184–194, 2016.

[20] A. G. Bakirtzis, J. B. Theocharis, S. J. Kiartzis, and K. J. Satsios, "Short term load forecasting using fuzzy neural networks," *IEEE Transactions on Power Systems*, vol. 10, no. 3, pp. 1518–1524, 1995.

[21] B. W. Tao, Z. L. Zhang, H. Pan et al., "Spatial electric load forecasting based on double-level Bayesian classification," in *Proceedings of the Chinese Society of Electrical Engineering*, vol. 27, pp. 13–17, 2007.

[22] T. Niimura and T. Nakashima, "Deregulated electricity market data representation by fuzzy regression models," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 31, no. 3, pp. 320–326, 2001.

[23] A. Deihimi, O. Orang, and H. Showkati, "Short-term electric load and temperature forecasting using wavelet echo state networks with neural reconstruction," *Energy*, vol. 57, pp. 382–401, 2013.

[24] T.-Y. Zhu, Y.-Q. Li, Y. Zhang, X.-Z. Zhang, and C.-Y. He, "New algorithm of advancing weather adaptability based on ARIMA model for day-ahead power load forecasting," *Proceedings of the Chinese Society of Electrical Engineering*, vol. 26, no. 23, pp. 14–19, 2006.

[25] A. Bogomolov, B. Lepri, R. Larcher, F. Antonelli, F. Pianesi, and A. Pentland, "Energy consumption prediction using people dynamics derived from cellular network data," *EPJ Data Science*, vol. 5, no. 1, article no. 13, 2016.

[26] Y. Han, X. Sha, E. Grover-Silva, and P. Michiardi, "On the impact of socio-economic factors on power load forecasting," in *Proceedings of the 2nd IEEE International Conference on Big Data, IEEE Big Data 2014*, pp. 742–747, USA, October 2014.

[27] L. Ghelardoni, A. Ghio, and D. Anguita, "Energy load forecasting using empirical mode decomposition and support vector regression," *IEEE Transactions on Smart Grid*, vol. 4, no. 1, pp. 549–556, 2013.

[28] J. Che, J. Wang, and G. Wang, "An adaptive fuzzy combination model based on self-organizing map and support vector regression for electric load forecasting," *Energy*, vol. 37, no. 1, pp. 657–664, 2012.

[29] P. Regulski, D. S. Vilchis-Rodriguez, S. Djurović, and V. Terzija, "Estimation of Composite Load Model Parameters Using an Improved Particle Swarm Optimization Method," *IEEE Transactions on Power Delivery*, vol. 30, no. 2, pp. 553–560, 2015.

[30] I. F. Visconti, D. A. Lima, J. M. C. De Sousa Costa, and N. R. D. B. C. Sobrinho, "Measurement-based load modeling using transfer functions for dynamic simulations," *IEEE Transactions on Power Systems*, vol. 29, no. 1, pp. 111–120, 2014.

[31] A. U. Haque, P. Mandal, J. Meng, A. K. Srivastava, T.-L. Tseng, and T. Senjyu, "A novel hybrid approach based on wavelet transform and fuzzy artmap networks for predicting wind farm power production," *IEEE Transactions on Industry Applications*, vol. 49, no. 5, pp. 2253–2261, 2013.

[32] N. Sovann, P. Nallagownden, and Z. Baharudin, "Electricity load forecasting using hybrid wavelet neural network based on parallel prediction method," in *Proceedings of the 6th International Conference on Intelligent and Advanced Systems, ICIAS 2016*, Malaysia, August 2016.

[33] G.-X. Li, "Electricity consumption forecast based on wavelet neural network," in *Proceedings of the 2016 International Conference on Information System and Artificial Intelligence, ISAI 2016*, pp. 361–364, China, June 2016.

[34] T. Nengling, J. Stenzel, and W. Hongxiao, "Techniques of applying wavelet transform into combined model for short-term load forecasting," *Electric Power Systems Research*, vol. 76, no. 6-7, pp. 525–533, 2006.

[35] N. M. Pindoriya, S. N. Singh, and S. K. Singh, "An adaptive wavelet neural network-based energy price forecasting in electricity markets," *IEEE Transactions on Power Systems*, vol. 23, no. 3, pp. 1423–1432, 2008.

[36] P. K. Pany and S. P. Ghoshal, "Dynamic electricity price forecasting using local linear wavelet neural network," *Neural Computing and Applications*, vol. 26, no. 8, pp. 2039–2047, 2015.

[37] J. Che and J. Wang, "Short-term electricity prices forecasting based on support vector regression and Auto-regressive integrated moving average modeling," *Energy Conversion and Management*, vol. 51, no. 10, pp. 1911–1917, 2010.

[38] X. Cheng, C. Kang, Q. Xia, and Y. Shen, "Integrated model of short term load forecasting," *Automation of Electric Power Systems*, vol. 24, no. 9, pp. 42–44, 2000.

[39] G. Chen, K. Li, T. Chung, H. Sun, and G. Tang, "Application of an innovative combined forecasting method in power system load forecasting," *Electric Power Systems Research*, vol. 59, no. 2, pp. 131–137, 2001.

[40] C. Kang, X. Cheng, Q. Xia, Y. Huang, and F. Gao, "Novel approach considering load-relative factors in short-term load forecasting," *Electric Power Systems Research*, vol. 70, no. 2, pp. 99–107, 2004.

[41] J. Wang, S. Zhu, W. Zhang, and H. Lu, "Combined modeling for electric load forecasting with adaptive particle swarm optimization," *Energy*, vol. 35, no. 4, pp. 1671–1678, 2010.

[42] G. Elliott and A. Timmermann, "Optimal forecast combinations under general loss functions and forecast error distributions," *Journal of Econometrics*, vol. 122, no. 1, pp. 47–79, 2004.

[43] Y. Yao, Z. Lian, S. Liu, and Z. Hou, "Hourly cooling load prediction by a combined forecasting model based on analytic hierarchy process," *International Journal of Thermal Sciences*, vol. 43, no. 11, pp. 1107–1118, 2004.

[44] R. K. Rietz and S. Suryanarayanan, "A review of the application of analytic hierarchy process to the planning and operation of electric power microgrids," in *Proceedings of the 40th North American Power Symposium, NAPS2008*, Canada, September 2008.

[45] V. Petridis, A. Kehagias, L. Petrou et al., "A Bayesian multiple models combination method for time series prediction," *Journal of Intelligent & Robotic Systems*, vol. 31, no. 1–3, pp. 69–89, 2001.

[46] M. Peng, S. Yan, K. Zhang, and C. Wang, "Fog-computing-based radio access networks: Issues and challenges," *IEEE Network*, vol. 30, no. 4, pp. 46–53, 2016.

[47] A. V. Dastjerdi and R. Buyya, "Fog Computing: Helping the Internet of Things Realize Its Potential," *The Computer Journal*, vol. 49, no. 8, Article ID 7543455, pp. 112–116, 2016.

[48] G. Barta, G. B. G. Nagy, S. Kazi, and T. Henk, "GEFCOM 2014—probabilistic electricity price forecasting," *Smart Innovation, Systems and Technologies*, vol. 39, pp. 67–76, 2015.

[49] K. B. Sahay, N. Kumar, and M. M. Tripathi, "Short-term load forecasting of Ontario Electricity Market by considering the effect of temperature," in *Proceedings of the 6th IEEE Power India International Conference, PIICON 2014*, ind, December 2014.

[50] Z. Liao, L. Kong, X. Wang et al., "A visual analytics approach for detecting and understanding anomalous resident behaviors in smart healthcare," *Applied Sciences (Switzerland)*, vol. 7, no. 3, article no. 254, 2017.

Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 7823979, 11 pages
https://doi.org/10.1155/2018/7823979

Research Article

NTRU Implementation of Efficient Privacy-Preserving Location-Based Querying in VANET

Bo Mi,[1] Darong Huang,[1] and Shaohua Wan [2]

[1] *Institute of Information Science and Engineering, Chongqing Jiaotong University, Chongqing 400074, China*
[2] *School of Information and Safety Engineering, Zhongnan University of Economics and Law, Wuhan 430073, China*

Correspondence should be addressed to Shaohua Wan; shaohua.wan@ieee.org

Received 26 January 2018; Accepted 21 March 2018; Published 3 May 2018

Academic Editor: Xuyun Zhang

The key for location-based service popularization in vehicular environment is security and efficiency. However, due to the constrained resources in vehicle-mounted system and the distributed structure of fog computation, disposing of the conflicts between real-time implementation and user's privacy remains an open problem. Aiming at synchronously preserving the position information for users as well as the data proprietorship of service provider, an efficient location-based querying scheme is proposed in this paper. We argue that a recent scheme proposed by Jannati and Bahrak is time-consuming and vulnerable against active adaptive corruptions. Thus accordingly, a postquantum secure oblivious transfer protocol is devised based on efficient NTRU cryptosystem, which then serves as the understructure of a complete location-based querying scheme in ad hoc manner. The security of our scheme is proved under universal composability frame, while performance analysis is also carried out to testify its efficiency.

1. Introduction

With the development and fusion of techniques such as sensing, controlling, communication, positioning, and fog computation, vehicular ad hoc network (VANET), which is identified as a specific application of Internet of Things (IoT), has become a promising understructure to enhance traffic safety and convenience. As an important element of the intelligent transportation system (ITS), VANET is typically composed of numerous on-board units (OBU) equipped on vehicles and road-side units (RSUs) serve as infrastructure [1]. Different from traditional networking, vehicular ad hoc network emphasizes heavily on adaptive computation as well as communication of end-users and edge devices, which is characterised as localized data storage, dense geographical distribution, boundary service providing, and compound data aggregation or analysis. A wide range of applications can be supported taking advantage of such fundamental installation; for example, when driving on the road, one can fall back on the VANET to locate services (shops, gas stations, etc.) on his route, or even be notified of any forecasted traffic condition along her itinerary. Though it is envisioned that the future transportation would be "information-driven" and "wirelessly enabled," the problems of confidential and privacy-preserving communication remain insufficiently solved due to the broadcasting nature of VANET [2]. Moreover, since one of the most attractive applications of VANET is location-based querying, it is always self-extended to traditional networks such as Internet. As illustrated in Figure 1, any authorized on-board unit may access the querying service providers (QSPs) in backbone or local RSUs to inquire about interested information via various communication channels, which makes the security issue more complicated in such foggy environment.

As for privacy, the user may not want anybody, including the infrastructure units or service providers, to be aware of any information about her query. That means it should either be impossible to link up a query with the real identity of the user or make the query itself indistinct to some extent. In order to accomplish privacy requirement, two research lines are followed in literatures.

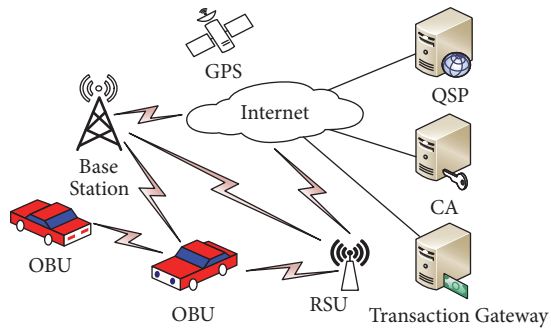


Figure 1: VANET extension.

1.1. Correlation Concealment. Due to the interactive nature of location-based querying, one can easily associate the identity of a user with a specific location which may severely violate the privacy of personal health condition, social relationships, habits, and so on. Accordingly, once the connectivity of inquired location and user ID are obscured, the sensitive information may be preserved to some extent. This kind of privacy-protection method includes the following.

1.1.1. Anonymity and Pseudonym. The goal of cryptonym methods is to prevent an adversary from reidentifying the data source by exploiting any exposed information. It generally relies on the fact that most location-based services are not strictly dependent on the knowledge of user's identity. Thereby, the most challenging issues turn into pseudonymous authentication, integrity, and nonrepudiation. Specifically, a large number of certificates are usually preloaded for each vehicle, which will be abandoned after usage in a short period of time. Coupled with reputation mechanisms, those certificates can thus be used to appraise credibility of anonymous sources or to fulfil the backtracking purpose [3]. Nevertheless, anonymity and pseudonym schemes are only robust to semihonest secure model, because malicious vehicles may not discard or update their certificates as required by the protocols [4].

1.1.2. Mix Zones. The technique of mix zones is originally introduced by Beresford and Stajano [5], where the pseudonym should be exchange amongst all users within a same zone. The time interval when a vehicle passes through a mix zone is called the silent period, which means it must dumb its position so as to break off the connection between its identities at the entry and exit points. Palanisamy and Liu [6] investigated various context information in traffic environment that may reveal detailed trajectories such as geometrical or temporal constraints and devised the MobiMix approach directing against such privacy infringement. It is worth mentioning that the idea of k-anonymity presented by Gruteser and Grunwald [7] is always served as a combination of anonymity (pseudonym) and mix zone implementation. For example, Caballero-Gil et al. [8] exploited the spatial and temporal cloaking to calculate the k-anonymity set, which makes a vehicle indistinguishable from other $k-1$ counterparts. To avoid active corruption, if a number of complaints are received pertinent to a malicious node, a track algorithm can also be carried out to prevent further detriment. Aiming at Sybil attack, Feng et al. [9] bounded k-anonymity and reputation schemes together, which can effectively suppress the spread of false messages when updating the anonymity.

1.2. Query Fuzzification. The service of location-based querying can be deemed as a process of information retrieval, which means only the users care about the correctness or precision of the research outcomes. Therefore, lots of approaches are presented taking the advantage of information asymmetry between users and the server, including the following.

1.2.1. Position Dummies. It is aiming to deceive the QSP by confounding the user's true position together with multiple false locations [10–13]. Nevertheless, since the traffic networks are always scattered but structured, it is difficult to create dummies indistinguishable from the true position. In order to generate plausible dummies, Shankar et al. [14] proposed the SybilQuery approach, which obfuscates the real position with dummies chosen from a historical traffic database.

1.2.2. Obfuscation. The idea behind position obfuscation is to intentionally reduce the precision of inquiry messages. Typically, the protocol proposed by Ardagna et al. [15] used a circular area to substitute the exact position of a user. Though the obfuscation area can be allodially determined by the querier, the trade-off between privacy and precision is of great significance. Thus, accordingly, Reynold et al. [16] introduced a model for probabilistic range queries depending on the overlapping size of the query area and the obfuscation shapes. Another way to obfuscate the user's position is the coordinate transformation, where some geometric mappings are carried out on a series of users' coordinates before sending to the server. However, in order to ensure the functionality of the QSP, it is impossible to find an all-sided protection scheme purely based on coordinate transformation because the service provider has to be able to determine the relative position of objects and areas to each other [17]. In addition, to preserve the trajectory of user, a great deal of spatiotemporal location obfuscation schemes are also proposed, which also took the temporal information associated with positions into account [18–25].

Table 1: Security parameters for diffident public key cryptosystems.

Mathematical problem	Integer factorization	Elliptic curve discrete logarithm	Discrete logarithm
Security level	RSA (bits)	ECC (bits)	ElGamal (bits)
Low security	1024	160	1024
Moderate security	2048	224	2048
Standard security	3072	256	3072
High security	7680	384	7680
Highest security	15360	512	15360

1.2.3. Position Sharing. With the existence of several untrusted servers, location information can be mathematically calculated as a series of shares and distributed on different databases. This approach was first proposed by Dürr et al. [26] who split up the location information into shares of strictly limited precision. After retrieving adequate shares from multiple servers, the user can execute a combination algorithm which fuses them into a message of higher precision. In order to prevent attackers from deriving the precise position via coordinate relationships of a map, Skvortsov et al. took map knowledge into account [27] and further improved their protocol by optimizing the placement of shares in terms of servers' trustworthiness [28]. Though position sharing schemes can also be implemented on account of obfuscation or coordinate transformation [29], cryptography-based fashions are preferable due security consideration [30].

1.2.4. Cryptographic Approaches. Due to the capacities such as confidentiality, integrity, and authenticity, cryptographic primitives are taken as desirable building blocks to realize position privacy. For the sake of concealing the real identity of a user, ring or group signature are generally used to confound the querier as a member of a vehicular set [31, 32]. By using private information retrieval (PIR) technique, a QSP can answer queries without learning or revealing any information of the query [33, 34]. Meanwhile, since the computational result of ciphertext matches that of the plaintext, homomorphic cryptosystems are also valued as promising tools for location privacy application [35, 36].

Nevertheless, the aforementioned approaches took only the querier's position information as a target for protection and simply lost the sight of QSP's data ownership. Practically, the charging models in nowadays always lie on a per-query basis which enable drivers to use the service in ad hoc manner and pay for their queries according to the quantity. With regard to the proprietorship of QSP's records, Paulet et al. [37] proposed a location-based querying approach in the light of 1-out-n oblivious transfer (OT_n^1). Their scheme made use of an ElGamal cryptosystem which imposed an additional privacy property for the sender such that the receiver could learn at most one of the retrieved items. However, Jannati and Bahrak [38] caught the sight of its security defect arguing that the receiver is able to decrypt all ciphered records; thus the QSP's data ownership cannot be preserved. In order to rectify the vulnerability of Paulet's scheme, they also reconstructed the oblivious transfer part of it at the cost of higher computational overhead.

Table 2: Security parameters for NTRUEncrypt.

Mathematical problem	Shortest vector problem		
Security level	N	q	p
Moderate security	401	2048	3
Standard security	439	2048	3
High security	593	2048	3
Highest security	743	2048	3

It is well known that ElGamal encryption is defined over a cyclic group G, whose security depends on the difficulty of computing discrete logarithms. Therefore, a large security parameter must be considered in order to make sure that it is unbreakable. Though other traditional public key cryptosystems may also be exploited as basic primitives to realize oblivious transfer, they are deficient in efficiency due to computational hardness assumptions depending on large parameters. For the same reason, these cryptosystems tend to be vulnerable with the advent of quantum machine era. The comparison of security parameters amongst diffident cryptosystems is given in Table 1.

During encryption phase, ElGamal requires two exponentiation operations, while one exponentiation should be correspondingly carried out for decryption. Since exponentiation on large numbers is always time-consuming and occupies a lot of memory, we argue that Jannati's scheme is not efficient enough, especially under embedded environments. Moreover, though their scheme is proved to be secure under game-based verification, active and adaptive corruptions are simply ignored because CCA (chosen ciphertext attack) security is unachievable by ElGamal cryptosystem itself [39].

In order to eliminate the defects of Jannati's protocol, we take the advantage of NTRUEncrypt to implement privacy-preserving location-based querying. As a relatively new public key cryptosystem developed in 1996, the number theory research unit encryption (NTRU) [40] runs faster compared to other asymmetric encryption schemes and is more competitive to be realized in resource-constraint environments such as mobile devices or smart cards. Up till 2017, literatures can be found that introduce new parameters to resist currently known attacks and increase its computation power [41, 42]. According to the latest research [43], the parameters in Table 2 are considered secure.

As for Table 2, the parameters, where N defines a truncated polynomial ring $R = Z[x]/(x^N - 1)$ used in NTRUEncrypt and q, p are two moduli, are relatively smaller

than that of traditional public key cryptosystems. Moreover, it uses only simple polynomial multiplications; the time of performing an NTRU operation increases only quadratically. Taking moderate security for example, if both exponentiation and polynomial multiplication are composed of $\log_2 q$-bits modular multiplications, the former must invoke the basis $q^2/\log_2 q$ times compared to $N \log_2 N$ of the latter. It is reported that, using a modern GTX280 GPU, a throughput of up to 200,000 encryptions per second can be reached at a security level of 256 bits [44], which is only approximately 20 times slower than a recent AES implementation [45]. Accordingly, we resort to the characteristics of high efficiency as well as postquantum security and employ NTRUEncrypt as a building block to realize oblivious transfer. Then, based on the novel OT_n^1 protocol, an adaptive secure location-based querying scheme can thus be achieved.

The rest of this paper is organized as follows. We first give some preliminaries about oblivious transfer and NTRUEncrypt in Section 2. In Section 3, a NTRU-based 1-out-n oblivious protocol will be devised in advance which is then used to structure the secure location-based querying scheme after describing the system model. Security analyses and performance evaluations are given in Sections 4 and 5. The paper is finally concluded in Section 6.

2. Preliminaries on 1-Out-n Oblivious Transfer and NTRUEncrypt

Oblivious transfer, originally introduced as conjugate coding, owns its name to Rabin [46]. Amongst different flavors of OT, 1-out-n oblivious transfer has been extensively studied in the literature since any cryptographic task can be achieved by this extremely basic primitive [47]. In cryptography, a 1-out-n oblivious transfer is a type of protocol in which a receiver R is entitled to obtain 1 out of n messages held by a sender S without learning any other messages, while the sender do not know which massage has been chosen. The protocol is formally described as in Table 3.

In order to optimize the performance of oblivious transfer protocol, several tricks can be imposed on it. For example, [48] enables the computation of many OTs with a small elementary cost from k OT at a normal cost and also enables to reduce oblivious transfers of long strings to oblivious transfers of short strings using a pseudorandom generator.

In this paper, an efficient and secure OT_n^1 protocol will be constructed based on NTRUEncrypt in the light of its linearity and resistance to quantum machines. The NTRU encryption algorithm works on a truncated polynomial ring $R = Z[x]/(x^N - 1)$ with convolution multiplication and all polynomials in the ring have integer coefficients and degree at most $N - 1$:

$$\mathbf{a}(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_{N-1} x^{N-1}. \tag{1}$$

Similar to the prime decomposition problem exploited by RSA, the security of NTRUEncrypt relies on hardness of factoring a reducible polynomial, which is equivalent to the shortest vector problem. Thus, it is infeasible to usurp the secret key if the parameters are chosen secure enough.

Table 3: Oblivious transfer paradigm $\mathcal{F}_{\mathrm{OT}}$.

Input
(i) S input $m_0, m_1, \ldots, m_{n-1} \in M$
(ii) R input $\delta \in \{0, 1, \ldots, n-1\}$
Output
(i) S output $\perp$ (i.e., nothing)
(ii) R output m_δ

For each system, three integer parameters (N, p, q) are specified, where p and q are two moduli who truncate the ring R as $R_p = (Z/pZ)[x]/(x^N - 1)$ and $R_q = (Z/qZ)[x]/(x^N - 1)$. It is always assumed that N, p are prime while q is coprime to both q and N. To generate a key pair, two key polynomials f and g whose coefficients lie within $\{-1, 0, 1\}$ must be generated in advance. An additional requirement that there exist two inverses f_p, f_q, where $f \cdot f_p = 1 \bmod p$ and $f \cdot f_q = 1 \bmod q$, must also be satisfied. Then, f together with f_p can be preserved as the secret key, while $h = p \cdot f_q \cdot g$ will be published to be the public key.

During encrypting phase, a message m should be represented as a binary or ternary string and transformed into a truncated polynomial within the ring R. Then a binding polynomial r with small coefficients should be randomly chosen to calculate the ciphertext as

$$c = r \cdot h + m \pmod q. \tag{2}$$

In order to decrypt the cryptograph c, the receiver first computes

$$a = f \cdot c \pmod q \tag{3}$$

and then lifts its coefficients to interval $[-q/2, q/2]$ and achieves the plaintext as

$$m = f_p \cdot a \pmod p. \tag{4}$$

In order to prove the correctness of our protocols, a polynomial set $\mathcal{T}(d_1, d_2)$ specified by two parameters is defined in advance.

Definition 1. For any positive integers d_1 and d_2,

$$\mathcal{T}(d_1, d_2) = \left\{ \mathbf{a}(x) \in R_q : \begin{array}{l} d_1 \text{ cofficients of } \mathbf{a}(x) \text{ are } 1 \\ d_2 \text{ cofficients of } \mathbf{a}(x) \text{ are } -1 \\ \text{other cofficients of } \mathbf{a}(x) \text{ are } 0. \end{array} \right. \tag{5}$$

According to Definition 1, the correctness of NTRU decryption can be guaranteed in terms of the condition described as below.

Lemma 2. *If the polynomials of NTRU cryptosystem are chosen from*

$$\begin{aligned} f &\in \mathcal{T}(d+1, d), \\ g &\in \mathcal{T}(d, d), \\ r &\in \mathcal{T}(d, d), \end{aligned} \tag{6}$$

whose coefficients satisfy

$$q > p(6d + 1), \tag{7}$$

then a legal receiver can accurately recover ciphertext c with her private key.

Proof. Since all polynomials of

$$a = p \cdot g \cdot r + f \cdot m \pmod q \tag{8}$$

are provided with coefficients designated by formula (6), the parameters of $g \cdot r$ after convolution polynomial multiplication will never overrange $[-2d, 2d]$. Similarly, the parameters of f as well as m are located within $[-p/2, p/2]$ which means that the maximal parameter of $f \cdot m$ is $p(d + 1/2)$ to its very extent. As a result, once the condition of $q > p(6d + 1)$ is met, all parameters of (8) can be lifted to $[-q/2, q/2]$ without losing any information. Then by computing

$$f_p \cdot a = f_p \cdot p \cdot g \cdot r + m = m \pmod p, \tag{9}$$

the message can accurately be recovered. □

3. Location Privacy-Preserving Querying Based on NTRU

In this paper, a novel location-based querying scheme is proposed aiming at not only protecting the position privacy of drivers but also preserving the data proprietary of QSP. Specifically, three goals must be achieved in terms of security and feasibility.

(a) Within authenticated but not confidential communication environments, any malicious third party is incapable of gaining or efficaciously modifying any information of the conversation.

(b) Even if active and adaptive corrupted participant exists, the driver must be insensible of any data hold by QSP except the one she requested while keeping her querying information concealed.

(c) The protocol should be feasible on both vehicle-mounted devices as well as location-based servers, which means that low computation and communication burden must be fulfilled.

For clarity, a novel 1-out-n oblivious transfer protocol will be presented in the first place. Then we will employ it as the building block to complete our entire scheme.

3.1. NTRU Implementation of 1-Out-n Oblivious Transfer. Different from traditional public key cryptosystems, NTRU is structured on a truncated polynomial ring which is provided with both addition and multiplication. Since the time of performing convolution multiplication is much faster than that of modular exponentiation on large numbers, the preferable efficiency and security property of NTRU are more appropriate to construct the basic oblivious transfer protocol.

In order to realize the NTRU-based 1-out-n oblivious transfer, the messages held by the sender are presented as $m_0, m_1, \ldots, m_{n-1}$, which must be kept unacquainted from the receiver except for m_δ. Accordingly, we describe the primitive 1-out-n oblivious transfer protocol as below.

During key generation phase, the sender constructs a key pair as in Section 2, she releases her public key $pk = h$ to all potential receivers or stores it in a communal database, while keeping the secret key $sk = (f, f_p)$ private.

In oblivious transfer phase, the sender is supposed to choose n random polynomials $\gamma_0, \gamma_1, \ldots, \gamma_{n-1}$ from $\mathcal{T}(d, d)$, where $r_i \cdot h$ can be represented as $p \cdot f_q \cdot \gamma_i$, and encrypt all plaintexts to be

$$c_i = r_i \cdot h + m_i \pmod q, \quad (i = \{0, 1, \ldots, n-1\}), \tag{10}$$

which is then sent to the receiver.

When all ciphertexts are received, the receiver first generates a random polynomial s belonging to $\mathcal{T}(d, d)$ and figures out its inverse $s^{-1} \pmod p$. If the inverse of polynomial s does not exist, she can simply resample another one and repeat the inversion process.

After that, the receiver must single out the δth ciphertext and compute it as

$$c'_\delta = r' \cdot h + s \cdot c_\delta \pmod q, \tag{11}$$

utilizing another random polynomial r' chosen from $\mathcal{T}(d, d)$. The result will be sent back to the sender.

Depending on the altered ciphertext c'_δ, the sender can calculate

$$\begin{aligned} a_\delta &= f \cdot c'_\delta \pmod q, \\ b_\delta &= a_\delta \pmod p \end{aligned} \tag{12}$$

and then

$$c''_\delta = f_p \cdot b_\delta \pmod p \tag{13}$$

to be her response for the driver.

Since the driver is aware of polynomial s, she can achieve the expected messages m_δ by multiply c''_δ with s^{-1} modulo p.

The above process is also characterized in Table 4.

Correctness of the 1-out-n oblivious transfer protocol relies on the computation of polynomials in truncated polynomial ring, as follows.

Lemma 3. *The driver can correctly obtain message m_δ if $q > 2p(2d^2 + 5d)$.*

Proof. Since the parameters of $g \cdot r$ or $\gamma_\delta \cdot s$ are seated within $[-2d, 2d]$ and the coefficients of $f \cdot s \cdot m_\delta$ cannot exceed $p(2d^2 + d)$ for polynomial $a_\delta = p \cdot g \cdot r' + p \cdot \gamma_\delta \cdot s + f \cdot s \cdot m_\delta \pmod q$. No information will be lost when lifting the coefficients of a_δ to $[-q/2, q/2]$, if $q > 2p(2d^2 + 5d)$ the same as Lemma 2. By computing $m_\delta = c''_\delta \cdot s^{-1} \pmod p$, where $c''_\delta = s \cdot m_\delta \pmod p$, the driver can achieve the exact message m_δ she expected. □

3.2. Efficient and Secure Location-Based Querying. The system is modelled as a QSP and a series of vehicles. More specifically, the QSP can be considered working in a distributed

Table 4: OT_n^1 protocol based on NTRUEncrypt.

Server	*Driver*
Holds n messages $m_i \in \{0,1\}^N$, $i = \{0,1,\ldots,n-1\}$.	Holds δ, $\delta \in \{0,1,\ldots,n-1\}$.
Key generation phase	
$(q, p, f, f_q, f_p, g) \leftarrow KeyGen(\kappa)$, $sk : (f, f_p)$, $pk : h$, Sends pk to the driver.	
Oblivious transfer phase	
For all $i = \{0,1,\ldots,n-1\}$, $\gamma_i \leftarrow_R \mathcal{T}(d,d)$, $c_i = p \cdot f_q \cdot \gamma_i + m_i (\mathrm{mod}\, q)$, Sends all c_i to the driver.	
	$r' \leftarrow_R \mathcal{T}(d,d)$, $s \leftarrow_R \mathcal{T}(d,d)$, where s is reversible modulo p, $c'_\delta = r' \cdot h + s \cdot c_\delta (\mathrm{mod}\, q)$, Sends c'_δ back to the server.
$a_\delta = f \cdot c'_\delta (\mathrm{mod}\, q)$, $b_\delta = a_\delta (\mathrm{mod}\, p)$, $c''_\delta = f_p \cdot b_\delta (\mathrm{mod}\, p)$, Sends c''_δ to the driver.	
	$m_\delta = c''_\delta \cdot s^{-1} (\mathrm{mod}\, p)$.

manner, which is composed of a centralized authentication server together with numerous delivery RSUs. The reason behind such configuration is to separate data retrieval from transaction process, which not only preserves the driver's position privacy but also abates the operating load of service centre. Resorting to the OBUs equipped on vehicles, drivers are able to determine their current position via localization devices such as GPS or WiFi.

In *initialization phase*, the QSP first generates its key pair and divides the geography to be a public grid G composed of V rows and W columns. For each cell $v \times w$ of the grid, she assembles all related data as a message d_i, where $i = w + v \cdot W$, $0 \le v \le V-1$, and $0 \le w \le W-1$, and encrypts it as $d'_i = E_{m_i = k_v \oplus k_w}(d_i)$ by symmetric cryptosystem E according to the keys k_v, k_w designated to each row and column. Then, the QSP stores its key pair together with all k_v, k_w as well as d'_i in distributed RSUs.

In *retrieving phase*, the driver should complete both the payment and oblivious transfer process as follows.

In order to actualize the requirement of pay-per-retrieval for location-based service, the driver should ask for a random number $\overline{r}$ from its adjacent RSU and sign it using her private key corresponding to the valid digital certificate. After verifying the digital signature sent by the driver, the authentication server should launch a preconcerted E-commerce protocol to accomplish the transaction, resign the random number $\overline{r}$ in terms of her own private key, and then send it back to the driver. Availing herself of the signed random number, the driver can thus prove to the RSU that she has paid for the service.

After that, the driver is in a position to interact with the adjacent RSU and acquire k_{v_δ} as well as k_{w_δ} corresponding to her interested coordinates in the light of the aforementioned 1-out-n oblivious transfer protocol. Then she retrieves all encrypted messages and decrypts $d_\delta = D_{m_\delta = k_{v_\delta} \oplus k_{w_\delta}}(d'_\delta)$ to recover the data she expected.

It is worth noting that the driver may retrieve all encrypted messages only once and store some of them for further queries. In addition, even if the driver's identity is exposed during the authenticating process, it will not jeopardize the confidentiality of her queried position due to the intrinsic nature of oblivious transfer.

The process is illustrated in Table 5.

In fact, the aforementioned protocol can be regarded as being based on 2-out-n oblivious transfer since two symmetric keys k_{v_δ}, k_{w_δ} should be retrieved. However, all encrypted data need only to be transmitted once during retrieval phase, which means the extra computation and communication overheads are trivial. Moreover, the public key pair of driver is only used for authentication and payment but not necessarily for oblivious transfer.

4. Security Analysis

We investigate the server's data proprietorship and the driver's position privacy in oblivious transfer at first. It should be noted that the messages obliviously transferred are symmetric keys k_{v_δ}, k_{w_δ} instead of m_δ actually; however, we will alternatively apply these notations for smooth representation.

Table 5: The proposed privacy-preserving location-based querying scheme.

Central Server	*RSUs*
Holds $V \times W$ messages d_i, $i = w + v \cdot W \in \{0, 1, \ldots, V \times W - 1\}$.	
Initialization phase	
$(sk, pk) \leftarrow KeyGen(\kappa)$, $G \leftarrow Geography$, G is a $V \times W$ grid, For all $v \in \{0, 1, \ldots, V-1\}$ and $w \in \{0, 1, \ldots, W-1\}$, $k_v, k_w \leftarrow_R \{0,1\}^\kappa$, $m_i = k_w \oplus k_v$, $i = w + v \cdot W$, $d_i' = E_{m_i}(d_i)$, Put sk, pk and all k_v, k_w, d_i' in RSUs.	
	Receives and stores sk, pk as well as all k_v, k_w, d_i' locally.
Local RSU	*Driver*
Holds sk, pk and all k_v, k_w, d_i' as above.	Holds (v_δ, w_δ), $v_\delta \in \{0, 1, \ldots, V-1\}$, $w_\delta \in \{0, 1, \ldots, W-1\}$.
Authentication and transaction phase	
	Requests for service.
$\bar{r} \leftarrow_R \{0,1\}^\kappa$ and sends it to the driver.	
	$\mathrm{Sign}_{sk_{\mathrm{driver}}}(\bar{r})$, Completes the authentication as well as payment process with the central server.
	Receives $\mathrm{Sign}_{sk}(\bar{r})$ from the central server and sends it to the RSU.
Verifies $\mathrm{Sign}_{sk}(\bar{r})$.	
Retrieving phase	
	Retrieves all d_i' from local RSU.
Executes the NTRU-based OT protocol with the driver.	Executes the NTRU-based OT protocol with the RSU and gets k_{v_δ} and k_{w_δ}. $\delta = w_\delta + v_\delta \cdot W$, $m_\delta = k_{v_\delta} \oplus k_{w_\delta}$, $d_\delta = D_{m_\delta}(d_\delta')$.

As for the driver's position privacy, we claim the following.

Lemma 4. *The QSP gains no information on the driver's choice δ in the proposed OT protocol.*

Proof. Using the private key sk, the QSP can compute $c_\delta'' = s \cdot m_\delta (\mathrm{mod}\ p)$. However, she is ignorant of the driver's secret polynomial s and thereby cannot differentiate the choice δ from any other by comparing it with possessed messages, though the QSP may fortunately figure out $r' + r_\delta \cdot s (\mathrm{mod}\ q)$ if g is reversible, which means she can further achieve $(r' + r_\delta \cdot s) \cdot (s \cdot m_\delta)^{-1} = r' \cdot s^{-1} \cdot m_\delta^{-1} + r_\delta \cdot m_\delta^{-1}$. Nevertheless, since r' and s are uniformly distributed, m_δ is totally indistinguishable. □

The server's data proprietorship can be found as follows.

Lemma 5. *The driver gains no information on m_i if $i \neq \delta$.*

Proof. The driver is aware of $c_i = r_i \cdot h_S + m_i (\mathrm{mod}\ q)$ for all messages. Since she does not possess the server's private key, the mistiness of m_i from c_i is straight-forward.

With regard to the processes of authentication and transaction, the driver would interact with a central server directly to achieve a voucher signed by the QSP's private key. That means the RSU is incapable of linking the driver's current position up to her identity. Moreover, since the voucher is generated according to a provisional random number chosen by the RSU, the chance that a driver replay her voucher to cheat the QSP out of her service is negligible. Thanks to the intrinsic characteristic of OT, even if the identity of the driver is exposed in case that the RSU colludes with the central server, the confidentially of required coordinate would never be compromised. Supposing that the driver's identity privacy is obligatory in certain circumstances, anonymous authentication schemes such as that of [49] are further suggested. □

Now, we argue UC security of the complete scheme. In order to testify that a real-world implementation of our scheme is indistinguishable from its simulation, the ideal functionality is firstly defined as follows.

Definition 6. The ideal functionality $\mathcal{F}_{\mathrm{OT}}^{-}$ receives a coordinate $(v_\delta, w_\delta) \in \{0, 1, \ldots, V-1\} \times \{0, 1, \ldots, W-1\}$ together with an identity from the driver and a vector of l-bits messages, that is, $(d_0, d_1, \ldots, d_{V \times W - 1})$, from the server S, but only outputs a l-bits string d_δ to the driver D.

Table 6: Comparison of computation and communication overheads.

	QSP		Driver	
	Jannati and Bahrak	Ours	Jannati and Bahrak	Ours
Modular exponentiation	$3(V+W+1)$	0	9	0
Modular (polynomial) multiplication	$2(V+W)+1$	$V+W+4$	7	6
Generating random numbers (polynomials)	$V+W+2$	$V+W$	2	4
Secret keys	2	1	2	0
Public keys	2	1	0	0
Communication burden (polynomials)	$V+W+1$	$V+W+2$	2	2

In line with Definition 6, two simulators $\mathcal{S}_1$, $\mathcal{S}_2$ can be established to emulate the corrupted QSP and driver, respectively. Since

$$\Pr\left[\mathcal{F}_{\mathrm{OT}}^{-} = \delta\right] = \frac{1}{(V \times W)}, \tag{14}$$

it is obvious that

$$\left|\Pr\left[\mathcal{F}_{\mathrm{OT}} = \delta\right] - \Pr\left[\mathcal{F}_{\mathrm{OT}}^{-} = \delta\right]\right| < \varepsilon(\kappa) \tag{15}$$

according to Lemma 4. So the indistinguishability

$$\begin{aligned}&\mathcal{S}_1\left(\kappa, \mathrm{ID}_{\mathrm{driver}}, (d_0, d_1, \ldots, d_{V\times W-1})\right)\\ &\quad\cong \mathrm{VIEW}_{\mathrm{QSP}}\left(\mathrm{ID}_{\mathrm{driver}}, (d_0, d_1, \ldots, d_{V\times W-1}),\right.\\ &\quad\left.(v_\delta, w_\delta)\right)\end{aligned} \tag{16}$$

is straight-forward.

Similarly, once the symmetric cryptosystem (E, D) is noncommitting, the distributions of $d'_{i\neq\delta}$ in $\mathcal{F}_{\mathrm{OT}}$ and $\mathcal{F}_{\mathrm{OT}}^{-}$ are both uniform and indiscernible, which means

$$\Pr\left[\mathcal{F}_{\mathrm{OT}} = d'_{i\neq\delta}\right] = \Pr\left[\mathcal{F}_{\mathrm{OT}}^{-} = d'_{i\neq\delta}\right] = \frac{1}{(|M|-1)}, \tag{17}$$

where $|M|$ stands for the size of plaintext space, and

$$\begin{aligned}&\mathcal{S}_2\left(\kappa, (v_\delta, w_\delta), \left(d'_0, d'_1, \ldots, d'_{V\times W-1}\right)\right)\\ &\quad\cong \mathrm{VIEW}_{\mathrm{driver}}\left((v_\delta, w_\delta), \left(d'_0, d'_1, \ldots, d'_{V\times W-1}\right)\right),\end{aligned} \tag{18}$$

due to the ignorance of $k_{v_{i\neq\delta}}$ and $k_{w_{i\neq\delta}}$ on driver's side in terms of Lemma 5.

Thus we claim the folowing.

Theorem 7. *Our protocol securely implements the functionality $\mathcal{F}_{OT}^{-}$ if the symmetric encryption scheme (E, D) is noncommitting.*

5. Performance Evaluation

Since only simple polynomial multiplications are needed for NTRU cryptosystem, it features high speed, low memory requirements, and reasonably short and easily created keys. The moduli used in NTRUEncrypt specially are logarithmically smaller than that of traditional asymmetric cryptosystems based on integer factorization or discrete logarithm, which implies preferable efficiency and practicability. According to the report from [50], the speed of NTRU is up to 1300 times faster than 2048-bit RSA and 117 times faster than ECC NIST-224 when comparing the number of encryptions per second. Our experimental results also signified that the ratio of encryption times between 2048-bit ElGamal and NTRU in moderate security is 355 : 1.

In order to impartially compare with Jannati's protocol, only retrieval process will be considered in the following performance analysis. Though authentication and transaction are introduced in our scheme for pay-per-service purpose, the extra overheads are ineluctable but negligible compared to that of oblivious transfer. Table 6 illustrates the comparison of computation as well as communication overheads between our and Jannati's scheme. However, since the basic operations used in NTRU are absolutely different from that of ElGamal, it should be noted that modular multiplications and modular polynomial multiplications are correspondingly applied to one of them.

Compared to Jannati's protocol in Table 6, it is obvious that no exponentiations would be necessary in our scheme and the overhead of modular multiplication is also halved even without regard for the scale of moduli. It is worth mentioning that, though the number of transmitted messages are almost the same between Jannati's scheme and ours, we have evidently depressed the communication burden because a ElGamal encryption works on a large cyclic group and produces a double expansion in size from plaintext to ciphertext. Meanwhile, our scheme is more applicable since the receiver is free of generating or distributing any public key during oblivious transfer process.

We also simulated our and Jannati's protocol by C program. The experiment is carried out on an Intel Core i3-2330M processor (Sandy Bridge) where each party runs on one core. The computation burden and communication overhead for each retrieval are averaged by 500 tests.

According to Table 7, it is obvious that our scheme dramatically outperforms Jannati's protocol with respect to both computation and communication overheads. Specifically, taking the resource limits of OBU into account, the operational efficiency is 479 times that of Jannati's protocol, which means our scheme is more applicative in embedded and real-time environments. We simply neglected the delivery

Table 7: Timings in milliseconds and delivery loads in kilobytes for per retrieval (moderate security).

	QSP		Driver	
	Jannati and Bahrak	Ours	Jannati and Bahrak	Ours
Running time	993.23	3.26	47.85	0.10
Data size ($V + W = 64$)	34.12	8.10	1.08	0.23

load of queried data in our experiment; however, retrieving all d_i' indistinguishably from the server is inevitable due to the query privacy for any oblivious transfer. Fortunately, the driver can only retrieve the ciphered messages ones and keep all expected portions in the local storage, or she may ignore any other messages except for d_δ' when receiving the QSP's broadcast.

6. Conclusion

This paper proposed a privacy-preserving location-based querying scheme in virtue of NTRUEncrypt. Thanks to the intrinsic nature of NTRU cryptosystem such as postquantum security, high speed, low storage requirements, and short keys, our scheme is resistant to active adaptive corruptions and more practicable within vehicular ad hoc network. Specifically, the computational overheads are only 0.33 and 0.21 percent while the communication burdens are 24 and 21 percent compared to those of a recent scheme presented by Jannati and Bahrak Besides the theoretical and experimental performance analyses, we also depicted the detailed process of authentication and transaction for pay-per-service purpose. In the light of universal composability frame, it is believable that our scheme is secure with the functionality of oblivious transfer realized. For further work, we expect to reduce the interactive round of retrieving phase from 3 to 2 and decrease the RSU's overheads to a higher degree.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The work was supported by National Natural Science Foundation under Grants 61703063, 61663008, 61573076, and 61004118; the Scientific Research Foundation for the Returned Overseas Chinese Scholars under Grant 2015-49; the Program for Excellent Talents of Chongqing Higher School under Grant 2014-18; the Petrochemical Equipment Fault Diagnosis Key Laboratory in Guangdong Province Foundation under Grant GDUPKLAB201501; the Research Project for the Education of Graduate Students of Chongqing under Grant yjg152011; Chongqing Association of Higher Education 2015-2016 Research Project under Grant CQGJ15010C; Higher Education Reform Project of Chongqing Municipal Education Commission under Grant 163069; the Key Research Topics of the 13th Five-year plan of Chongqing Education Science under Grant 2016-GX-040; the Chongqing Natural Science Foundation under Grants CSTC2015jcyjA0540 and CSTC2017jcyjA1665; and Science and Technology Research Project of Chongqing Municipal Education Commission of China under Grants KJ1600518, KJ1705139, and KJ1705121.

References

[1] F.-Y. Wang, D. Zeng, and L. Yang, "Smart cars on smart roads: An IEEE intelligent transportation systems society update," *IEEE Pervasive Computing*, vol. 5, no. 4, pp. 68-69, 2006.

[2] H. Hasrouny, A. E. Samhat, C. Bassil, and A. Laouiti, "VANet security challenges and solutions: A survey," *Vehicular Communications*, vol. 7, pp. 7–20, 2017.

[3] J. Wang, Y. Zhang, Y. Wang, and X. Gu, "RPRep: A Robust and Privacy-Preserving Reputation Management Scheme for Pseudonym-Enabled VANETs," *International Journal of Distributed Sensor Networks*, vol. 12, no. 3-4, Article ID 6138251, pp. 1–15, 2016.

[4] B. Wiedersheim, Z. Ma, F. Kargl, and P. Papadimitratos, "Privacy in inter-vehicular networks: why simple pseudonym change is not enough," in *Proceedings of the IEEE/IFIP International Conference on Wireless On-Demand Network Systems and Services (WONS '10)*, pp. 176–183, February 2010.

[5] A. R. Beresford and F. Stajano, "Mix zones: user privacy in location-aware services," in *Proceedings of the 2nd IEEE Annual Conference on Pervasive Computing and Communications Workshops*, pp. 127–131, Orlando, Fla, USA, March 2004.

[6] B. Palanisamy and L. Liu, "MobiMix: protecting location privacy with mix-zones over road networks," in *Proceedings of the IEEE 27th International Conference on Data Engineering*, pp. 494–505, Hannover, Germany, April 2011.

[7] M. Gruteser and D. Grunwald, "Anonymous usage of location-based services through spatial and temporal cloaking," in *Proceedings of the 1st International Conference on Mobile Systems, Applications and Services*, pp. 31–42, ACM, San Francisco, Calif, USA, May 2003.

[8] C. Caballero-Gil, J. Molina-Gil, J. Hernández-Serrano, O. León, and M. Soriano-Ibañez, "Providing k-anonymity and revocation in ubiquitous VANETs," *Ad Hoc Networks*, vol. 36, pp. 482–494, 2016.

[9] X. Feng, C.-Y. Li, D.-X. Chen, and J. Tang, "A method for defensing against multi-source Sybil attacks in VANET," *Peer-to-Peer Networking and Applications*, vol. 10, no. 2, pp. 306–314, 2017.

[10] H. Kido, Y. Yanagisawa, and T. Satoh, "An anonymous communication technique using dummies for location-based services," in *Proceedings of the 2nd International Conference on Pervasive Services (ICPS '05)*, pp. 88–97, IEEE Press, July 2005.

[11] 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.

[12] L. Qi, X. Xu, X. Zhang et al., "Structural Balance Theory-based E-commerce recommendation over big rating data," *IEEE Transactions on Big Data*, p. 1, 2016.

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

[14] P. Shankar, V. Ganapathy, and L. Iftode, "Privately querying location-based services with sybilquery," in *Proceedings of the 11th ACM International Conference on Ubiquitous Computing, UbiComp'09*, pp. 31–40, Orlando, Fla, USA, October 2009.

[15] C. A. Ardagna, M. Cremonini, E. Damiani, S. de Capitani di Vimercati, and P. Samarati, "Location privacy protection through obfuscation-based techniques," in *Data and Applications Security XXI*, vol. 4602 of *Lecture Notes in Computer Science*, pp. 47–60, Springer, Berlin, Germany, 2007.

[16] C. Reynold, Y. Zhang, E. Bertino, and S. Prabhakar, "Preserving user location privacy in mobile data management infrastructures," in *Privacy Enhancing Technologies*, G. Danezis and P. Golle, Eds., vol. 4258 of *Lecture Notes in Computer Science*, pp. 393–412, Springer, Berlin, Germany, 2006.

[17] A. Gutscher, "Coordinate transformation - A solution for the privacy problem of location based services?" in *Proceedings of the 20th IEEE International Parallel and Distributed Processing Symposium, IPDPS 2006*, p. 7, IEEE, Rhodes Island, Greece, April 2006.

[18] J. Yang, Z. Zhu, J. Seiter, and G. Tröster, "Informative yet unrevealing: Semantic obfuscation for location based services," in *Proceedings of the 2nd ACM SIGSPATIAL Workshop on Privacy in Geographic Information Collection and Analysis, GeoPrivacy*, New York, NY, USA, 2015.

[19] F. Li, S. Wan, B. Niu, H. Li, and Y. He, "Time obfuscation-based privacy-preserving scheme for Location-Based Services," in *Proceedings of the 2016 IEEE Wireless Communications and Networking Conference Workshops, WCNCW 2016*, pp. 465–470, IEEE, Doha, Qatar, April 2016.

[20] J. D. Park, E. Seglem, E. Lin, and A. Züfle, "Protecting User Privacy," in *Proceedings of the the 1st ACM SIGSPATIAL Workshop*, pp. 1–4, Redondo Beach, CA, USA, November 2017.

[21] L. Qi, H. Xiang, W. Dou, C. Yang, Y. Qin, and X. Zhang, "Privacy-preserving distributed service recommendation based on locality-sensitive hashing," in *Proceedings of the 2017 IEEE International Conference on Web Services (ICWS)*, pp. 49–56, Honolulu, Hawaii, USA, June 2017.

[22] X. Zhang, L. Qi, W. Dou et al., "MRMondrian: Scalable Multidimensional Anonymisation for Big Data Privacy Preservation," *IEEE Transactions on Big Data*, p. 1, 2017.

[23] S. Wan, Y. Zhang, and J. Chen, "On the Construction of Data Aggregation Tree with Maximizing Lifetime in Large-Scale Wireless Sensor Networks," *IEEE Sensors Journal*, vol. 16, no. 20, pp. 7433–7440, 2016.

[24] W. Huang, S.-K. Oh, and W. Pedrycz, "Fuzzy Wavelet Polynomial Neural Networks: Analysis and Design," *IEEE Transactions on Fuzzy Systems*, vol. 25, no. 5, pp. 1329–1341, 2017.

[25] S. Wan, "Energy-efficient adaptive routing and context-aware lifetime maximization in wireless sensor networks," *International Journal of Distributed Sensor Networks*, vol. 10, no. 11, Article ID 321964, 2014.

[26] F. Dürr, P. Skvortsov, and K. Rothermel, "Position sharing for location privacy in non-trusted systems," in *Proceedings of the 9th IEEE International Conference on Pervasive Computing and Communications, PerCom 2011*, pp. 189–196, IEEE, Seattle, Wash, USA, March 2011.

[27] P. Skvortsov, F. Dürr, and K. Rothermel, "Map-aware position sharing for location privacy in non-trusted systems," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics): Preface*, vol. 7319, pp. 388–405, 2012.

[28] P. Skvortsov, B. Schembera, F. Dürr et al., Optimized Secure Position Sharing with Non-trusted Servers, 2017.

[29] J. Venkatanathan, J. Lin, and M. Benisch, Who, when, where: Obfuscation preferences in location-sharing applications, 2011.

[30] R. Yang, Q. Xu, M. H. Au, Z. Yu, H. Wang, and L. Zhou, "Position based cryptography with location privacy: A step for Fog Computing," *Future Generation Computer Systems*, vol. 78, pp. 799–806, 2018.

[31] S. Zeng, Y. Huang, and X. Liu, "Privacy-preserving communication for VANETs with conditionally anonymous ring signature," *International Journal of Network Security*, vol. 17, no. 2, pp. 135–141, 2015.

[32] Z. Zhao, J. Chen, and Y. Zhang, "Efficient revocable group signature scheme with batch verification in VANET," *Journal of Cryptologic Research*, vol. 3, no. 3, pp. 292–306, 2016.

[33] G. Ghinita, P. Kalnis, A. Khoshgozaran, C. Shahabi, and K.-L. Tan, "Private queries in location based services: anonymizers are not necessary," in *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD '08)*, pp. 121–132, ACM, 2008.

[34] S. Bittl, "Privacy conserving low volume information retrieval from backbone services in VANETs," *Vehicular Communications*, vol. 9, pp. 1–7, 2017.

[35] M. Hur and Y. Lee, "Privacy Preserving Top-k Location-Based Service with Fully Homomorphic Encryption," *Journal of the Korea Society For Simulation*, vol. 24, no. 4, pp. 153–161, 2015.

[36] P. Hu and S. Zhu, "POSTER: Location privacy using homomorphic encryption," *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, LNICST*, vol. 198, pp. 758–761, 2017.

[37] R. Paulet, M. G. Kaosar, X. Yi, and E. Bertino, "Privacy-preserving and content-protecting location based queries," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 5, pp. 1200–1210, 2014.

[38] H. Jannati and B. Bahrak, "An oblivious transfer protocol based on elgamal encryption for preserving location privacy," *Wireless Personal Communications*, vol. 97, no. 2, pp. 1–11, 2017.

[39] S. Tan, H. Mingxing E, and S. Zeng, "CCA secure extended ElGamal encryption scheme over CF (p~n)," *Journal of Xihua University*, vol. 36, no. 1, pp. 12–16, 2017.

[40] J. Hoffstein, J. Pipher, and J. H. Silverman, "NTRU: a ring-based public key cryptosystem," in *Algorithmic Number Theory*, vol. 1423, pp. 267–288, Springer, Berlin, Germany, 1998.

[41] H. B. A. Wahab and T. A. Jaber, "Improve NTRU algorithm based on Chebyshev polynomial," in *Proceedings of the 2015 World Congress on Information Technology and Computer Applications (WCITCA)*, pp. 1–5, IEEE, Hammamet, Tunisia, June 2015.

[42] D. H. Duong, M. Yasuda, and T. Takagi, "Choosing Parameters for the Subfield Lattice Attack Against Overstretched NTRU," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics): Preface*, vol. 10599, pp. 79–91, 2017.

[43] J. Hoffstein, J. Pipher, J. M. Schanck, J. H. Silverman, W. Whyte, and Z. Zhang, "Choosing parameters for NTRUEncrypt," *Lecture Notes in Computer Science (including subseries Lecture Notes*

in Artificial Intelligence and Lecture Notes in Bioinformatics): Preface, vol. 10159, pp. 3–18, 2017.

[44] https://tbuktu.github.io/ntru/.

[45] J. Hermans, F. Vercauteren, and B. Preneel, "Speed records for NTRU," in *Cryptographers' Track at the RSA Conference*, vol. 5985, pp. 73–88, Springer, Berlin, Germany, 2010.

[46] M. O. Rabin, "How To Exchange Secrets with Oblivious Transfer," *Cryptology ePrint Archive*, 2005.

[47] J. Kilian, "Founding cryptography on oblivious transfer," in *Proceedings of the 20th Annual ACM Symposium on Theory of Computing, STOC 1988*, pp. 20–31, Chicago, Ill, USA, May 1988.

[48] Y. Ishai, J. Kilian, K. Nissim, and E. Petrank, "Extending oblivious transfers efficiently," in *Advances in Cryptology CRYPTO*, vol. 2729, pp. 145–161, Springer, Berlin, Germany, 2003.

[49] R. Chen and D. Peng, "A novel NTRU-based handover authentication scheme for wireless networks," *IEEE Communications Letters*, vol. 22, no. 3, pp. 586–589, 2017.

[50] J. Hermans, F. Vercauteren, and B. Preneel, "Speed records for NTRU," in *Topics in Cryptology CT-RSA*, vol. 5985 of *Lecture Notes in Computer Science*, pp. 73–88, Springer, Berlin, Germany, 2010.

Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 6421607, 15 pages
https://doi.org/10.1155/2018/6421607

Research Article

Dynamic Resource Allocation for Load Balancing in Fog Environment

Xiaolong Xu,[1,2,3,4] **Shucun Fu**,[1,2] **Qing Cai**,[1,2] **Wei Tian**,[1,2] **Wenjie Liu**,[1,2] **Wanchun Dou**,[3] **Xingming Sun**,[1,2] **and Alex X. Liu**[1,4]

[1] *School of Computer and Software, Nanjing University of Information Science and Technology, Nanjing, China*
[2] *Jiangsu Engineering Centre of Network Monitoring, Nanjing University of Information Science and Technology, Nanjing, China*
[3] *State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China*
[4] *Department of Computer Science and Engineering, Michigan State University, East Lansing, MI, USA*

Correspondence should be addressed to Wanchun Dou; douwc@nju.edu.cn

Received 6 December 2017; Accepted 19 March 2018; Published 26 April 2018

Academic Editor: Deepak Puthal

Fog computing is emerging as a powerful and popular computing paradigm to perform IoT (Internet of Things) applications, which is an extension to the cloud computing paradigm to make it possible to execute the IoT applications in the network of edge. The IoT applications could choose fog or cloud computing nodes for responding to the resource requirements, and load balancing is one of the key factors to achieve resource efficiency and avoid bottlenecks, overload, and low load. However, it is still a challenge to realize the load balance for the computing nodes in the fog environment during the execution of IoT applications. In view of this challenge, a dynamic resource allocation method, named DRAM, for load balancing in fog environment is proposed in this paper. Technically, a system framework for fog computing and the load-balance analysis for various types of computing nodes are presented first. Then, a corresponding resource allocation method in the fog environment is designed through static resource allocation and dynamic service migration to achieve the load balance for the fog computing systems. Experimental evaluation and comparison analysis are conducted to validate the efficiency and effectiveness of DRAM.

1. Introduction

In recent years, the Internet of Things (IoT) has attracted attention from both industry and academia, which is beneficial to humans' daily lives. The data extracted from the smart sensors are often transmitted to the cloud data centers and the applications are generally executed by the processors in the data centers [1]. The cloud computing paradigm is efficient in provisioning computation and storage resources for the IoT applications, but the ever-increasing amount of resource requirements of the IoT applications leads to the explosively increased energy consumption and the performance degradation of the computing nodes due to data transmission and computing node migration; thus, how to perform IoT applications becomes an urgent issue [2–6]. Fog computing extends the computing process to the edge of the network rather than performing the IoT applications in the cloud platforms.

In the fog environment, the routers are the potential physical servers which could provision resources for the fog services at the edge of the network [7, 8]. The routers could enhance the performance of computation and storage, which could be fully utilized as the computing nodes. In the big data era, there are different performance requirements for the IoT applications, especially for the real-time applications; thus, such applications choose the edge computing nodes as a priority to host [9, 10]. In the fog environment, the users could access and utilize the computation, storage, and network resources, like the way the customers use the cloud resources, and the virtualized technology is also applicative to provision the on-demand resources dynamically [11]. The IoT applications could be performed by the fog computing nodes and the physical resources deployed in the remote cloud data center. The resource allocation for the IoT applications should take into account both the centralized and the geodistributed computing nodes, and the resource schedulers and managers

should choose the appropriate computing nodes to host the fog services combined in the IoT applications through the design of resource allocation strategies.

Resource allocation and resource scheduling are the key technologies to manage the data centers, which contribute a great deal to lowering the carbon emission, improving the resource utilization, and obtaining load balancing for the data centers [12–14]. In the cloud environment, the main goal of resource allocation is to optimize the number of active physical machines (PMs) and make the workloads of the running PMs distributed in a balanced manner, to avoid bottlenecks and overloaded or low-loaded resource usage [14–17]. In the fog environment, resource allocation becomes more complicated since the applications could be responded to by the computing nodes both in fog and in clouds. The computing nodes in the fog are distributed dispersedly in the network of edge, while the computing nodes in the cloud are distributed in a centralized data center. The resource requirements of the IoT applications for the computing nodes are various as the applications have different demands of computing power, storage capacity, and bandwidth. Therefore, it is necessary to undergo resource allocation for the dynamic resource requirements of the IoT applications, to achieve the goal of load balancing.

With the above observations, it is still a challenge to realize the load balance for the computing nodes in the fog environment during the execution of IoT applications. In view of this challenge, a dynamic resource allocation method, named DRAM, for load balancing in fog environment is proposed in this paper. Specifically, our main contributions are threefold. Firstly, we present a system framework for IoT applications in fog environment and conduct the load-balance analysis for various types of computing nodes. Then, a corresponding resource allocation method in the fog environment is designed through static resource allocation and dynamic service migration to achieve the load balance for the fog computing systems. Finally, adequate experimental analysis is conducted to verify the performance of our proposed method.

The rest of this paper is organized as follows. In Section 2, formalized concepts and definitions are presented for load-balance analysis in the fog environment. Section 3 elaborates the proposed resource allocation method DRAM. Section 4 illustrates the comparison analysis and performance evaluation. Section 5 summarizes the related work, and Section 6 concludes the paper and presents the prospect for the future work.

2. Preliminary Knowledge

In this section, a fog computing framework for IoT applications is designed and the load-balance analysis is conducted as well.

To facilitate dynamic resource allocation for load balancing in fog environment, formal concepts and load-balance analysis are presented in this section. Key notations and descriptions used in this section are listed in the section named "Key Terms and Descriptions Involved in Resource Scheduling in Fog Environment."

2.1. System Framework for Fog Computing. Fog computing is a new computing paradigm, which sufficiently leverages the decentralized resources through the fog and cloud environments, to provision the computation and storage services for the IoT applications. Fog computing extends the data process and data storage between the smart sensors and the cloud data centers. Some of the tasks from the IoT applications could be processed in the fog rather than performing all the tasks in the cloud environment. The virtualized technology could be employed to improve the resource usage in the fog environment.

Figure 1 shows a hierarchical framework for computing tasks from IoT applications in the fog environment. There are four layers in this framework, that is, the IoT application layer, the service layer, the fog layer, and the cloud layer. The IoT application contains a large amount of service requirements that need to be responded to by selecting appropriate computing nodes according to the time urgency and the resource amount from fog and cloud. The fog layer consists of the edge computing nodes and the intermediate computing nodes. The services with the highest time urgency and less computation density can be calculated by the edge computing nodes in the fog layer, and the less urgent tasks could choose intermediate computing nodes for execution. The cloud layer is appropriate to hosting the loose-fitting tasks with high-density computation and huge-volume storage which often demand a large amount of physical resources. The intermediate computing nodes could be the routers for data transmission, the edge computing nodes could be the mobile devices or sensors, and the computing nodes in the cloud are the PMs.

Fog computing is useful for IoT applications which combine many fog services. The fog services cover all the procedures of the data extraction, data transmission, data storage, and service execution from the IoT applications.

Definition 1 (fog service). The services requested by the IoT applications are available to be performed in the fog and remote cloud data centers, denoted as $S = \{s_1, s_2, \ldots, s_N\}$, where N is the number of fog services generated by the IoT applications.

The PMs in the remote cloud data centers, the intermediate computing nodes, and the edge computing nodes in the fog environment are all available to be leveraged to provision physical resources for the fog services. Suppose there are M computing nodes in the fog and cloud environment, denoted as $P = \{p_1, p_2, \ldots, p_M\}$. The virtualized technology has been widely applied in the cloud environment, which is also adaptive in the fog environment to measure the resource capacity of all the computing nodes and the resource requirement of the fog services.

Definition 2 (resource capacity of computing nodes). For all the computing nodes in the fog and cloud, their resource capacities are quantified as the number of resource units, and each resource unit contains various physical resources, including CPU, memory, and bandwidth.

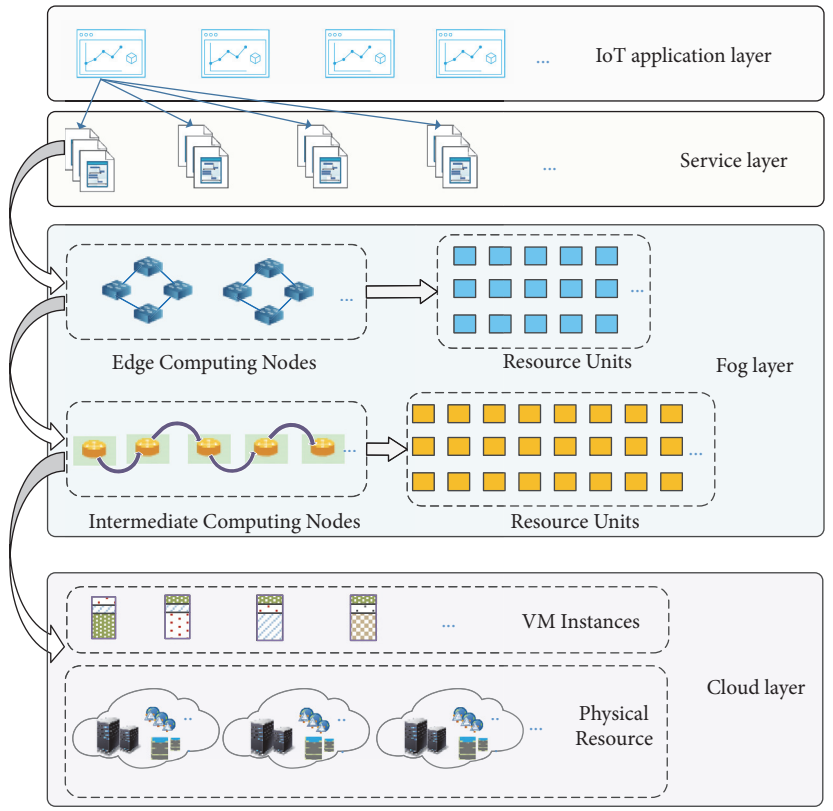


Figure 1: Fog computing framework for IoT applications.

Definition 3 (resource requirement of s_n). The corresponding resource requirement of the nth fog service s_n could be quantified as the time requirements, the resource type, and the resource amount to perform s_n, denoted as $r_n = \{stim_n, dutim_n, type_n, cou_n\}$, where $stim_n$, $dutim_n$, $type_n$, and cou_n represent the request start time, the duration time, the resource type, and the requested amount of s_n, respectively.

Note that the resource requirements in this paper are measured by the amount of resource units. For example, in the cloud data centers, the resource units are the VM instances, and the customers often rent several VM instances to host one application.

2.2. Load Balancing Model in Fog Environment. Fog computing paradigm releases the load distribution in the cloud environment. Due to the diversity of execution duration and specifications for the computing nodes in the fog, the resources could not be fully utilized. In the fog environment, we try to achieve load balancing for the computing nodes to avoid low utilization or overload of the computing nodes.

Let $I_n^m(t)$ be the binary variable to judge whether s_n $(1 \le n \le N)$ is assigned to the computing node p_m $(1 \le m \le M)$ at time instant t, which is calculated by

$$I_n^m(t) = \begin{cases} 1, & \text{if } s_n \text{ is assigned to } p_m, \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

With the judgement of fog service distribution, the resource utilization for the mth computing node p_m at time t is calculated by

$$ru_m(t) = \frac{1}{c_m} \sum_{n=1}^{N} I_n^m(t) \cdot r_n, \tag{2}$$

where c_m is the capacity of the computing node p_m.

According to the service distribution, the number of services deployed on p_m at time instant t is calculated by

$$ns_m(t) = \sum_{n=1}^{N} I_n^m(t). \tag{3}$$

The load-balance variance should be specified for each type of computing node. Suppose there are W types of computing nodes in cloud and fog environment for performing the fog services.

The load-balance variance is closely relevant to the resource utilization. To calculate the resource utilization, the employed amount of computing nodes with type w at the time instant t is calculated by

$$a_w(t) = \sum_{m=1}^{M} f_m(t) \cdot \beta_w^m, \tag{4}$$

where β_w^m is a flag to judge whether p_m is a type w of computing nodes, which is described in (5), and $f_m(t)$ is used to judge whether p_m is empty at t, presented in (6).

$$\beta_w^m = \begin{cases} 1, & \text{if } p_m \text{ is a } w\text{th type computing node,} \\ 0, & \text{otherwise.} \end{cases} \quad (5)$$

$$f_m(t) = \begin{cases} 1, & \text{if } ns_m > 0, \\ 0, & \text{otherwise.} \end{cases} \quad (6)$$

The resource utilization for the computing nodes with wth type at time t is calculated by

$$RU_w(t) = \frac{1}{a_w(t)} \sum_{m=1}^{M} \sum_{n=1}^{N} I_n^m(t) \cdot ru_m(t) \cdot f_m(t). \quad (7)$$

Definition 4 (variance of load balance of p_m at time instant t). The load-balance value is measured by the variance of the resource utilization. The variance value for p_m is calculated by

$$lb_m(t) = \left(ru_m(t) - \sum_{w=1}^{W} RU_w(t) \cdot \beta_w^m \right)^2. \quad (8)$$

Then, the average variance value for all the type w computing nodes is calculated by

$$LB_w(t) = \frac{1}{a_w(t)} \sum_{m=1}^{M} \sum_{n=1}^{N} I_n^m(t) \cdot lb_m(t) \cdot f_m(t). \quad (9)$$

For the execution period $[T_0, T]$ in the fog and cloud environment, the load-balance variance could be calculated by

$$LB_w = \frac{1}{T - T_0} \int_{T_0}^{T} LB_w(t)\, dt. \quad (10)$$

With these observations, the problem of minimizing the variance of load balance can be formulated as follows:

$$\min \quad LB_w, \quad \forall w = 1, \ldots, W \quad (11)$$

$$\text{s.t.} \quad a_w(t) \le \sum_{m=1}^{M} \beta_w^m, \quad (12)$$

$$\sum_{n=1}^{N} r_n \le \sum_{m=1}^{M} c_m, \quad (13)$$

$$\sum_{n=1}^{N} I_n^m r_n \le c_m, \quad (14)$$

where $\sum_{n=1}^{N} r_n$ in formula (13) represents the resource requirements for all services, $\sum_{m=1}^{M} c_m$ in formula (14) represents the capacity of all computing nodes, and $\sum_{n=1}^{N} I_n^m r_n$ in formula (14) represents the resource requirements for all services allocated to the type m computing node.

From (11) to (14), we can find that the final objective solved in this paper is an optimization problem with multiple constraints [18–20].

3. A Dynamic Resource Allocation Method for Load Balancing in Fog Environment

In this section, we propose a dynamic resource allocation method, named DRAM, for load balancing in fog environment. Our method aims to achieve high load balancing for all the types of computing nodes in the fog and the cloud platforms.

3.1. Method Overview. Our method consists of four main steps, that is, fog service partition, spare space detection for computing nodes, static resource allocation for fog service subset, and the load-balance driven global resource allocation, as shown in the section named "Specification of our Proposed Resource Allocation Method for Load Balancing in Fog Environment." In this method, Step 1 is the preprocess procedure, Step 2 is employed to detect the resource usage for Steps 3 and 4, and Step 3 is designed for static resource allocation for the fog services in the same subset and it provides the primary resource provision strategies for Step 4. Finally, Step 4 is a global resource allocation method to realize dynamic load balance.

Specification of Our Proposed Resource Allocation Method for Load Balancing in Fog Environment

Step 1 (fog service partition). There are different types of computing nodes for the performance of fog services. To efficiently provision resources, the fog services are classified as several sets based on the resource requirements of node type. Furthermore, these sets are divided into multiple subsets according to the request start time.

Step 2 (spare space detection for computing nodes). To judge whether a computing node is portable to host the fog service, it is necessary to detect the spare space of all the computing nodes. We analyze the employed resource units through the analysis of occupation records, and then the spare space of the computing nodes could be obtained.

Step 3 (static resource allocation for fog service subset). For the fog services in the same service subset, the proper computing nodes are identified to host these services. When allocating resource units for a fog service, the computing node with the least and enough spare space is selected. Besides, some workloads from the computing nodes with higher resource usage are migrated to the computing nodes with low resource usage.

Step 4 (load-balance driven global resource allocation). For all the fog service subsets, we could find the initialized resource allocation strategies in Step 4, and then the dynamic resource allocation adjustment is conducted at the competition moments of the fog services to achieve the global load balance during the execution period of the fog services.

3.2. Fog Service Partition. The fog services from different IoT applications have different requirements of computing resources; that is, the fog services need to choose different

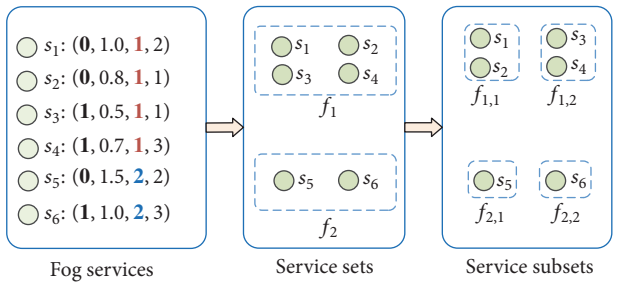


FIGURE 2: An example of subset acquisition with 6 fog services (i.e., s_1~s_6).

types of computing nodes for resource response. Suppose there are W types of processing nodes, including the PMs in cloud platforms, the intermediate nodes, and the edge nodes near the sensors.

In this paper, we try to achieve the goal of load balancing for each type of computing node; we assume that the fog services need to be performed with the same type of computing nodes. As a result, the fog services could be partitioned as W different service sets, denoted as $\mathbf{F} = \{\mathbf{f_1}, \mathbf{f_2}, \ldots, \mathbf{f_W}\}$.

The resource requirements of the fog services in the same set have different resource response time. To efficiently realize resource allocation for the services, the fog services in the same set should be partitioned to several subsets according to the start time for occupying the resource units of the computing nodes. Then, we can allocate resource units for the fog services in the same set to achieve high load balancing.

The subset $\mathbf{f_w}$ $(w = 1, 2, \ldots, W)$ in $\mathbf{F}$ is divided into multiple subsets according to the requested start time of the fog services. Let $\mathbf{f_{w,i}}$ be the ith $(1 \le i \le |\mathbf{f_w}|)$ subset contained in $\mathbf{f_w}$. After the partition process, the fog services in $f_{w,i}$ have the same request start time.

For example, there are 6 fog services, that is, s_1~s_6, and the resource requirements of these 6 services are s_1: (0, 1, **1**, 2), s_2: (0, 0.8, **1**, 1), s_3: (1, 0.5, **1**, 1), s_4: (1, 0.7, **1**, 3), s_5: (0, 1.5, **2**, 2), and s_6: (1, 1, **2**, 3), as shown in Figure 2. These 6 fog services are put in two different sets f_1 and f_2 according to the requested type of computing nodes, $f_1 = \{s_1, s_2, s_3, s_4\}$ and $f_2 = \{s_5, s_6\}$. Then, f_1 and f_2 are partitioned to the subsets according to the requested start time. Partition $f1$ is divided into 2 subsets, $f_{1,1} = \{s_1, s_2\}$ and $f_{1,2} = \{s_3, s_4\}$; meanwhile, f_2 is also separated as 2 subsets, that is, $f_{2,1} = \{s_5\}$ and $f_{2,2} = \{s_6\}$.

Algorithm 1 specifies the process of fog services subset acquisition. In Algorithm 1, the input is the resource requirements of IoT applications, that is, R, and the output is the partitioned fog service set $\mathbf{F}$. We traverse all the fog services (Line (1)), and the services are put in different sets according to the requested resource type (Lines (1) to (6)), and then the fog services are put in W different service sets. Then, the set $\mathbf{f_w}$ is divided to multiple subsets, according to the required start time (Lines (8) to (17)).

3.3. Spare Space Detection for Computing Nodes. The fog services need to be put on the computing nodes; thus, those computing nodes with spare space should be identified. For all the computing nodes, the allocation records are employed

Input: The resource requirements of IoT applications R
Output: The partitioned fog service set $\mathbf{F}$
(1) for $i = 1$ to N **do**
(2) **for** $j = 1$ to W **do**
//There are W types of computing nodes in fog and cloud
(3) **if** $type_i == j$ **then**
(4) Add r_i to f_i
(5) **end if**
(6) **end for**
(7) end for
(8) for $i = 1$ to W **do**
(9) $nn = 0, q = 0, f = stim_0$
(10) **while** $nn < |fs_i|$ **do**
(11) **if** $stim_q \le f$ **then**
(12) Add the mth fog service to $fs_{i,nn}$
(13) **else** $nn = nn + 1, q = q + 1, f = stim_q$
(14) Add the qth fog service to $fs_{i,nn}$
(15) **end if**
(16) **end while**
(17) end for
(18) Return F

ALGORITHM 1: Fog service subset acquisition.

to monitor the resource usage of all the computing nodes in the fog and cloud.

Definition 5 (occupation record $rs_{m,i}$). The ith $(1 \le i \le |rs_m|)$ occupation record in rs_m contains the response fog service, the occupation start time, the duration time, and the resource units, which is a 4-tuple, denoted as $rs_{m,i} = (fs_{m,i}, st_{m,i}, dt_{m,i}, us_{m,i})$, where $fs_{m,i}$, $st_{m,i}$, $dt_{m,i}$, and $us_{m,i}$ are the fog service, the start time, the duration time, and the resource unit sets of $rs_{m,i}$, respectively.

A computing node has a set of occupation records, since it could host several fog services. The record set for the computing node p_m $(1 \le m \le M)$ is recorded as rs_m. Then, for all the computing nodes in the fog and cloud, there are M occupation record sets, denoted as $RS = \{rs_1, rs_2, \ldots, rs_M\}$. In rs_m, there are many occupation records, which reflect the resource usage of the computing node p_m.

The occupation records are dynamically updated according to the real-time resource provisioning for the fog services. Once the fog services are moved to the other computing nodes during their lifetime, the occupation time parameter should be updated accordingly for the relevant records. The occupation records are generated from the users when they apply for resources in the fog and cloud data center for implementing the services generated from the IoT applications.

Benefiting from the resource monitoring, the occupied resource units of all the computing nodes and the spare units could be detected for resource allocation at any time.

Definition 6 (spare space of p_m). The spare space of the computing node p_m is defined as the idle amount of resource units, which is measured by the difference value of the capacity and the occupied amount of resource units on p_m.

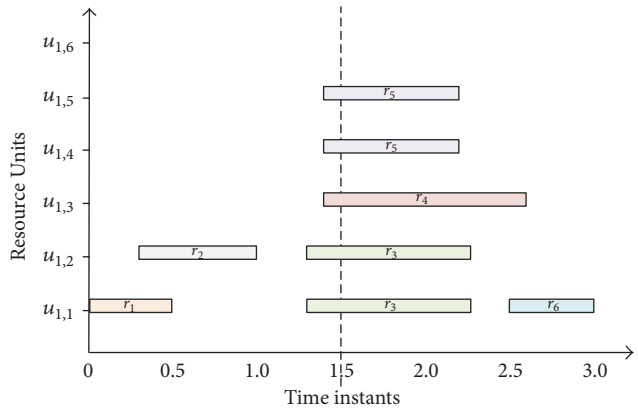


Figure 3: An example of spare space detection with 6 occupation records (i.e., r_1~r_6).

Input: The occupation record for the computing node p_m
Output: The spare space for p_m
(1) $cou = 0$
(2) for $i = 1$ to $|rs_m|$ **do**
(3) $ct_{m,i} = st_{m,i} + dt_{m,i}$
//$ct_{m,i}$ is the finish time for the occupation of resource units
(4) **if** $t \geq st_{m,i}$ && $t < ct_{m,i}$ **then**
//t is the request time for statistics
(5) $cou = cou + |us_{m,i}|$
(6) **end if**
(7) end for
(8) $cou = c_m - cou$
(9) Return cou

Algorithm 2: Spare space detection for computing nodes.

The spare space of the computing node p_m could be detected from the analysis of occupation records. In these records, if occupation start time is less than the statistic time instant for checking the PM status and the occupation finish time is over the statistic time, the relevant resource units combined in the occupation records could be obtained. With these acquired resource units and the resource capacity of p_m, the spare space could be finally detected.

For example, there are 6 occupation records for the computing node p_1, that is, $r_1 : (1, \mathbf{0}, \mathbf{0.5}, \{u_{1,1}\})$, $r_2 : (2, \mathbf{0.3}, \mathbf{0.7}, \{u_{1,2}\})$, $r_3 : (3, \mathbf{1.3}, \mathbf{1}, \{u_{1,1}, u_{1,2}\})$, $r_4 : (4, \mathbf{1.4}, \mathbf{1.2}, \{u_{1,3}\})$, $r_5 : (5, \mathbf{1.4}, \mathbf{0.8}, \{u_{1,4}, u_{1,5}\})$, and $r_6 : (6, \mathbf{2.5}, \mathbf{0.5}, \{u_{1,1}\})$, as shown in Figure 3. If the statistic instant for spare space detection is 1.5, the identified occupation records within the statistic time are r_3, r_4, and r_5. Based on the analysis of the occupation records, the employed resource units at this moment are $u_{1,1}, u_{1,2}, u_{1,3}, u_{1,4}$, and $u_{1,5}$. Suppose the capacity of p_1 is 6; then, the spare space of p_1 at time instant 1.5 is 1.

Algorithm 2 specifies the key idea of spare space detection for computing nodes. In Algorithm 2, the input and the output are the occupation records of the computing node p_m and the spare space for p_m. According to Definition 7, we need to calculate the occupation amount of resource units on p_m first (Lines (1) to (8)), and then the spare space could be detected (Line (8)).

3.4. Static Resource Allocation for Fog Service Subset. Based on the fog service partition in Section 3.1 and spare space detection for computing nodes in Section 3.2, the fog services that need to be processed and the available computing resources for fog service execution are identified, which are all beneficial for resource allocation.

In the fog environment, the fog services need to be responded to by the computing nodes, and the time requirements also should be presented when allocating the resources to the fog services. In this section, we define the resource allocation records to reserve the allocation history about resource provisioning for the fog services.

Definition 7 (resource allocation record for s_n). The resource allocation record for s_n consists of the node type, the number of resource units, the start time, and the desired duration time, which is denoted as $a_n = (nt_n, num_n, st_n, dt_n)$, where nt_n, num_n, st_n, and dt_n are the node type, the amount of resource units, the service start time, and the resource occupation time for s_n, respectively.

The fog services in the same fog service subset have the same required node type and start time. When allocating resource units for the fog service subset, each fog service should find a portable computing node to host it. Thus, we need to find the available nodes first. The computing nodes with the requested type, which have spare space, which could be detected by Algorithm 2, are chosen as the candidate resources to be provided for the fog services in the subset.

To achieve the load balancing of the computing nodes, we try to achieve high resource usage of the employed computing nodes. The problem of static resource allocation is like bin packing problem, which is NP-hard. Here, we leverage the idea of Best Fit Decreasing to realize the process of computing node matching for the fog services. Before resource allocation, the fog services are sorted in the decreasing order of requested resource amount. The service with more required resource units will be processed first. And it chooses the computing node with the least and enough spare space for hosting the service.

For example, there are 3 computing nodes P_1, P_2, and P_3, and the spare spaces of these 3 computing nodes are 4, 6, and 9, respectively, as shown in Figure 4. There are two fog services in the same subset, that is, S_1 and S_2, and the requested resource amounts of these two services are 2 and 6. When conducting resource allocation for S_1 and S_2, S_2 has more resource requirements, and thus S_2 is processed first. After resource allocation, S_2 chose P_3 for hosting and S_1 chose P_1 for hosting.

The above allocation may lead to the unbalanced distribution of the workloads of some computing nodes. In this section, a threshold ρ is employed to judge whether the computing node is in low resource utilization. If a computing node is in low resource utilization and there are no other computing nodes that could host the workloads in this computing node, we choose to move some workloads to this computing node to improve the load balance.

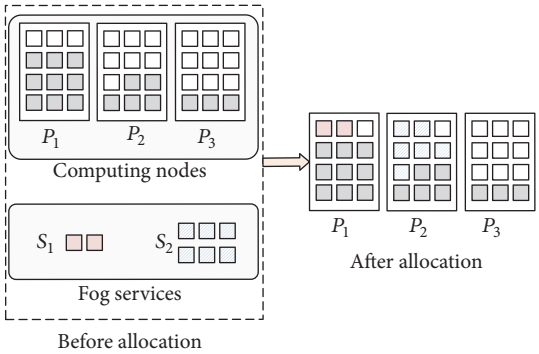


Figure 4: An example of static resource allocation for fog services S_1 and S_2, with computing nodes P_1, P_2, and P_3.

After resource allocation, there are several allocation records generated, to record the allocation history for the fog services in the service subset. Meanwhile, there are some computing nodes, providing resource units for executing the fog services, which also generate some occupation records.

Algorithm 3 illustrates the key idea of static resource allocation for fog service subset $\mathbf{f_{w,i}}$. The input of Algorithm 3 is the fog service subset $\mathbf{f_{w,i}}$, and the output of this algorithm is the resource allocation records for the fog services in $\mathbf{f_{w,i}}$. The required computing nodes of fog services in the same subset have the same node type, which should be obtained first (Line (1)). The services with fewer requested resource units will be processed first, so $\mathbf{f_{w,i}}$ should be sorted in the increasing order of the amount of values of required resources. Then, each service could be responded to with computing nodes sequentially (Line (3)). When selecting a computing node to provision resources for a fog service, the available computing nodes with enough spare space to host the service, calculated by Algorithm 2, should be achieved first (Lines (4) to (11)). The computing nodes also should be sorted in the increasing order of spare space (Line (12)). Then, the computing node with the least spare space will be selected to accommodate the service (Line (13)). Some workloads from the computing nodes with higher resource usage will be migrated to the computing nodes with low resource utilization to improve load balance (Lines (15) to (24)). Finally, the relevant occupation records and the resource allocation record for the fog service should be updated (Lines (25) and (26)).

3.5. Load-Balance Driven Global Resource Allocation. From the analysis in Sections 3.2 and 3.4, the initialized resource allocation is conducted, and the different types of computing nodes could achieve high resource utilization and load balancing at the allocation moment. However, the static resource allocation only could achieve the temporary load balancing at the service arrival moments. During the service running, the resource usage of the computing nodes is dynamically changed due to the various lifetimes of the fog services. In this section, a global resource allocation strategy is designed to make load balancing come true during the execution of the fog services.

Input: The fog service subset $\mathbf{f_{w,i}}$
Output: The relevant resource allocation records
(1) Get the node type nt in the service subset $\mathbf{f_{w,i}}$
(2) Sort $\mathbf{f_{w,i}}$ in decreasing order of required resource amount
(3) for each fog service in $\mathbf{f_{w,i}}$ **do**
(4) **for** $i = 1$ to M **do**
(5) **if** p_m has the same type with nt **then**
(6) Get the spare space by **Algorithm 2**
(7) **if** p_m has enough space to host the service **then**
(8) Add the computing node to CL
(9) **end if**
(10) **end if**
(11) **end for**
(12) Sort CL in increasing order of spare space
(13) Put the service in the first computing node in CL
(14) end for
(15) flag = 1, $i = 1$
(16) Identify the occupied computing nodes from CL to CL'
(17) Sort CL' in the decreasing order of spare space
(18) while flag == 1 **do**
(19) **if** the resource usage of cl'_i is less than ρ **then**
(20) Select the tasks to migrate to the computing node
(21) $i = i + 1$
(22) **else** flag = 0
(23) **end if**
(24) end while
(25) Update the relevant occupation records
(26) Generate an allocation records

Algorithm 3: Static resource allocation for fog service subset.

The fog service subsets demand the same type of computing nodes for hosting sequentially according to the requested start time of the resources. Let $bt_{w,i}$ be the requested start time for resource occupation of the ith subset $f_{w,i}$ in $\mathbf{f_w}$. For dynamic adjustment of resource allocation at the finish time of the fog services in $bt_{w,i}$ ($1 \leq i < |\mathbf{f_w}|$), the scheduling time should be with the time period $(bt_{w,i}, bt_{w,i+1})$. When $i = |\mathbf{f_w}|$, the scheduling time should be the competition time for the rest of the fog service loads during the period for the execution of the fog services in $\mathbf{f_{w,i}}$.

At the scheduling time, the running workloads occupy some computing nodes, and these computing nodes may be with low resource usage due to the service competition. The computing nodes with the same type in fog or cloud could be sorted in the decreasing order of the resource usage. The workloads in the computing nodes with the lower resource usage could be migrated to the computing nodes with higher resource usage, to achieve higher resource utilization. Besides, the migration of workloads could also help to realize the goal of load balancing as the computing nodes with more spare space could be moved vacant and shut down further.

For the workloads from different fog services, it is necessary to find the destination computing nodes to host them. The selection of destination computing nodes decides on the resource requirements of the workloads and the spare space of the computing nodes. If all the workloads from the

Input: The fog service set S
Output: The resource allocation records
The occupation records on computing nodes
(1) Obtain fog service subset **F** by **Algorithm 1**
(2) for $i = 1$ to W **do**
(3) **for** $j = 1$ to $|f_w|$ **do**
(4) **Algorithm 3** Static resource allocation for $\mathbf{f_{w,j}}$
(5) Calculate CT
// CT is the competition time list
// CT = $\{ct_1, ct_2, \ldots, ct_K\}$
(6) **for** $k = 1$ to K **do**
(7) Update the current run list in $\mathbf{f_{w,j}}$
(8) **for** $l = 1$ to M **do**
(9) Get spare space by **Algorithm 2** at ct_k
(10) **if** p_l has spare space and is not empty **then**
(11) Add p_l to SL
(12) **end if**
(13) **end for**
(14) Sort SL in increasing order of spare space
(15) flag = 1, $q = 1$
(16) **while** flag == 1 **do**
(17) Get the occupied resources sets on sl_q
(18) **for** each occupied resource set **do**
(19) Confirm the destination PM
(20) **end for**
(21) **if** the resource sets can be moved **then**
(22) $q = q + 1$
(23) Update the relevant allocation records
(24) Update the occupation records
(25) **else** flag = 0
(26) **end if**
(27) **end while**
(28) **end for**
(29) **end for**
(30) end for

Algorithm 4: Load-balance driven global resource allocation.

same computing node could find the destination computing nodes, these workloads could be migrated to the destination computing nodes. Finally, the resource allocation records and the occupation records are generated or updated according to the real occupation computing nodes and the usage time of the corresponding resource units.

Algorithm 4 illustrates the key process of load-balance driven global resource allocation. The key idea of Algorithm 4 is to conduct static resource allocation for the fog services at the start execution time and dynamically adjust the service placement according to the resource usage of all the employed computing nodes. The input of this algorithm is the fog service set S, and the final output of this algorithm is the resource allocation records and the occupation records. In this algorithm, the fog service subset is achieved first by Algorithm 1 (Line (1)), and then we traverse each subset for resource allocation (Line (2)) and conduct static resource allocation for the subsets by Algorithm 3 (Line (3)). Then, for each subset, the competition time list CL is extracted for load-balance driven dynamic resource allocation (Line (5)). Then, at each competition instant, we adjust the resource allocation records for the running services (Line (6)). The fog services on the computing node with less spare space would be moved to the computing node with higher resource usage, which has enough spare space to host the services (Lines (8) to (20)). When all the fog services on a computing node could find the destination node, the relevant allocation records and the occupation records for the resource units will be generated and updated (Lines (15) to (27)).

Table 1: Parameter settings.

Parameter	Domain
Number of fog services	{500, 1000, 1500, 2000}
Number of computing nodes	3000
Number of node types	3
Resource capacity	{7, 12, 18}
Resource requirements of fog services	[1, 15]
Duration for each service	[0.1, 4.8]

4. Experimental Analysis

In this section, the cloud simulator Cloudsim is applied to evaluate our proposed method DRAM. The intermediate computing nodes and the edge computing nodes are simulated as two computing data centers. The resource allocation method for fog environment is NP-hard, like the bin packing problem; thus, the typical and efficient resource allocation methods FF, BF, FFD, and BFD are employed for comparison analysis.

4.1. Experimental Context. To discuss the effectiveness of DRAM, 4 datasets with different scale of fog services are utilized, which are shared at https://drive.google.com/drive/folders/0B0T819XffFKrZTV4MFdzSjg0dDA?usp=sharing. The parameters for experimental evaluation are presented in Table 1.

The fog services employ three types of computing nodes, that is, the edge computing node, the intermediate node, and the PMs in cloud for resource response. The number of services for each type of computing node contained in the 4 different datasets is shown in Figure 5. For example, when the number of fog services is 1000, there are 324 fog services that need edge computing nodes, 353 fog services that need intermediate computing nodes, and 323 fog services that need PMs in the remote cloud for resource response.

4.2. Performance Evaluation. Our proposed method tends to minimize the load-balance variance, which is relevant to the resource utilization of each computing node and the average resource utilization. Therefore, we conduct performance evaluation for this fog computing system on the employed number of computing nodes, resource utilization, and load-balance variance.

(1) Performance Evaluation on the Employed Number of Computing Nodes. The number of the computing nodes could reflect the efficiency of resource usage. Figure 6 shows the comparison of the number of employed computing nodes

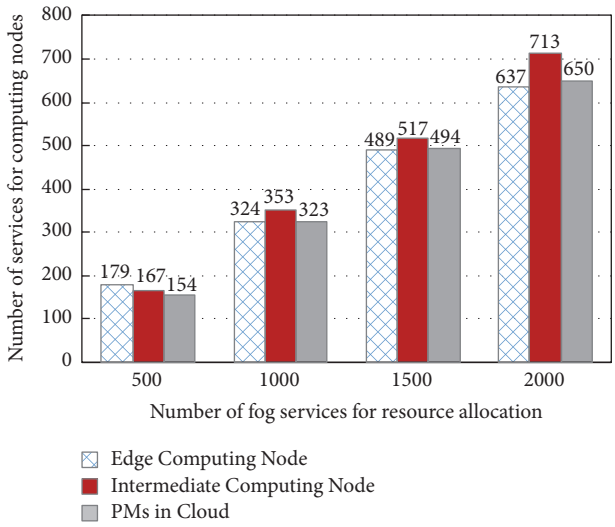


Figure 5: Number of fog services for the 3 types of computing nodes with different datasets.

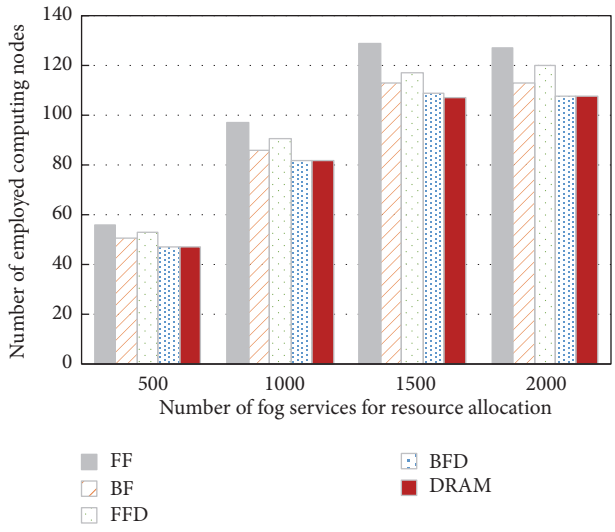


Figure 6: Comparison of the number of employed computing nodes by FF, BF, FFD, BFD, and DRAM with different datasets.

by FF, BF, FFD, BFD, and DRAM by using the 4 different scales of datasets. From Figure 6, we can find that our proposed method DRAM as well as BFD could employ fewer computing nodes, compared with FF, BF, and FFD, when the number of fog services is 500, 1000, and 2000. When the number of fog services is 1500, our proposed DRAM could even get higher efficiency on the employed number of computing nodes than BFD. In the process of static resource allocation, DRAM leverages the basic idea of BFD; thus, in most cases, DRAM and BFD have similar performance on the amount of employed computing nodes. But when some of the computing nodes are spared through the process of DRAM, thus in some cases, DRAM is superior to BFD.

As there are 3 types of computing nodes in our experimental evaluations, we should evaluate DRAM for the different types of computing nodes, compared to the other four methods. The 4 subfigures in Figure 7 show the comparison of the number of the employed computing nodes with different types by FF, BF, FFD, BFD, and DRAM using different datasets. It is intuitive from Figure 7 that our proposed method DRAM is fit for all kinds of computing nodes, which employs fewer computing nodes than FF, BF, and FFD, and gets similar performance to BFD in most cases.

(2) Performance Evaluation on Resource Utilization. The resource utilization is a key factor to decide the load-balance variance; thus we evaluate this value to discuss the resource usage achieved by FF, BF, FFD, BFD, and DRAM with different datasets. The resource utilization is referenced to the resource usage of the resource units on the computing nodes. Figure 8 shows the comparison of average resource utilization by FF, BF, FFD, BFD, and DRAM with different datasets. It is intuitive from Figure 8 that DRAM could obtain better resource utilization than FF, BF, FFD, and BFD, since DRAM is a dynamic and adaptive method which could adjust the load distribution during the fog service execution.

Similar to the evaluation on the employed amount of computing nodes, the performance evaluation on resource utilization is conducted from the perspective of the different types of the computing nodes. Figure 9 shows the comparison of resource utilization for different types of computing nodes by FF, BF, FFD, BFD, and DRAM with different datasets. From Figure 9, we can find that DRAM could achieve higher resource utilization than FF, BF, FFD, and BFD. For example, in Figure 9(c), when the number of fog services is 1500, DRAM obtains the resource utilization over 80%, whereas FF, BF, FFD, and BFD obtain near or below 70% resource utilization for each type of computing node.

(3) Performance Evaluation on Load-Balance Variance. The evaluation of the load-balance variance is also conducted by FF, BF, FFD, BFD, and DRAM using 4 different scales of datasets. Figure 10 shows the comparison of average load-balance variance, where we can find that our proposed method DRAM is superior to the other methods, that is, FF, BF, FFD, and BFD. For example, when the number of fog services is 500, the load-balance variance obtained by DRAM is near 2.5×10^{-2}, whereas FF, BF, FFD, and BFD obtain the load-balance value over 3×10^{-2}.

The evaluation on the load-balance variance also should take into consideration the computing node type. Figure 11 shows the comparison of load-balance variance values for different types of computing nodes by FF, BF, FFD, BFD, and DRAM with different datasets. From Figure 11, we can find that when changing the scale of datasets, our method can keep the priority on the load-balance variance for each type of computing node.

5. Related Work

The IoT technology has been widely applied in many fields, including weather forecasting and traffic monitoring. The

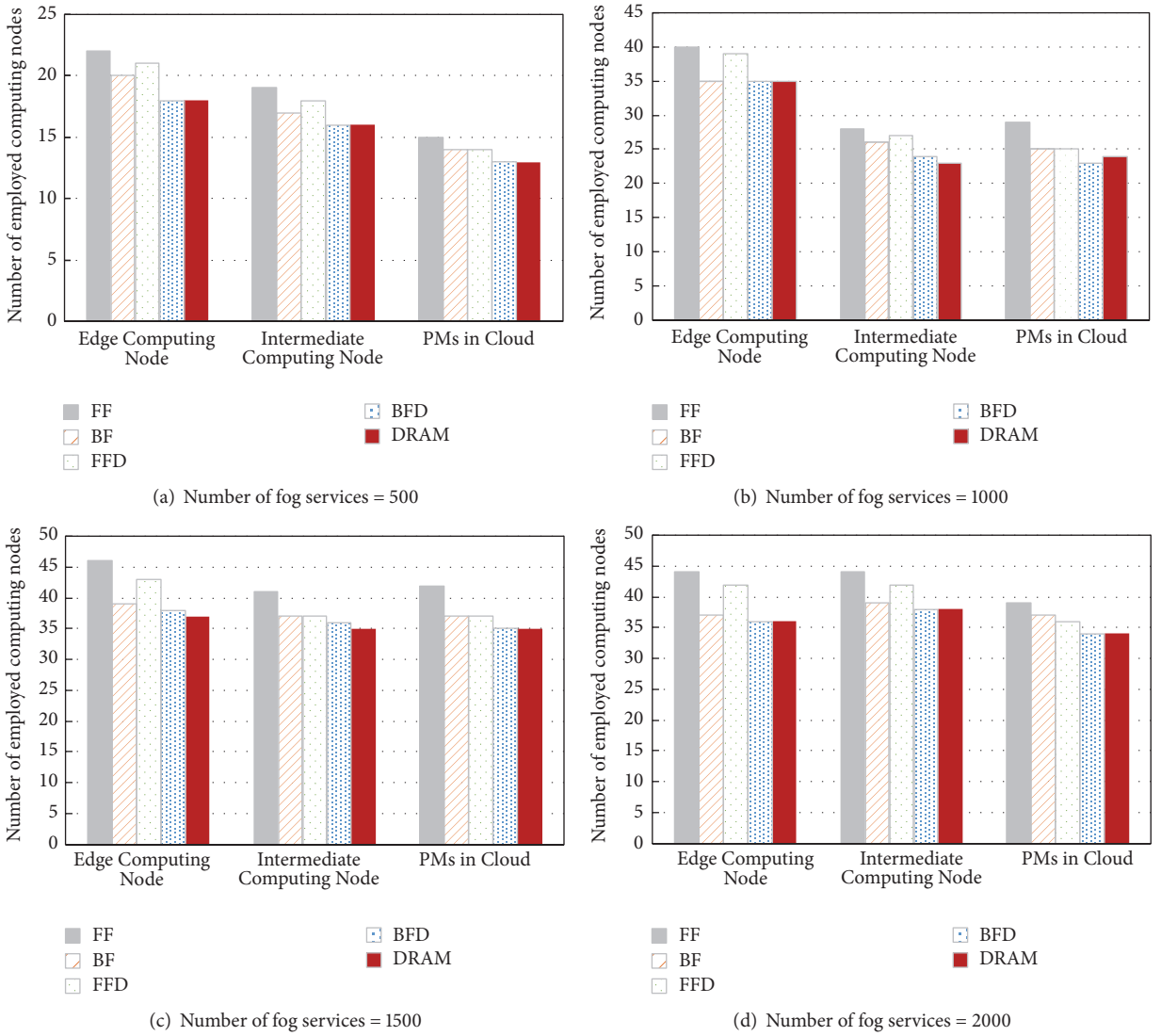


FIGURE 7: Comparison of the number of the employed computing nodes with different types by FF, BF, FFD, BFD, and DRAM using different datasets.

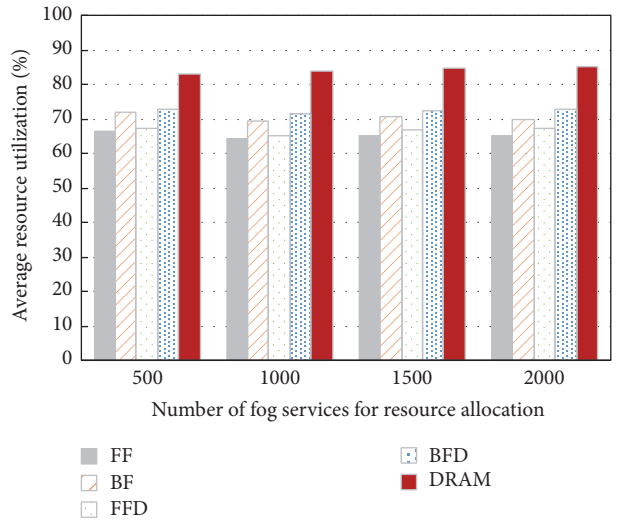


FIGURE 8: Comparison of average resource utilization by FF, BF, FFD, BFD, and DRAM with different datasets.

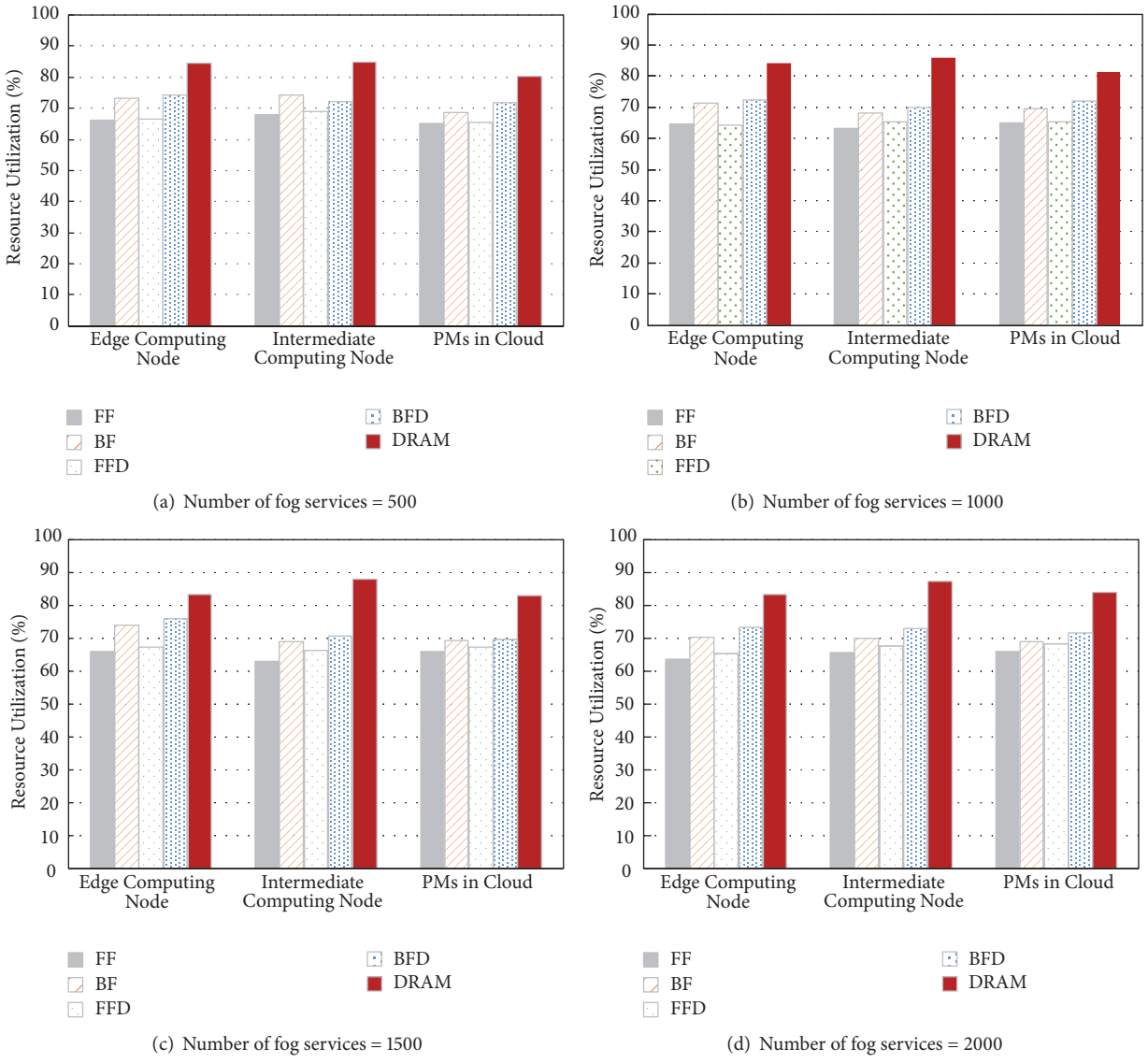


FIGURE 9: Comparison of resource utilization for different types of computing nodes by FF, BF, FFD, BFD, and DRAM with different datasets.

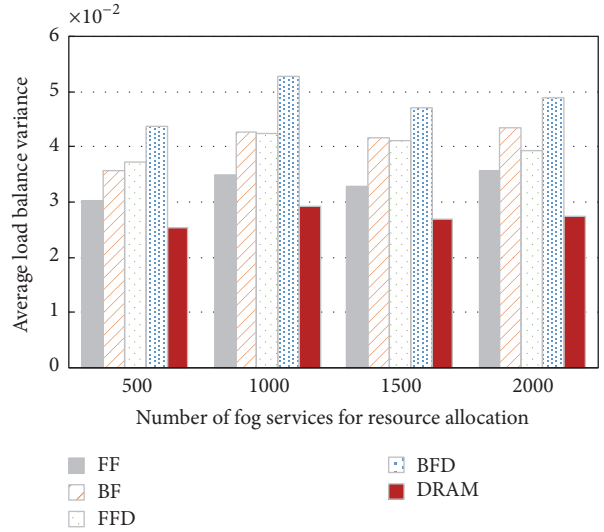


FIGURE 10: Comparison of average load-balance variance by FF, BF, FFD, BFD, and DRAM with different datasets.

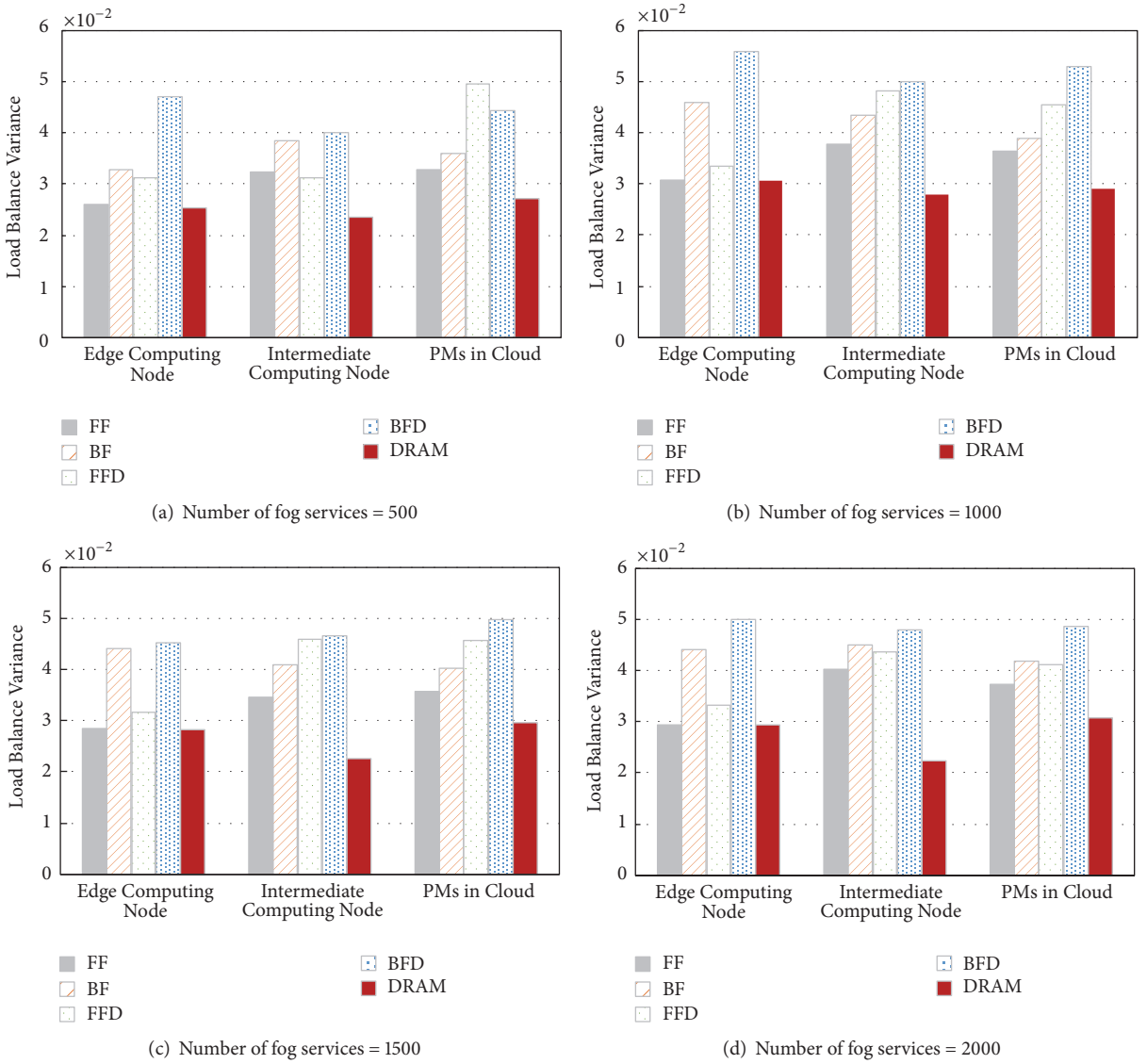


FIGURE 11: Comparison of load-balance variance values for different types of computing nodes by FF, BF, FFD, BFD, and DRAM with different datasets.

data storage and processing usually benefit from the cloud computing which provides scalable and elastic resources for executing the IoT applications [21–26]. In the ever-expanding data volume, cloud computing is difficult to provide efficient, low-latency computing services, and fog computing is proposed to complement the above shortage of cloud computing [1, 2].

Compared to the remote cloud computing center, fog computing is closer to the Internet of Things devices and sensors. Fog computing can quickly solve lightweight tasks with fast response. In the era of big data, with cloud computing expansion, fog computing is widely used in the medical, transportation, and communication fields, to name a few [21, 27–30].

Hu et al. [21] designed a fog calculation framework in detail and compared it with the traditional cloud computing, and a practical application case in the fog computing environment was put forward. Similarly, Kitanov and Janevski [27] compared the cloud computing on computing performance with fog computing in the 5G network, reflecting the superiority of fog computing. Akrivopoulos et al. [28] presented a technology to respond to the combination of IoT applications and the fog computing and introduced the use of fog computing in an automated medical monitoring platform to improve the medical work for the patients. Arfat et al. [29] not only unprecedentedly proposed the integration of mobile applications, big data analysis, and fog computing, but also introduced Google Maps as an example, showing the system of information feedback diversification. Taneja and Davy [30] studied the computational efficiency in the fog environment and constructed a resource-aware placement.

Generally, resource allocation refers to the allocation of specific, limited resources and effective management to achieve the optimal use of resources. The original intention of cloud computing is to allocate network resources on demand, so that it is the same as the use of water and electricity billing

[31, 32]. In the cloud computing, resource allocation method can effectively help to achieve the goal of high resource usage and energy saving for centralized resource management for different types of applications [4, 33–35]. Fog computing, as an extension of cloud computing paradigm, also needs to conduct resource allocation to achieve high-efficiency resource usage.

Mashayekhy et al. [36] proposed an auction-based online mechanism; it can access in real time the actual needs of users and the allocation of appropriate resources to the user price. Kwak et al. [37] developed a DREAM algorithm for complex tasks in mobile devices, saving 35% of total energy and managing network resources. To address the challenges of high latency and resource shortage in clouds, Alsaffar et al. [38] proposed the resource management framework, collaborating the cloud computing and the fog computing, and then they optimized the resource allocation in fog computing. Xiang et al. [39] designed a RAN (F-RAN) architecture based on atomization calculation, which effectively achieved the high resource usage and could coordinate the global resource scheduling.

Load balancing is an effective factor to determine the resource allocation strategy. For multiple computing tasks, load balancing could promote the resource managers to assign these tasks to multiple computing nodes for execution. The realization of load balancing not only can save the cost of hardware facilities but also can improve resource efficiency.

Banerjee and Hecker [40] proposed a distributed resource allocation protocol algorithm to realize load balancing in a large-scale distributed network; as a result, compared to the FIFO, the response time and resource utilization could be greatly improved. Govindaraju and Duran-Limon [41] designed a method based on the lifecycle-related Service Level Agreement (SLA) parameters of the virtual machines in cloud environment to address resource utilization and cost issues. Evolution algorithms are proved to be powerful to solve the multiobjective problem, which could be leveraged in the resource scheduling in the fog computing [42]. Jeyakrishnan and Sengottuvelan [43] developed a new algorithm, while saving operating costs, while maximizing the use of resources, in the balanced scheduling compared to SA, PSO, and ADS being more outstanding.

For the load balancing maximization problem solved in this paper, the traditional operations research is proved to be efficient in optimization problem with constraints [44, 45]. The game theory is also efficient for the resource allocation with resource competition, and the Nash Equilibria are often needed to be verified first [46, 47].

To the best of our knowledge, there are few studies focusing on the resource allocation of fog services in the fog environment which aims to realize the load balancing for the computing nodes in both fog and cloud.

6. Conclusion and Future Work

In recent years, IoT has been one of the most popular technologies for daily lives. With rapid development of IoT, fog computing is emerging as one of the most powerful paradigms for processing the IoT applications. In the fog environment, the IoT applications are performed by the edge computing nodes and the intermediate computing nodes in the fog, as well as the physical machines in the cloud platforms. To achieve the dynamic load balancing for each type of computing node in the fog and cloud, a dynamic resource allocation method, named DRAM, for load balancing has been developed in this paper. Firstly, a system framework in fog computing was presented and load balancing for the computing nodes is analyzed accordingly. Then, the DRAM method has been implemented based on the static resource allocation and dynamic resource scheduling for fog services. As a result, the experimental evaluations and comparison analysis were carried out to verify the validity of our proposed method.

For future work, we try to analyze the negative impact of the service migration, including the traffic for different types of computing nodes, the cost for service migration, the performance degradation for the service migration, and the data transmission cost. Furthermore, we will design a corresponding method to balance the negative effects and the positive impacts for service migration.

Key Terms and Descriptions Involved in Resource Scheduling in Fog Environment

M: The number of computing nodes
P: The set of computing nodes, $P = \{p_1, p_2, \ldots, p_M\}$
N: The number of services
S: The set of services, $S = \{s_1, s_2, \ldots, s_N\}$
p_m: The mth ($1 \le m \le M$) computing node in P
s_n: The nth ($1 \le n \le N$) service in S
c_m: The resource capacity of the mth computing node p_m
r_n: The set of resource requirements of the nth service s_n
W: The number of types of processing nodes
β_w^m: A flag to judge whether p_m is a type w of computing nodes
$ru_m(t)$: The resource utilization for p_m at time t
$RU_w(t)$: The resource utilization for the wth type computing nodes at time t
$lb_m(t)$: The load-balance variance for p_m at time t
$LB_w(t)$: The load-balance variance for the wth type computing nodes at time t
LB_w: The average load-balance variance for the wth type computing nodes.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research is supported by the National Natural Science Foundation of China under Grants nos. 61702277, 61672276, 61772283, 61402167, and 61672290, the Key Research and Development Project of Jiangsu Province under Grants nos.

BE2015154 and BE2016120, and the Natural Science Foundation of Jiangsu Province (Grant no. BK20171458). Besides, this work is also supported by the Startup Foundation for Introducing Talent of NUIST, the Open Project from State Key Laboratory for Novel Software Technology, Nanjing University, under Grant no. KFKT2017B04, the Priority Academic Program Development of Jiangsu Higher Education Institutions (PAPD) fund, Jiangsu Collaborative Innovation Center on Atmospheric Environment and Equipment Technology (CICAEET), and the project "Six Talent Peaks Project in Jiangsu Province" under Grant no. XYDXXJS-040. Special thanks are due to Dou Ruihan, Nanjing Jinling High School, Nanjing, China, for his intelligent contribution to our algorithm discussion and experiment development.

References

[1] Y. Kong, M. Zhang, and D. Ye, "A belief propagation-based method for task allocation in open and dynamic cloud environments," *Knowledge-Based Systems*, vol. 115, pp. 123–132, 2017.

[2] S. Sarkar, S. Chatterjee, and S. Misra, "Assessment of the Suitability of Fog Computing in the Context of Internet of Things," *IEEE Transactions on Cloud Computing*, vol. 6, no. 1, pp. 46–59, 2015.

[3] 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.

[4] E. Luo, M. Z. Bhuiyan, G. Wang, M. A. Rahman, J. Wu, and M. Atiquzzaman, "PrivacyProtector: Privacy-Protected Patient Data Collection in IoT-Based Healthcare Systems," *IEEE Communications Magazine*, vol. 56, no. 2, pp. 163–168, 2018.

[5] P. Li, S. Zhao, and R. Zhang, "A cluster analysis selection strategy for supersaturated designs," *Computational Statistics & Data Analysis*, vol. 54, no. 6, pp. 1605–1612, 2010.

[6] G.-L. Tian, M. Wang, and L. Song, "Variable selection in the high-dimensional continuous generalized linear model with current status data," *Journal of Applied Statistics*, vol. 41, no. 3, pp. 467–483, 2014.

[7] 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.

[8] 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.

[9] X. L. 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, Article ID 2376484, 15 pages, 2017.

[10] X. Xu, W. Dou, X. Zhang, C. Hu, and J. Chen, "A traffic hotline discovery method over cloud of things using big taxi GPS data," *Software: Practice and Experience*, vol. 47, no. 3, pp. 361–377, 2017.

[11] F. Bonomi, R. Milito, P. Natarajan, and J. Zhu, "Fog computing: A platform for internet of things and analytics," in *Big Data and Internet of Things: A Roadmap for Smart Environments*, vol. 546 of *Studies in Computational Intelligence*, pp. 169–186, 2014.

[12] 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.

[13] L. Yu, L. Chen, Z. Cai, H. Shen, Y. Liang, and Y. Pan, "Stochastic Load Balancing for Virtual Resource Management in Datacenters," *IEEE Transactions on Cloud Computing*, pp. 1–14, 2016.

[14] Y. Sahu, R. K. Pateriya, and R. K. Gupta, "Cloud server optimization with load balancing and green computing techniques using dynamic compare and balance algorithm," in *Proceedings of the 5th International Conference on Computational Intelligence and Communication Networks, CICN 2013*, pp. 527–531, India, September 2013.

[15] G. Soni and M. Kalra, "A novel approach for load balancing in cloud data center," in *Proceedings of the 2014 4th IEEE International Advance Computing Conference, IACC 2014*, pp. 807–812, India, February 2014.

[16] X. Xu, W. Dou, X. Zhang, and J. Chen, "EnReal: An Energy-Aware Resource Allocation Method for Scientific Workflow Executions in Cloud Environment," *IEEE Transactions on Cloud Computing*, vol. 4, no. 2, pp. 166–179, 2016.

[17] S. Li and Y. Zhang, "On-line scheduling on parallel machines to minimize the makespan," *Journal of Systems Science & Complexity*, vol. 29, no. 2, pp. 472–477, 2016.

[18] G. Wang, X. X. Huang, and J. Zhang, "Levitin-Polyak well-posedness in generalized equilibrium problems with functional constraints," *Pacific Journal of Optimization. An International Journal*, vol. 6, no. 2, pp. 441–453, 2010.

[19] B. Qu and J. Zhao, "Methods for solving generalized Nash equilibrium," *Journal of Applied Mathematics*, vol. 2013, Article ID 762165, 2013.

[20] 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.

[21] 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.

[22] A. V. Dastjerdi and R. Buyya, "Fog Computing: Helping the Internet of Things Realize Its Potential," *The Computer Journal*, vol. 49, no. 8, Article ID 7543455, pp. 112–116, 2016.

[23] J. Shen, T. Zhou, D. He, Y. Zhang, X. Sun, and Y. Xiang, "Block design-based key agreement for group data sharing in cloud computing," *IEEE Transactions on Dependable and Secure Computing*, vol. PP, no. 99, 2017.

[24] Z. Xia, X. Wang, L. Zhang, Z. Qin, X. Sun, and K. Ren, "A privacy-preserving and copy-deterrence content-based image retrieval scheme in cloud computing," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 11, pp. 2594–2608, 2016.

[25] J. Shen, D. Liu, J. Shen, Q. Liu, and X. Sun, "A secure cloud-assisted urban data sharing framework for ubiquitous-cities," *Pervasive and Mobile Computing*, vol. 41, pp. 219–230, 2017.

[26] Z. Fu, X. Wu, C. Guan, X. Sun, and K. Ren, "Toward efficient multi-keyword fuzzy search over encrypted outsourced data with accuracy improvement," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 12, pp. 2706–2716, 2016.

[27] S. Kitanov and T. Janevski, "Energy efficiency of Fog Computing and Networking services in 5G networks," in *Proceedings of*

the 17th IEEE International Conference on Smart Technologies, EUROCON 2017, pp. 491–494, Macedonia, July 2017.

[28] O. Akrivopoulos, I. Chatzigiannakis, C. Tselios, and A. Antoniou, "On the Deployment of Healthcare Applications over Fog Computing Infrastructure," in *Proceedings of the 41st IEEE Annual Computer Software and Applications Conference Workshops, COMPSAC 2017*, pp. 288–293, Italy, July 2017.

[29] Y. Arfat, M. Aqib, R. Mehmood et al., "Enabling Smarter Societies through Mobile Big Data Fogs and Clouds," in *Proceedings of the International Workshop on Smart Cities Systems Engineering (SCE 2017)*, vol. 109, pp. 1128–1133, Procedia Computer Science, Portugal, May 2017.

[30] M. Taneja and A. Davy, "Resource Aware Placement of Data Analytics Platform in Fog Computing," in *Proceedings of the 2nd International Conference on Cloud Forward: From Distributed to Complete Computing, CF 2016*, pp. 153–156, Procedia Computer Science, Spain, October 2016.

[31] N. Fernando, S. W. Loke, and W. Rahayu, "Mobile cloud computing: a survey," *Future Generation Computer Systems*, vol. 29, no. 1, pp. 84–106, 2013.

[32] S. S. Manvi and G. Krishna Shyam, "Resource management for Infrastructure as a Service (IaaS) in cloud computing: a survey," *Journal of Network and Computer Applications*, vol. 41, no. 1, pp. 424–440, 2014.

[33] Y. Ren, J. Shen, D. Liu, J. Wang, and J.-U. Kim, "Evidential quality preserving of electronic record in cloud storage," *Journal of Internet Technology*, vol. 17, no. 6, pp. 1125–1132, 2016.

[34] Y. Chen, C. Hao, W. Wu, and E. Wu, "Robust dense reconstruction by range merging based on confidence estimation," *Science China Information Sciences*, vol. 59, no. 9, Article ID 092103, pp. 1–11, 2016.

[35] T. Ma, Y. Zhang, J. Cao, J. Shen, M. Tang, and Y. Tian, "Abdullah Al-Dhelaan, Mznah Al-Rodhaan, KDVEM: a k-degree anonymity with Vertex and Edge Modification algorithm," *Computing*, vol. 70, no. 6, pp. 1336–1344, 2015.

[36] L. Mashayekhy, M. M. Nejad, D. Grosu, and A. V. Vasilakos, "An online mechanism for resource allocation and pricing in clouds," *Institute of Electrical and Electronics Engineers. Transactions on Computers*, vol. 65, no. 4, pp. 1172–1184, 2016.

[37] 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.

[38] A. A. Alsaffar, H. P. Pham, C.-S. Hong, E.-N. Huh, and M. Aazam, "An Architecture of IoT Service Delegation and Resource Allocation Based on Collaboration between Fog and Cloud Computing," *Mobile Information Systems*, vol. 2016, Article ID 6123234, pp. 1–15, 2016.

[39] H. Xiang, M. Peng, Y. Cheng, and H.-H. Chen, "Joint mode selection and resource allocation for downlink fog radio access networks supported D2D," in *Proceedings of the 11th EAI International Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness, QSHINE 2015*, pp. 177–182, Taiwan, August 2015.

[40] S. Banerjee and J. P. Hecker, "A Multi-agent System Approach to Load-Balancing and Resource Allocation for Distributed Computing," in *First Complex Systems Digital Campus World E-Conference*, pp. 393–408, 2017.

[41] Y. Govindaraju and H. Duran-Limon, "A QoS and energy aware load balancing and resource allocation framework for iaas cloud providers," in *Proceedings of the 9th IEEE/ACM International Conference on Utility and Cloud Computing, UCC 2016*, pp. 410–415, China, December 2016.

[42] Y. Yuan, H. Xu, B. Wang, and X. Yao, "A new dominance relation-based evolutionary algorithm for many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 1, pp. 16–37, 2016.

[43] V. Jeyakrishnan and P. Sengottuvelan, "A Hybrid Strategy for Resource Allocation and Load Balancing in Virtualized Data Centers Using BSO Algorithms," *Wireless Personal Communications*, vol. 94, no. 4, pp. 2363–2375, 2017.

[44] H. Wu, Y. Ren, and F. Hu, "Continuous dependence property of BSDE with constraints," *Applied Mathematics Letters*, vol. 45, pp. 41–46, 2015.

[45] Y. Wang, X. Sun, and F. Meng, "On the conditional and partial trade credit policy with capital constraints: a Stackelberg model," *Applied Mathematical Modelling: Simulation and Computation for Engineering and Environmental Systems*, vol. 40, no. 1, pp. 1–18, 2016.

[46] 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.

[47] C. Wang, C. Ma, and J. Zhou, "A new class of exact penalty functions and penalty algorithms," *Journal of Global Optimization*, vol. 58, no. 1, pp. 51–73, 2014.

Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 3075849, 8 pages
https://doi.org/10.1155/2018/3075849

Research Article

Privacy-Aware Multidimensional Mobile Service Quality Prediction and Recommendation in Distributed Fog Environment

Wenwen Gong,[1] Lianyong Qi ,[1,2] and Yanwei Xu[1]

[1]*School of Information Science and Engineering, Qufu Normal University, Qufu, China*
[2]*State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China*

Correspondence should be addressed to Lianyong Qi; lianyongqi@gmail.com

Received 5 February 2018; Accepted 20 March 2018; Published 24 April 2018

Academic Editor: Deepak Puthal

With the ever-increasing popularity of mobile computing technology, a wide range of computational resources or services (e.g., movies, food, and places of interest) are migrating to the mobile infrastructure or devices (e.g., mobile phones, PDA, and smart watches), imposing heavy burdens on the service selection decisions of users. In this situation, service recommendation has become one of the promising ways to alleviate such burdens. In general, the service usage data used to make service recommendation are produced by various mobile devices and collected by distributed edge platforms, which leads to potential leakage of user privacy during the subsequent cross-platform data collaboration and service recommendation process. Locality-Sensitive Hashing (LSH) technique has recently been introduced to realize the privacy-preserving distributed service recommendation. However, existing LSH-based recommendation approaches often consider only one quality dimension of services, without considering the multidimensional recommendation scenarios that are more complex but more common. In view of this drawback, we improve the traditional LSH and put forward a novel LSH-based service recommendation approach named $\text{SerRec}_{\text{multi-qos}}$, to protect users' privacy over multiple quality dimensions during the distributed mobile service recommendation process.

1. Introduction

With the advent of mobile computing age, an increasing number of web services are migrating to the mobile infrastructure or devices (e.g., mobile phones, PDA, and wearable devices), producing a wide range of mobile services that cater for the mobile environment, such as mobile payment and mobile shopping [1–3]. This rapid growth of number of mobile services, on one hand, provides more service candidates that can satisfy the users' functional or nonfunctional requirements and, on the other hand, places a heavy burden on the users' service selection decisions as selecting an appropriate mobile service is really a tiresome and time-consuming job [4, 5]. In this situation, various recommendation techniques are proposed to alleviate such a burden on users, such as the well-known collaborative filtering (CF). Typically, through analyzing the historical service usage data generated from past service invocations, CF can find out the similar friends of a target user and then make appropriate service recommendation based on the taste of the found similar friends, as they often hold the same or similar preferences.

However, in the mobile environment, the historical service usage data generated from mobile terminals are often collected and stored in various fog platforms, instead of being transferred to the remote cloud platform directly, due to the big volume of data and heavy transmission cost. In this situation, the historical service usage data used to make service recommendation are not centralized, but distributed, which raises a new topic of distributed service recommendation where user privacy may be exposed [6, 7]. Due to privacy concerns, a fog platform is often reluctant to share its inner data to other platforms, which significantly impede the cross-platform data collaboration and subsequent service recommendation. Therefore, how to integrate or fuse these distributed service usage data across different fog platforms while guaranteeing user privacy has become a challenging task that needs further investigation.

In view of this challenge, in our previous work [8–11], Locality-Sensitive Hashing (LSH) technique is introduced into distributed service recommendation so as to protect the private information of users. However, only one QoS (quality of service) dimension (e.g., response time) of mobile services is considered in the previous work, while, in this paper, we further extend our work by considering multiple quality dimensions simultaneously and further put forward a novel service recommendation approach that can protect user privacy over multiple QoS dimensions, named $\text{SerRec}_{\text{multi-qos}}$. Our proposed $\text{SerRec}_{\text{multi-qos}}$ approach can significantly extend the applicability of traditional LSH-based service recommendation.

In general, our contributions are twofold.

(1) We improve the traditional LSH-based service recommendation approach by considering multiple QoS dimensions, instead of only one QoS dimension, so that the applicability of LSH in recommendation domain can be enlarged significantly.

(2) A wide range of experiments are deployed on a real distributed service quality dataset, that is, *WS-DREAM*, to validate the feasibility of our proposed $\text{SerRec}_{\text{multi-qos}}$ approach. Experiment results show that our proposal outperforms the other state-of-the-art approaches in terms of efficiency and accuracy while guaranteeing user privacy.

The remainder of this paper is organized as follows. In Section 2, we formulate the multiple QoS-aware distributed service recommendation approach and demonstrate the motivation of our paper. In Section 3, we briefly introduce the main idea of LSH technique and then propose a novel multiple QoS-aware distributed service recommendation approach, that is, $\text{SerRec}_{\text{multi-qos}}$. A set of experiments are conducted in Section 4 to validate the feasibility and advantages of our proposal. Related work and comparison analyses are presented in Section 5. Finally, in Section 6, we summarize the paper and point out the future research directions.

2. Formulation and Motivation

In Section 2.1, we formulate the multiple QoS-aware distributed service recommendation approach; afterwards, in Section 2.2, we present an intuitive example to demonstrate the motivation of our paper.

2.1. Problem Formulation. The distributed mobile service recommendation problem with multiple QoS dimensions can be specified by a five-tuple SerRer (FOG, MS, U, u_{target}, Q), where

(1) $\text{FOG} = \{\text{fog}_1, \ldots, \text{fog}_z\}$ denotes the fog platform set in which each platform consists of a set of mobile services;

(2) $\text{MS} = \{\text{ms}_1, \ldots, \text{ms}_n\}$ represents the set of mobile services that are ready to be recommended to potential users;

(3) $U = \{u_1, \ldots, u_m\}$ denotes the set of users in the user-service invocation network;

(4) u_{target} is a target user to whom the recommender system intends to recommend services. Here, $u_{\text{target}} \in U$ holds;

(5) $Q = \{q_1, \ldots, q_k\}$ represents the set of QoS dimensions of mobile services.

With the above formulation, we can specify the multi-QoS-aware mobile service recommendation problems in the distributed fog environment as follows: according to users ($\in U$)' historical QoS data ($\in Q$) over mobile services ($\in$MS) in different fog platforms ($\in$FOG), predict the future quality of mobile services that are never invoked by u_{target} and recommend the quality-optimal mobile service to u_{target}. As a user's historical service quality data are often distributed in different fog platforms, the service recommendation process often calls for a cross-platform data collaboration process during which the private information of users (e.g., the historical QoS data observed by a user) may be exposed to the outside. In this paper, we will investigate how to make accurate service recommendations while guaranteeing users' privacy over multiple QoS dimensions.

2.2. Motivation. In this subsection, an intuitive example is presented in Figure 1 to motivate our paper. In Figure 1, there are two users, that is, u_{target} and u_1, whose ever-invoked service quality data are located in *Microsoft* and *SAP* fog platforms, respectively; each candidate mobile service has k QoS dimensions, that is, $q_1, \ldots, q_k$. Then according to collaborative filtering theory, if a recommender system intends to recommend new services to u_{target}, it is necessary to calculate the user similarity (e.g., the well-known PCC distance) between u_{target} and u_1 by fusing or integrating the historical QoS data across both *Microsoft* and *SAP* fog platforms. However, such a cross-platform collaboration process may reveal the private information of u_{target} and u_1, which significantly impedes the recommendation process for u_{target}.

In view of this challenge, we introduce LSH technique into the cross-platform and privacy-preserving service recommendation process and modify it to adapt the recommendation scenario where multiple QoS dimensions of mobile services are involved. We will introduce our proposal in detail in the next section.

3. Privacy-Aware Multidimensional Mobile Service Recommendation

In this section, we first briefly introduce the LSH technique recruited in our proposal; afterwards, we modify traditional LSH to adapt the multi-QoS-aware service recommendation scenario so as to tackle the privacy-preservation problem raised in the distributed fog environment.

3.1. Locality-Sensitive Hashing (LSH). Locality-Sensitive Hashing, an fast lookup technique of approximate nearest neighbor (ANN) search for massive and high dimensional data, was put forward by Alex Andoni in 1999 [12]. The privileged advantage of LSH in ANN search is that it possesses a good characteristic of "similarity-keeping." Namely, two

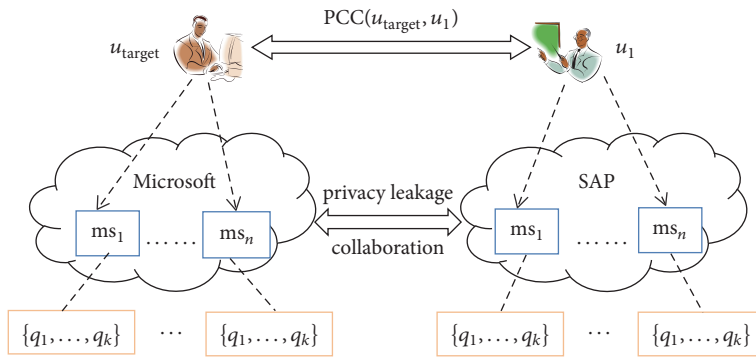


Figure 1: Multi-QoS-aware distributed mobile service recommendation.

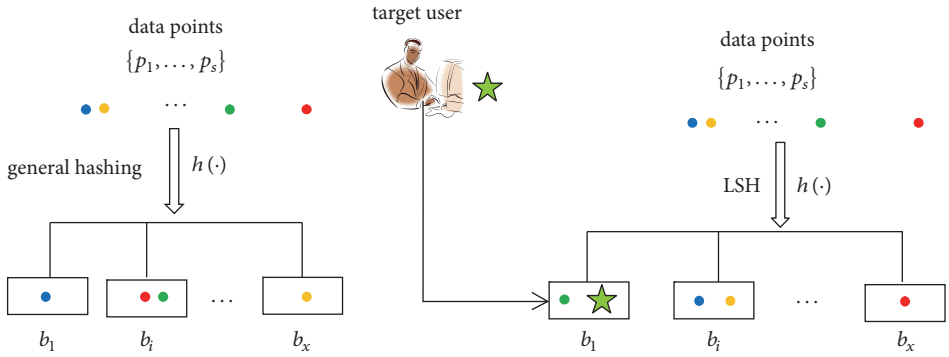


Figure 2: Comparison of general hashing and Locality-Sensitive Hashing (LSH).

points that are close to each other in the raw data space would be projected into the same bucket of a hash table after LSH process, with high probability; on the contrary, for two points which are far away from each other in the raw data space, they would be projected into different buckets after LSH process, with high probability. This way, we can utilize the hash values with little privacy to realize the ANN search, so that the private information can be protected during the search process.

The main idea behind LSH can be illustrated more intuitively by Figure 2 where the general hashing and LSH are compared. In Figure 2(a), $h(\cdot)$ denotes a hash function; through general hashing, the raw data points $p_1, \ldots, p_s$ are projected into corresponding buckets of a hash table, that is, $b_1, \ldots, b_x$, without following the "similarity-keeping" rule. In contrast, the hashing projection process in Figure 2(b) follows the "similarity-keeping" rule; namely, the blue point and yellow point that are close to each other are projected into the same bucket (i.e., b_i in Figure 2(b)) with high probability, while the green point and red point that are far away from each other are projected into buckets b_1 and b_x, respectively. Thus if a target user (marked with green five-star shape) is projected into bucket b_1, then we can conclude that the target user is similar to the green point with high probability. This is the main idea of ANN search based on LSH. In this paper, we use this search idea to find the similar friends of a target user.

3.2. Multi-QoS-Aware and Privacy-Preserving Distributed Service Recommendation Approach Based on LSH: $\text{SerRec}_{\text{multi-qos}}$. In this subsection, we introduce our proposed multidimensional service quality prediction and recommendation approach, that is, $\text{SerRec}_{\text{multi-qos}}$. Concretely, $\text{SerRec}_{\text{multi-qos}}$ approach consists of the three steps in Algorithm 1. Here, u and ms denote a user in set U and a mobile service in set MS, respectively; $h(\cdot)$ denotes a LSH function; L and r denote the number of hash tables and the number of hash functions in each hash table, respectively.

Step 1 (build multidimensional user indices offline). In this step, we build the index for each user $u \in U$ based on u's historical QoS data over dimensions $q_1, \ldots, q_k$. Concretely, u's historical QoS data can be specified by matrix u_{k*n} in Figure 3, where n denotes the number of candidate mobile services. Then the index for user u can be calculated by $H(u)$ in Figure 3 where r denotes the number of hash functions in each hash table. Here, $h_{i,j}(u)$ $(1 \le i \le k,\ 1 \le j \le r)$ denotes a Boolean hash value of user u, which can be calculated by the hash function $h_{i,j}(\cdot)$ in (1)-(2). In (2), X^T denotes the transpose operation of vector X; symbol "$\circ$" denotes the dot product of two vectors; $v_{y,j}$ $(1 \le y \le n)$ is a random value in range $[-1, 1]$ [13] (according to the LSH theory, each kind of distance metric is corresponding to a specific LSH function; as Pearson Correlation Coefficient (PCC) is often taken as the similarity metric in recommender systems, we adopt the

Step 1 (build multi-dimensional user indices offline). For each $u \in U$, according to his/her observed QoS data over dimensions $q_1, \ldots, q_k$, build his/her index (denoted by $H(u)$) offline based on a LSH function family $H(\cdot) = \{h_1(\cdot), h_2(\cdot), \ldots, h_r(\cdot)\}$. Repeat the above index building process L times so as to obtain L hash tables.
Step 2 (search for similar friends of u_{target} online). For each $u \in U$, compare $H(u)$ and $H(u_{\text{target}})$ online; if $H(u) = H(u_{\text{target}})$ holds in any of the L hash tables, then u can be regarded as similar with u_{target} and put into set Friend(u_{target}).
Step 3 (service recommendation). For each ms ($\in$MS) never invoked by u_{target}, predict its quality over dimensions $q_1, \ldots, q_k$ by u_{target}, denoted by $Q_{\text{target}} = (\text{ms} \cdot q_{1(\text{target})}, \ldots, \text{ms} \cdot q_{k(\text{target})})$, based on set Friend($u_{\text{target}}$) obtained in Step 2. Finally, return the quality-optimal mobile service to u_{target}.

Algorithm 1: Three steps of $\text{SerRec}_{\text{multi-qos}}$ approach.

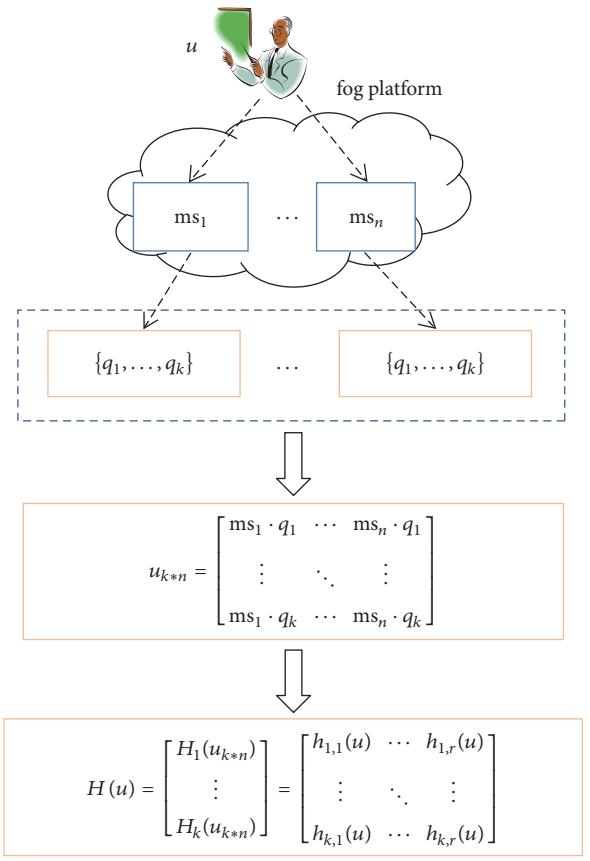


Figure 3: User indices building process.

LSH function corresponding to PCC distance here; i.e., $v_{y,j}$ is randomly selected from $[-1, 1]$). Thus we can get a Boolean hash value $h_{i,j}(u)$ through the following:

$$h_{i,j}(u) = \begin{cases} 1 & \text{if } \omega_{i,j} > 0 \\ 0 & \text{if } \omega_{i,j} \le 0, \end{cases} \tag{1}$$

$$\omega_{i,j} = (\text{ms}_1 \cdot q_i, \ldots, \text{ms}_n \cdot q_i) \circ (v_{1,j}, \ldots, v_{n,j})^T. \tag{2}$$

Repeat the above process until the user index $H(u)$ in Figure 3 is obtained, after which a hash table (denoted by H_Table) is obtained. Then we repeat the hash table building process offline so as to derive L hash tables, that is, $H\_\text{Table}_1, \ldots, H\_\text{Table}_L$. The advantages of hash tables are twofold: first, a hash table only stores or records the less sensitive hash values of users, without revealing the private information of users; second, hash tables can be built offline before a service recommendation request arrives, through which the efficiency of further service recommendation can be improved significantly.

Step 2 (search for similar friends of u_{target} online). In Step 1, we have derived the index for each user $u \in U$, that is, $H(u)$. Likewise, in this step, we can obtain the index for the target user, that is, $H(u_{\text{target}})$. Next, if $H(u) = H(u_{\text{target}})$ holds in any of the L hash tables, then we can conclude that u is a similar friend of u_{target} with high probability according to the LSH theory; so we put u into a new set Friend(u_{target}) which contains all the similar friends of the target user.

Step 3 (service recommendation). In this step, we make mobile service recommendation to the target user based on u_{target}'s friend set Friend (u_{target}) derived in Step 2. Concretely, for each mobile service (denoted by ms) never invoked by u_{target} before, we predict its quality over dimension q_i ($1 \le i \le k$) by u_{target}, denoted by $\text{ms} \cdot q_{i(\text{target})}$, based on (3). Here, $\text{ms} \cdot q_{i(u)}$ denotes ms's quality over dimension q_i observed by user u. Next, we predict ms's comprehensive quality (denoted by $\text{ms} \cdot q_{(\text{target})}$) by considering all the k quality dimensions $q_1, \ldots, q_k$. Concretely, $\text{ms} \cdot q_{(\text{target})}$ can be calculated by (4), where Norm$(\cdot)$ denotes the normalization operation for eliminating the interference of different quality units. Finally, we select the candidate mobile service with the optimal predicted quality and return it to the target user, so as to finish the whole distributed mobile service recommendation process.

$$\text{ms} \cdot q_{i(\text{target})} = \sum_{u \in \text{Friend}(u_{\text{target}})} \frac{\text{ms} \cdot q_{i(u)}}{\left|\text{Friend}\left(u_{\text{target}}\right)\right|}, \tag{3}$$

$$\text{ms} \cdot q_{(\text{target})} = \sum_{i=1}^{k} \text{Norm}\left(\text{ms} \cdot q_{i(\text{target})}\right). \tag{4}$$

4. Experiments

In this section, a set of experiments are designed and tested to validate the feasibility of our proposed service recommendation approach, that is, $\text{SerRec}_{\text{multi-qos}}$.

4.1. Experiment Configurations. Our experiments are deployed on a real-world distributed service quality dataset *WS-DREAM* [14] which collects real-world QoS data of 5825 web services observed by 339 users. To simulate the distributed service recommendation scenario, we take each country that hosts a set of services as an independent fog platform. The service QoS matrix in *WS-DREAM* is very dense, so we randomly remove 90% entries from the matrix to simulate the missing QoS prediction and service recommendation requirements. Moreover, two quality dimensions of services are considered, that is, *response time* and another one whose QoS values are randomly generated according to the range of *response time* values (although *WS-DREAM* provides QoS data of two quality dimensions, *response time* and *throughput*, the QoS data distribution of *throughput* is very skew which makes it not suitable for LSH-based ANN search very much according to the LSH theory).

The following two evaluation criteria are tested in the experiments:

(1) *time cost*: time consumed for generating the final recommended service.

(2) *MAE* (Mean Absolute Error, the smaller the better): average difference between the predicted service quality and the real service quality of recommended service.

In order to validate the feasibility and advantages of our proposed $\text{SerRec}_{\text{multi-qos}}$ approach, we compare our proposal with another three state-of-the-art approaches: *UPCC* [5], *P-UIPCC* [6], and *PPICF* [7]. Concretely, *UPCC* is a benchmark collaborative filtering recommendation approach, while *P-UIPCC* and *PPICF* utilize data perturbation or division technique to protect the private information of users.

The experiments were conducted on a Lenovo laptop with 2.40 GHz processors and 12.0 GB RAM. The machine runs Windows 10, JAVA 8, and MySQL 5.7. Each experiment was performed 10 times, and the average experimental results are reported.

4.2. Experiment Results. Concretely, four profiles are tested and compared. Here, m and n denote the number of users and number of services, respectively; L and r denote the number of hash tables and number of hash functions in each hash table, respectively.

Profile 1 (recommendation accuracy comparison of four approaches). In this profile, we test the recommendation accuracy of our proposed $\text{SerRec}_{\text{multi-qos}}$ approach and compare it with the other three approaches. The experiment parameters are set as follows: m is varied from 100 to 300, n is varied from 1000 to 3000, $L = 10$, and $r = 4$. Concrete experiment results are shown in Figure 4. As Figure 4 indicates, the recommendation accuracy of our proposal is close to that of the benchmark approach *UPCC*. Besides, our $\text{SerRec}_{\text{multi-qos}}$ approach outperforms *PPICF* and *P-UIPCC* in terms of recommendation accuracy; this is because in our proposal, only the "most similar" friends of a target user can be found and recruited to make service recommendation, while, in both *PPICF* and *P-UIPCC*, additional perturbation or division operation is applied on the real service QoS data so as to protect user privacy, which reduces the recommendation accuracy to some extent.

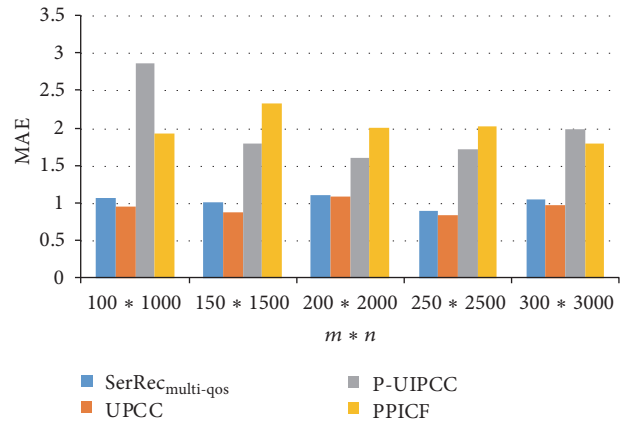


Figure 4: Recommendation accuracy of four approaches with respect to m and n.

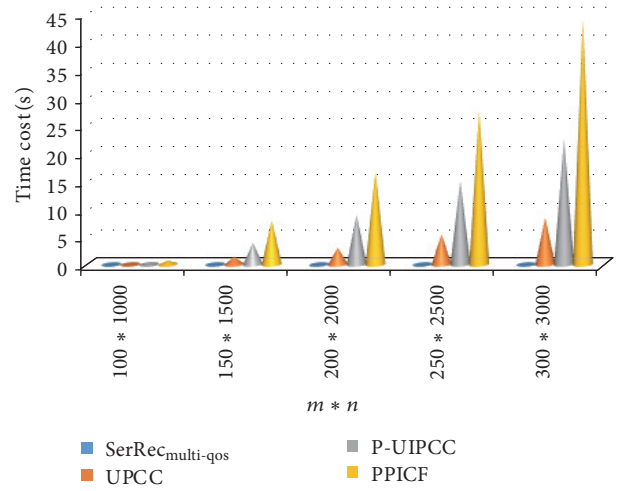


Figure 5: Recommendation efficiency of four approaches with respect to m and n.

Profile 2 (recommendation efficiency comparison of four approaches). Recommendation efficiency is an important metric to evaluate the performance of a recommender system. So in this profile, we test and compare the recommendation efficiency and scalability of four approaches. Experiment parameters are set as follows: m is varied from 100 to 300, n is varied from 1000 to 3000, $L = 10$, and $r = 4$. Concrete experiment results are presented in Figure 5. As Figure 5 shows, the time cost of our $\text{SerRec}_{\text{multi-qos}}$ approach is rather low and outperforms the other three approaches significantly; this is because most jobs (e.g., user indices building) in our approach can be done offline before a service recommendation request arrives, while the rest job (e.g., online similar friends search) can be done quickly as its time complexity is about $O(1)$ [13]. The low time complexity means

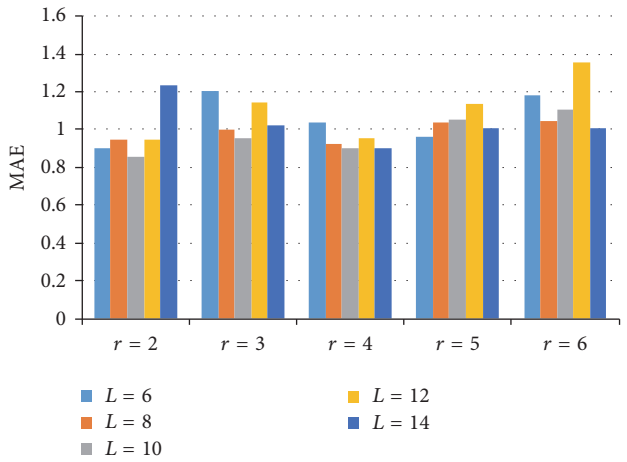


Figure 6: Recommendation accuracy of $\text{SerRec}_{\text{multi-qos}}$ with respect to L and r.

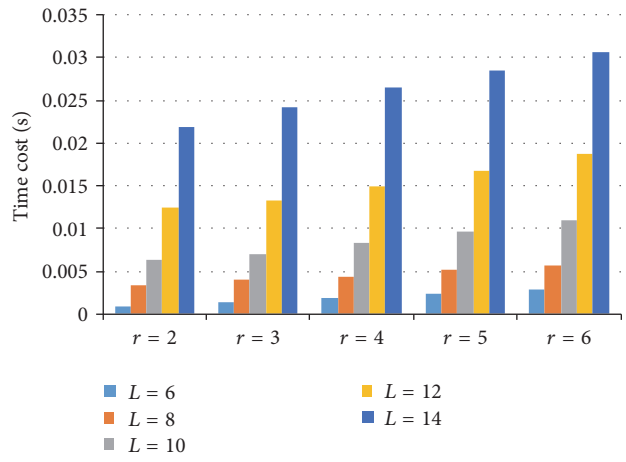


Figure 7: Recommendation efficiency of $\text{SerRec}_{\text{multi-qos}}$ with respect to L and r.

that our recommendation approach can satisfy the quick response requirements from mobile users.

Profile 3 (recommendation accuracy of $\text{SerRec}_{\text{multi-qos}}$ with respect to L and r). LSH is essentially a probability-based ANN search technique; therefore, the recommendation accuracy of our $\text{SerRec}_{\text{multi-qos}}$ approach is correlated with the number of hash tables (i.e., L) and the number of hash functions in each hash table (i.e., r). In view of this, in this profile, we test the relationship between recommendation accuracy and parameter combination (L, r). Experiment parameters are set as follows: $m = 200$, $n = 2000$, L is varied from 6 to 14, and r is varied from 2 to 6. Concrete experiment results are illustrated in Figure 6.

As Figure 6 indicates, the recommendation accuracy of our proposal does not show a very regular variation tendency with respect to L or r, and this is because both L and r can affect the recommendation accuracy simultaneously. Specifically, our approach achieves the highest recommendation accuracy (i.e., lowest *MAE* value) when parameter combination $(L, r) = (10, 2)$ holds. However, we argue that our LSH-based recommendation approach is very sensitive to the experiment data; namely, (10, 2) is not always the optimal parameter combination when different experiment data are adopted.

Profile 4 (recommendation efficiency of $\text{SerRec}_{\text{multi-qos}}$ with respect to L and r). In this profile, we test the time cost of $\text{SerRec}_{\text{multi-qos}}$ approach with respect to L and r. Experiment parameters are set as follows: $m = 200$, $n = 2000$, L is varied from 6 to 14, and r is varied from 2 to 6. Concrete experiment results are illustrated in Figure 7. As Figure 7 shows, the time cost of our proposal increases with both L and r, because more comparison operations are incurred during the online friend search process when L and r grow. However, as Figure 7 indicates, the time cost of our proposal is rather low (<0.3 s) in most cases, which means that our recommendation approach can satisfy the users' quick response requirements regardless of the parameter values of L and r.

5. Related Work and Further Discussions

The distribution of service QoS data used to make service recommendation has raised the problem of distributed service recommendation and privacy-preservation. Many researchers have investigated this problem and put forward their respective privacy-preservation resolutions. In [15], the authors suggest that a user only releases partial historical QoS data to the public so as to ensure that the majority of QoS data are still secure; however, the released partial QoS data can still reveal partial privacy of the user. In [16], the authors propose a K-anonymous approach to protect the sensitive user information; however, the data availability after K-anonymity process is often reduced considerably, which decreases the service recommendation accuracy to some extent. In [17], a homomorphic encryption-based approach is proposed to make e-commerce recommendation while guaranteeing private-preservation; however, similar to other encryption-based approaches, the proposed homomorphic encryption-based privacy-preservation approach in [17] is often heavy-weight and hence cannot satisfy the light-weight service recommendation requirements from mobile users. In [6], data perturbation technique is recruited to protect the real service QoS data observed by users, so that only the obfuscated QoS data can be released to the public; however, as the QoS data used to make service recommendation has been obfuscated, the service recommendation accuracy is reduced accordingly. In [7], data division technique is adopted to protect user privacy. Concretely, each piece of QoS data is divided into several QoS segments with little privacy and then the QoS segments are employed to calculate user similarity and make service recommendations. However, this approach can still reveal partial privacy of users, such as the service intersection commonly invoked by two users.

In our previous work [8–11], LSH technique has been recruited to protect the private information of users while performing distributed service recommendation. However, only one quality dimension (e.g., *response time*) of services is considered in the existing research, while, in the actual

situations, multiple dimensions are more common [18–27]. In view of this drawback, in this paper, we extend our previous work by considering more quality dimensions and put forward a multi-QoS-aware mobile service recommendation approach based on LSH. Through experiments deployed on a real distributed service quality dataset *WS-DREAM*, we validate the feasibility of our proposal in terms of recommendation accuracy and efficiency while guaranteeing privacy-preservation.

However, there are still several shortcomings in our approach. First, we only consider the quality dimensions whose values are real and continuous, for example, *response time*, while neglecting some other dimensions whose values are discrete [28–38], binary [39], and fuzzy [40–42]. So in the future, we will investigate how to integrate the quality dimensions with different data types. Besides, in the multidimensional applications, each dimension is often assigned a weight to indicate its importance [43–49]; therefore, we will take the weight information of different quality dimensions into consideration so as to make the recommendation decisions more reasonable.

6. Conclusions

The service QoS data generated from mobile devices are often first handled and filtered by different fog platforms, instead of sending the QoS data directly to a remote cloud platform, so as to reduce the data transmission cost from mobile devices to the cloud platform. However, such a cross-platform data distribution raises a novel problem of distributed mobile service recommendation as well as the resulting privacy leakage risks. Existing research often falls short in protecting the sensitive information of users during the distributed service recommendation process, especially when multiple QoS dimensions are involved. In view of this drawback, we put forward a multi-QoS-aware mobile service recommendation approach so as to protect user privacy in the distributed fog environment. Through a set of experiments on a real-world distributed service quality dataset *WS-DREAM*, we validate the feasibility of our proposal in terms of service recommendation accuracy and efficiency while guaranteeing privacy-preservation. In the future, we plan to improve our approach by taking into consideration the data types and weight of different quality dimensions.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This paper is partially supported by Open Project of State Key Laboratory for Novel Software Technology (no. KFKT2016B22).

References

[1] J. Schobel, R. Pryss, W. Wipp, M. Schickler, and M. Reichert, "A mobile service engine enabling complex data collection applications," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics): Preface*, vol. 9936, pp. 626–633, 2016.

[2] H. Xiang, X. Xu, H. Zheng et al., "An adaptive cloudlet placement method for mobile applications over GPS big data," in *Proceedings of the 59th IEEE Global Communications Conference, GLOBECOM 2016*, USA, December 2016.

[3] G. Skourletopoulos, C. X. Mavromoustakis, G. Mastorakis, J. M. Batalla, and J. N. Sahalos, "An evaluation of cloud-based mobile services with limited capacity: a linear approach," *Soft Computing*, vol. 21, no. 16, pp. 4523–4530, 2017.

[4] 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.

[5] J. S. Breese, D. Heckerman, and C. Kadie, "Empirical Analysis of Predictive Algorithms for Collaborative Filtering," in *Proceedings of the The Fourteenth Conference on Uncertainty in Artificial Intelligence*, pp. 43–52, 1998.

[6] J. Zhu, P. He, Z. Zheng, and M. R. Lyu, "A privacy-preserving qos prediction framework for web service recommendation," in *Proceedings of the IEEE International Conference on Web Services (ICWS '15)*, pp. 241–248, New York, NY, USA, July 2015.

[7] D. Li, C. Chen, Q. Lv et al., "An algorithm for efficient privacy-preserving item-based collaborative filtering," *Future Generation Computer Systems*, vol. 55, pp. 311–320, 2016.

[8] 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.

[9] L. Qi, H. Xiang, W. Dou, C. Yang, Y. Qin, and X. Zhang, "Privacy-preserving distributed service recommendation based on locality-sensitive hashing," in *Proceedings of the 2017 IEEE International Conference on Web Services (ICWS)*, pp. 49–56, Honolulu, Hawaii, USA, June 2017.

[10] L. Qi, W. Dou, X. Zhang, and S. Yu, "Amplified locality-sensitive hashing for privacy-preserving distributed service recommendation," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics): Preface*, vol. 10656, pp. 280–297, 2017.

[11] L. Qi, W. Dou, and X. Zhang, "Two-phase locality-sensitive hashing for privacy-preserving distributed service recommendation," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics): Preface*, vol. 10581, pp. 176–188, 2017.

[12] M. Datar, A. Gionis, P. Indyk, and R. Motwani, "Maintaining stream statistics over sliding windows," *SIAM Journal on Computing*, vol. 31, no. 6, pp. 1794–1813, 2002.

[13] I. Yannis et al., *Data Mining and Query Log Analysis for Scalable Temporal and Continuous Query Answering*, 2015, http://www.optique-project.eu/.

[14] Z. Zheng, Y. Zhang, and M. R. Lyu, "Investigating QoS of real-world web services," *IEEE Transactions on Services Computing*, vol. 7, no. 1, pp. 32–39, 2014.

[15] X. Zheng, Z. Cai, J. Li, and H. Gao, "Location-privacy-aware review publication mechanism for local business service systems," in *Proceedings of the IEEE INFOCOM 2017 - IEEE Conference on Computer Communications*, pp. 1–9, Atlanta, GA, USA, May 2017.

[16] F. Casino, J. Domingo-Ferrer, C. Patsakis, D. Puig, and A. Solanas, "A k-anonymous approach to privacy preserving collaborative filtering," *Journal of Computer and System Sciences*, vol. 81, no. 6, pp. 1000–1011, 2015.

[17] S. Sobitha Ahila and K. L. Shunmuganathan, "Role of Agent Technology in Web Usage Mining: Homomorphic Encryption Based Recommendation for E-commerce Applications," *Wireless Personal Communications*, vol. 87, no. 2, pp. 499–512, 2016.

[18] M. Wang and G.-L. Tian, "Robust group non-convex estimations for high-dimensional partially linear models," *Journal of Nonparametric Statistics*, vol. 28, no. 1, pp. 49–67, 2016.

[19] X. Wang and M. Wang, "Variable selection for high-dimensional generalized linear models with the weighted elastic-net procedure," *Journal of Applied Statistics*, vol. 43, no. 5, pp. 796–809, 2016.

[20] P. Wang and L. Zhao, "Some geometrical properties of convex level sets of minimal graph on 2-dimensional Riemannian manifolds," *Nonlinear Analysis: Theory, Methods & Applications*, vol. 130, pp. 1–17, 2016.

[21] X. Wang and M. Wang, "Adaptive group bridge estimation for high-dimensional partially linear models," *Journal of Inequalities and Applications*, vol. 2017, article no. 158, 2017.

[22] X. Wang, S. Zhao, and M. Wang, "Restricted profile estimation for partially linear models with large-dimensional covariates," *Statistics & Probability Letters*, vol. 128, pp. 71–76, 2017.

[23] H. Tian and M. Han, "Bifurcation of periodic orbits by perturbing high-dimensional piecewise smooth integrable systems," *Journal of Differential Equations*, vol. 263, no. 11, pp. 7448–7474, 2017.

[24] P. Wang and X. Wang, "The geometric properties of harmonic function on 2-dimensional Riemannian manifolds," *Nonlinear Analysis: Theory, Methods & Applications*, vol. 103, pp. 2–8, 2014.

[25] M. Wang and X. Wang, "Adaptive Lasso estimators for ultrahigh dimensional generalized linear models," *Statistics & Probability Letters*, vol. 89, no. 1, pp. 41–50, 2014.

[26] X. Wang, M. Wang, and X. Wang, "A note on the one-step estimator for ultrahigh dimensionality," *Journal of Computational and Applied Mathematics*, vol. 260, pp. 91–98, 2014.

[27] G.-L. Tian, M. Wang, and L. Song, "Variable selection in the high-dimensional continuous generalized linear model with current status data," *Journal of Applied Statistics*, vol. 41, no. 3, pp. 467–483, 2014.

[28] H. Liu and F. Meng, "Some new generalized Volterra-Fredholm type discrete fractional sum inequalities and their applications," *Journal of Inequalities and Applications*, vol. 2016, no. 1, article no. 213, 2016.

[29] P. Li and G. Ren, "Some classes of equations of discrete type with harmonic singular operator and convolution," *Applied Mathematics and Computation*, vol. 284, pp. 185–194, 2016.

[30] P. Li, "Singular integral equations of convolution type with Hilbert kernel and a discrete jump problem," *Advances in Difference Equations*, vol. 2017, no. 1, article no. 360, 2017.

[31] Y. Bai and L. Liu, "New oscillation criteria for second-order delay differential equations with mixed nonlinearities," *Discrete Dynamics in Nature and Society*, vol. 2010, Article ID 796256, 2010.

[32] Y. Wang and C. Yin, "Approximation for the ruin probabilities in a discrete time risk model with dependent risks," *Statistics & Probability Letters*, vol. 80, no. 17-18, pp. 1335–1342, 2010.

[33] Q. Feng, F. Meng, and Y. Zhang, "Generalized Gronwall-Bellman-type discrete inequalities and their applications," *Journal of Inequalities and Applications*, vol. 2011, article no. 47, 2011.

[34] P. Li, "Two classes of linear equations of discrete convolution type with harmonic singular operators," *Complex Variables and Elliptic Equations*, vol. 61, no. 1, pp. 67–75, 2016.

[35] Z. Zheng, "Invariance of deficiency indices under perturbation for discrete Hamiltonian systems," *Journal of Difference Equations and Applications*, vol. 19, no. 8, pp. 1243–1250, 2013.

[36] M. Han, X. Hou, L. Sheng, and C. Wang, "Theory of Rotated Equations and Applications to A Population Model," *Discrete & Continuous Dynamical Systems-A*, vol. 38, no. 4, pp. 2171–2185, 2018.

[37] J. Cai and H. Li, "A new sufficient condition for pancyclability of graphs," *Discrete Applied Mathematics*, vol. 162, pp. 142–148, 2014.

[38] L. L. Liu and B.-X. Zhu, "Strong q-log-convexity of the Eulerian polynomials of Coxeter groups," *Discrete Mathematics*, vol. 338, no. 12, pp. 2332–2340, 2015.

[39] Z. Bin, "Remarks on the maximum gap in binary cyclotomic polynomials," *Bulletin Mathematique De La Societe Des Sciences Mathematiques De Roumanie*, vol. 59, no. 1, pp. 109–115, 2016.

[40] L. Wang, "Intuitionistic fuzzy stability of a quadratic functional equation," *Fixed Point Theory and Applications*, vol. 2010, Article ID 107182, 2010.

[41] X. Du and Z. Zhao, "On fixed point theorems of mixed monotone operators," *Fixed Point Theory and Applications*, vol. 2011, Article ID 563136, 2011.

[42] B. Zhu, L. Liu, and Y. Wu, "Local and global existence of mild solutions for a class of nonlinear fractional reaction-diffusion equations with delay," *Applied Mathematics Letters*, vol. 61, pp. 73–79, 2016.

[43] S. Yang, Z.-A. Yao, and C.-A. Zhao, "The weight distributions of two classes of p-ary cyclic codes with few weights," *Finite Fields and Their Applications*, vol. 44, pp. 76–91, 2017.

[44] Y.-F. Wang, C.-C. Yin, and X.-S. Zhang, "Uniform estimate for the tail probabilities of randomly weighted sums," *Acta Mathematicae Applicatae Sinica*, vol. 30, no. 4, pp. 1063–1072, 2014.

[45] S. Yang, Z.-A. Yao, and C.-A. Zhao, "A class of three-weight linear codes and their complete weight enumerators," *Cryptography and Communications*, vol. 9, no. 1, pp. 133–149, 2017.

[46] J. Cai, "An implicit $\sigma 3$ type condition for heavy cycles in weighted graphs," *Ars Combinatoria*, vol. 115, pp. 211–218, 2014.

[47] S. Yang and Z.-A. Yao, "Complete weight enumerators of a family of three-weight linear codes," *Designs, Codes and Cryptography*, vol. 82, no. 3, pp. 663–674, 2017.

[48] S. Yang and Z.-A. Yao, "Complete weight enumerators of a class of linear codes," *Discrete Mathematics*, vol. 340, no. 4, pp. 729–739, 2017.

[49] S. Yang, X. Kong, and C. Tang, "A construction of linear codes and their complete weight enumerators," *Finite Fields and Their Applications*, vol. 48, pp. 196–226, 2017.

Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 1245720, 11 pages
https://doi.org/10.1155/2018/1245720

Research Article

Network Coding for Backhaul Offloading in D2D Cooperative Fog Data Networks

Ben Quinton and Neda Aboutorab

University of New South Wales, Campbell, ACT 2612, Australia

Correspondence should be addressed to Neda Aboutorab; n.aboutorab@unsw.edu.au

Received 26 January 2018; Accepted 19 March 2018; Published 22 April 2018

Academic Editor: Xuyun Zhang

Future distributed data networks are expected to be assisted by users cooperation and coding schemes. Given the explosive increase in the end-users' demand for download of the content from the servers, in this paper, the implementation of instantly decodable network coding (IDNC) is considered in full-duplex device-to-device (D2D) cooperative fog data networks. In particular, this paper is concerned with designing efficient transmission schemes to offload traffic from the expensive backhaul of network servers by employing IDNC and users cooperation. The generalized framework where users send request for multiple packets and the transmissions are subject to erasure is considered. The optimal problem formulation is presented using the stochastic shortest path (SSP) technique over the IDNC graph with induced subgraphs. However, as the optimal solution suffers from the intractability of being NP-hard, it is not suitable for real-time communications. The complexity of the problem is addressed by presenting a greedy heuristic algorithm used over the proposed graph model. The paper shows that by implementing IDNC in a full-duplex cooperative D2D network model significant reduction in the number of downloads required from the servers can be achieved, which will result in offloading of the backhaul servers and thus saving valuable servers' resources. It is also shown that the performance of the proposed heuristic algorithm is very close to the optimal solution with much lower computational complexity.

1. Introduction

With the modern advancements of wireless communications, wireless networks have seen an explosion in data traffic over the past decade [1]. This rapid demand for more data is largely attributed to video and multimedia streaming, where it is expected that three-fourths of data traffic will be consumed by video [1, 2]. To compound this further, it is expected that the next generation of wireless networks will encapsulate the new paradigm of the Internet of things (IoT). This concept moves to further integrate more and more devices into communication networks, where it is foreseen that the IoT will add a further 50 billion heterogeneous wireless devices by 2020 [1]. Consequently, this growing demand puts further pressure on data networks, where the offloading of the servers becomes an increasingly important problem.

This ever-growing demand for real-time data, where users expect to maintain their quality-of-experience, has led to much research to address the data networks backhaul problem. Multiple areas of research have shown promising methods to deal with this problem; one such option is to distribute the data closer to the users with improved redundancy [3–7]. The idea of distributing resources to the edge of a network is known as "fog" networking [8]. Fog networking is motivated by distributing the "popular" content that is demanded by the end-users closer to the edge of the network or even the end-Users. It is expected that with the proactive (i.e., without the end-users request) diffusing of the popular content from large network backhauls and distributing it in a "fog" of low-cost geographically close caches to the end-users will help to serve user download requests, in turn dramatically improving the network performance and quality-of-service [9]. Using this approach not only can the users' requests be immediately and efficiently addressed, but also the access to the backhaul could be significantly offloaded [10–12].

In addition to distributing the data, with the rapid increase in the number of wireless devices, there are more and

more devices in each others proximity. These wireless devices can form an autonomous cooperative local network in a "geographically close" region where they can communicate and share files without the need for a centralized server. For example, this scenario may occur when coworkers are using wireless devices to update files stored in the cloud (e.g., Dropbox), or when users, in the subway, are interested in watching the same trending and popular video. Under these conditions, the benefits of communicating over a local network can be utilized to not only reduce the users' download time but also offload the backhaul of the data network (i.e., minimizing the download from it).

Furthermore, network coding (NC) as an efficient approach to further assist and improve the offloading of the backhaul servers while providing a faster and more reliable service to the users has attracted interest in recent years. NC, initially introduced in [13] by Ahlswede et al., can help in offloading of the backhaul servers in the considered distributed cooperative data network scenario by maximizing the number of served users in one transmission through users cooperation as well as coded servers transmissions, thus maximizing the backhaul offloading. In general, NC can be defined as a communication technique where information is encoded at the transmitter or the intermediate nodes in the network and then decoded by the users. NC has shown the ability to improve throughput, reduce delays, and provide more robust networks [14–18].

Multiple areas of study have focused on various types of network coding, where this paper will focus on opportunistic network coding (ONC) [19], in particular instantly decodable network coding (IDNC) [14]. This technique has recently gained much attention due to its instant decodability (as the name suggests) by using a simple XOR operation that results in reducing the computational complexity of the decoding at the end-users. It also provides a significant benefit to real-time communications, where studies in [4, 14, 20, 21] show through heuristic algorithms utilizing IDNC results in significant performance improvements over uncoded transmissions in both centralized point-to-multipoint (PMP) and decentralized network settings.

Although much work has focused on implementing IDNC in various network models, the studies have focused on centralized PMP and noncooperative scenarios. Only limited results exist on implementing IDNC in a cooperative setting where there is a focus on reducing the number of downloads required from the backhaul servers. For instance, the authors' previous work [22] considers the problem of designing network coding schemes for cooperative fog networks with a somewhat limited set of numerical analysis and under simplified assumptions such as single file request from the users as well as the case with no channel erasures. A *generalized* network coded cooperative D2D-enabled fog architecture under more realistic assumptions (i.e., multiple file requests from the users over multiple time epochs as well as channel erasures in the users-to-users and servers-to-users transmission links) is considered in this paper, where an attractive solution that aims to minimize the number of downloads from the backhaul servers in a cooperative fog data network is presented. More specifically, this paper aims to address the following question: *How should files be encoded (using IDNC) amongst users in generalized cooperative D2D-enabled transmissions over multiple time epochs, such that the remaining requests from the users (if any) can be delivered (employing IDNC) with a minimum number of transmissions from the backhaul servers?*

To address the question above, in this paper we have utilized the stochastic shortest path (SSP) technique to study the maximum backhaul offloading problem in the cooperative network coded fog data networks. The problem is first modelled using graphical representation proposed in [22], namely, the IDNC graph with induced subgraphs. This graph representation overcomes the limitations of the conventional graphical representation [23], as it is not compatible to implement with a system model having full-duplex communications and the additional constraints that arise from D2D-enabled communications. With this graph modelling of the system, the optimal solution is formulated using the SSP technique and shown to be NP-hard and not applicable for real-time applications [24]. As the optimal solution is NP-hard, a greedy heuristic algorithm is proposed, employing a maximum weighted vertex search over the graph model. Simulation results show that the proposed algorithm significantly outperforms the conventional uncooperative IDNC approach in reducing the downloads required from the servers of the cooperative fog data networks.

The main contributions of the paper can be summarized as follows:

(i) We solve the backhaul servers offloading problem in generalized network coded cooperative fog data networks by utilizing the IDNC graph with induced subgraphs as well as the SSP technique under the general assumptions where users can demand multiple files over multiple time epochs and the users and servers wireless channels are subject to erasures. This, to the best of our knowledge, did not exist in the literature before.

(ii) After theoretically formulating the maximum offloading problem in the generalized network coded cooperative fog data networks and guided by the properties of the optimal solution, we present a computationally simple heuristic graph-based algorithm to find network coded transmissions amongst the users and from the servers that efficiently reduce the number of transmissions required from the network servers.

(iii) We also present the complexity analysis of the proposed heuristic algorithm and the optimal solution and show that the complexity of the proposed algorithm is much lower than the optimal solution.

(iv) We then assess the performance of the proposed algorithm through extensive simulations as well as comparison with the optimal performance. The simulations and comparison results confirm the effectiveness of the proposed algorithm and the fact that its performance is very close to the optimal performance.

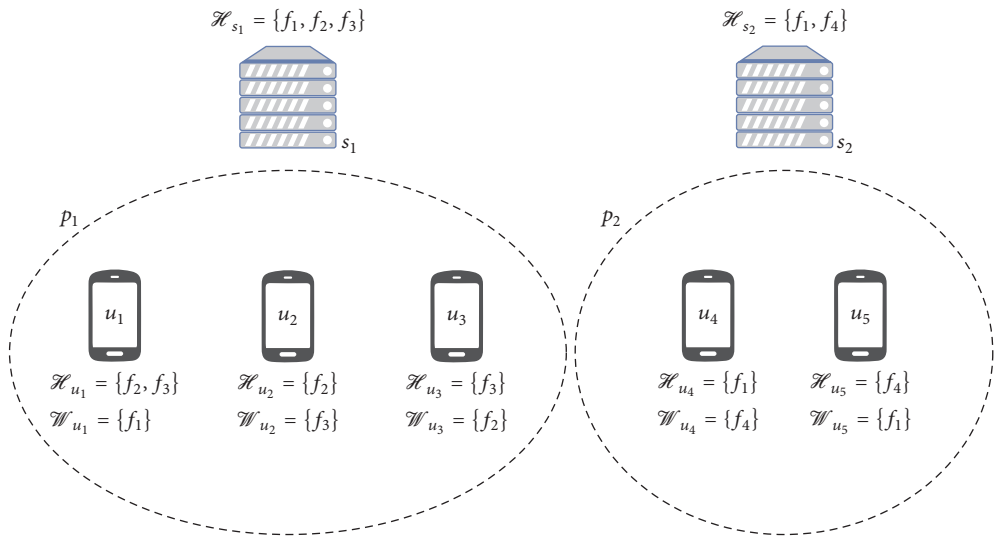


Figure 1: An illustration depicting the system model for a distributed storage network, showing two servers, six users, and two proximity based wireless networks, where users can conduct cooperative D2D communications.

The organization of the paper is as follows: System model and mathematical notation are presented in Section 2. In Section 3, the SSP-based problem formulation for the generalized cooperative fog data network model is proposed. Then the proposed heuristic algorithm is provided in Section 5. The simulation results and complexity analysis are presented in Sections 6 and 7, respectively. Lastly, the paper is concluded in Section 8.

2. System Model

A distributed wireless data network model is illustrated in Figure 1. In this model, there is a set of N_u users defined as $\mathcal{U} = \{u_1, \ldots, u_{N_u}\}$. In the system model, the assumption is made that all users are capable of full-duplex communications. The users will request to receive some files from a library of files defined as $\mathcal{F} = \{f_1, \ldots, f_{N_f}\}$ with N_f files that are collectively stored at the servers. The servers are defined in the set $S = \{s_1, \ldots, s_{N_s}\}$ with N_s servers. All servers are assumed to have full coverage, where the users in the coverage area are denoted by $\mathcal{U}(s_i)$ and must satisfy $\mathcal{U}(s_i) \cap \mathcal{U} = \mathcal{U}$. The model shows a distributed setting where the users are in coverage of multiple servers. Also in the model, multiple proximity networks (possibly Wi-Fi or LAN) are shown, where the users are capable of full-duplex cooperative D2D communications. The proximity regions are therefore defined as the proximity set $\mathcal{P} = \{p_1, \ldots, p_{N_p}\}$ with N_p proximity-enabled D2D communication networks. The proximity networks contain a subset of the users in $\mathcal{U}$, defined as $\mathcal{U}(p_i)$, that is, the users in the coverage area of the proximity-enabled network p_i. It is assumed that there is no overlap of the users in each proximity set; that is, the users in each proximity network that are "geographically close" can communicate locally but not outside this network. Unlike previous works [22, 23] that consider ideal transmissions amongst the users and from the servers, in this manuscript, we have made the more realistic assumption that the users-to-users and servers-to-users' communications links are subject to random erasure.

In order to leverage IDNC, it is assumed that the users have received files in previous time epochs, where a user u_i has partially downloaded some of the files from a transmitted frame which constitutes the users *Has* set $\mathcal{H}_{u_i}$. This first phase of the transmission is known as the initial transmission phase. During the initial transmission the servers will attempt to serve all files to the users in the network; following this initial transmission some users will have received only a portion of the files requested due to channel erasure. Furthermore, the remaining files wanted by user u_i in the frame form the user's *Wants* set, denoted as $\mathcal{W}_{u_i}$. The remaining files (if any) that are neither in the users *Wants* or *Has* sets are defined as the users *Lacks* set by $\mathcal{B}_{u_i}$. Similarly, the servers will store a subset of the files in $\mathcal{F}$; however the union of all files at the servers should contain the complete set of $\mathcal{F}$ (with possible repetition). Here, a server's *Has* set is defined as $\mathcal{H}_{s_i}$. It is assumed that the servers will maintain a global knowledge of the system state during the initial transmissions; that is, the users will respond with positive/negative feedback depending whether the user receives their files successfully or not. At completion of this phase the system will move into the recovery transmission phase.

IDNC can now be utilized to exploit users' side data to optimize the transmissions in the current time epoch, where there is now the possibility for IDNC transmissions with cooperative D2D communications supported by the servers. With the servers now aware of the state of the network, the servers will generate a *state feedback matrix* (SFM) to represent the network state. The SFM is generated as follows; generate the matrix $\mathbf{F} = [\gamma_{ij}]$, $\forall u_i \in \mathcal{U}, f_j \in \mathcal{F}$ where

$$\gamma_{ij} = \begin{cases} 0, & j \in \mathcal{H}_{u_i} \\ 1, & j \in \mathcal{W}_{u_i} \\ -1, & j \in \mathcal{B}_{u_i} \end{cases}$$

$$\mathbf{F} = \begin{array}{l} \\ \text{Users } (u_i) \downarrow \end{array} \overset{\text{Files } (f_j) \longrightarrow}{\begin{pmatrix} \gamma_{1,1} & \gamma_{1,2} & \cdots & \gamma_{1,j} \\ \gamma_{2,1} & \gamma_{2,2} & \cdots & \gamma_{2,j} \\ \vdots & \vdots & \ddots & \vdots \\ \gamma_{i,1} & \gamma_{i,2} & \cdots & \gamma_{i,j} \end{pmatrix}} \quad (1)$$

The SFM is generated for the recovery transmission phase and is shown in (1). Finally, when channel erasure is considered the probability of erasure between the receiving user u_i and the transmitter (either user u_j or server s_j) is defined as e_{ij} (also define $q_{ij} = 1 - e_{ij}$).

3. Problem Formulation

As discussed earlier, we consider the backhaul offloading maximization problem in cooperative fog data networks where the users in the network may demand multiple files and the recovery transmissions can occur over multiple time epochs. The network state is considered to be known after the initial transmission phase. The model will allow for D2D communications to take priority, where all remaining requests will be transmitted by the servers on orthogonal channels (assuming the servers have unlimited capacity). In this model, it is also assumed that the recovery transmissions (both users-to-users and servers-to-users transmissions) will be subject to erasures (i.e., for the probability of channel erasure to occur during any transmission).

4. IDNC Graph with Induced Subgraphs Model

To be able to formulate the optimal solution to the above-mentioned problem, the IDNC graph with induced subgraphs model [22] is employed in this paper. This IDNC graph model represents all the possible files that can be XORed together to create a network coded transmission that can be decoded by the targeted end-users (or transmissions not requiring any network coding). To form the model, firstly the graphs of interest in our considered system model are defined as follows: Graph $\mathcal{G} = \{\mathcal{G}_1, \ldots, \mathcal{G}_{N_p}\}$ with the subgraph $\mathcal{G}_i$ representing each discrete proximity network, as well as graph $\Psi = \{\Psi_1, \ldots, \Psi_{N_s}\}$ that is representing all servers, where the subgraph Ψ_i represents each individual server. To construct each of the graphs previously mentioned, proceed as follows:

(1) Generate Vertex Set: vertices are generated from a server and user perspective under the two following conditions:

(i) Generate a vertex set for every server s_i in $\mathcal{S}$ that is represented in the subgraph Ψ_i, generating the vertices v_{ijk}, $\forall s_i \in \mathcal{S}$ and $f_k \in (\mathcal{H}_{s_i} \cap \mathcal{W}_{u_j})$. The vertices of the subgraph are defined as $\Psi_{i_{(ijk)}}$.

(ii) Generate a vertex set for every user $u_i \in p_n$ in $\mathcal{U}$ that is represented in the subgraph $\mathcal{G}_n$, generating the vertices v_{ijk}, $\forall u_i \in \mathcal{U}$ and $f_k \in (\mathcal{H}_{u_i} \cap \mathcal{W}_{u_j})$ on the conditions $u_i \neq u_j$ and both $u_i, u_j \in p_n$. The vertices of the subgraph are defined as $\mathcal{G}_{n_{(ijk)}}$.

(2) Generate Coding Opportunity Edges: in each individual subgraph in $\mathcal{G}$ and Ψ, two vertices v_{ijk} and v_{lmn} are connected with an edge if they satisfy one of the following two conditions:

(i) $f_k = f_n$, $u_j \neq u_m$ and $u_i = u_l$ if in $\mathcal{G}$ (or $s_i = s_l$ if in Ψ), meaning the two requested files are the same, and these files are requested by two different users.

(ii) $f_n \in \mathcal{H}_{u_j}$ and $f_k \in \mathcal{H}_{u_m}$, representing a potential coding opportunity, so that when f_n and f_k are XORed both users can successfully decode and retrieve their requested file.

In the formulation so far, the graphs implemented represent coding or transmission opportunities from a user/server viewpoint. To further create a global awareness, induced subgraphs (subgraphs of $\mathcal{G}$ and Ψ) are incorporated that will represent the transmission conflicts (subgraph $\mathcal{K}$) and a subgraph to ensure only one transmission per user is permitted in the current time epoch (subgraph $\mathcal{L}$).

(3) Generate Induced Subgraphs: the two induced subgraphs described are generated as follows:

(i) The first set of induced subgraphs are defined as $\mathcal{K} = \{\mathcal{K}_1, \ldots, \mathcal{K}_D\}$; these are a subset of both graphs $\mathcal{G}$ and Ψ, where the subgraphs may contain the null-set of either $\mathcal{G}$ or Ψ but not both. To generate the subgraph $\mathcal{K}_j$, each vertex v_{ijk} in both $\mathcal{G}$ and Ψ will form a member of the subgraph $\mathcal{K}_j$ for every vertex that has the same user u_j and file f_k.

(ii) The next set of induced subgraphs are defined as $\mathcal{L} = \{\mathcal{L}_1, \ldots, \mathcal{L}_E\}$; these are a subset of both graphs $\mathcal{G}$ and Ψ. A subgraph $\mathcal{L}_i$ is formed for any two vertices v_{ijk} and v_{lmn}, where $u_i = u_l$ but $u_j \neq u_m$ or $f_k \neq f_n$.

In Figure 2, an illustration depicts the implementation of IDNC with the prescribed theoretical graph model for the example shown in Figure 1. In this example, it is helpful to show independently the IDNC subgraphs for each individual proximity network in subgraphs $\mathcal{G}_i$. While in subgraphs Ψ_i, the potential coded transmissions are shown in maximal cliques (a clique is a subset of the graph, where every distinct pair of vertices in the induced subgraph are pairwise adjacent. A maximal clique is one that cannot be

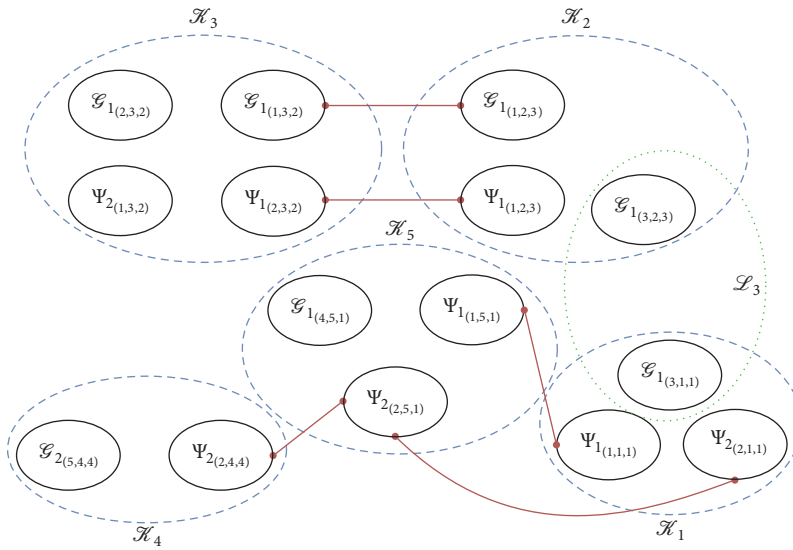


FIGURE 2: A visualization of the theoretical graph model proposed for system model shown in Figure 1. The figure shows coding opportunities represented by edges, transmission conflicts represented in subgraphs $\mathcal{K}_j$, and limitation of one transmission per user represented in subgraph $\mathcal{L}_3$.

a subset of a larger clique [25]) for each server s_i. In the graph model shown in Figure 2, it is clear that there is no interconnection between the graphs of $\mathcal{G}$ and Ψ (no edges connecting vertices). Therefore, the induced subgraphs approach allows the representation of particular conditions that need to be accounted for in the network setting. The graph $\mathcal{K}$ represents a set of subgraphs that ensures conflict-free transmissions. Additionally, the graph $\mathcal{L}$ contains a set of subgraphs that ensures users in a proximity network will not transmit more than once in the current time epoch.

4.1. The Proposed Optimal Problem Formulation Using SSP. In order to formulate the optimal solution using the IDNC graph with induced subgraphs model that represents the global state of the network, a shortest stochastic path problem (SSP) technique is employed. SSP is in fact a special case of the well known Markov decision processes (MDP) [26] where MDPs are problems where a sequence of decisions are made in stochastic environments, such as is the network model with channel erasures. In SSP problem, different possible situations that the system could encounter are modelled as states. In each state s, the system must select an action a from an action space that will charge it an immediate cost $c(s, a)$. The terminating condition of the system can be thus represented as a zero cost absorbing state. Once an action a is taken at state s, the system can move to a state s with probability $P_a(s, a)$, which only depends on the current state and the taken action. An SSP policy $\pi = [\pi(s)]$ is a mapping from state space to action space that associates a given action with each of the states. The optimal policy π^* is the one that minimizes the cumulative mean cost until the absorbing state is reached.

To formulate the SSP problem, the following definitions are required [14]:

(1) *State space* $\mathcal{S}$: the state space $\mathcal{S}$ is represented by all the possibilities that the SFM may represent. At each state s, the SFM will represent the *Lacks*, *Wants* and *Has* sets of all users. The state s_a (the state where the system has reached absorption) is defined for when all users have received their wanted packets such that $|\mathcal{W}_{u_i}(s_a)| = 0, \ \forall i \in U$.

(2) *Action spaces* $\mathcal{A}(s)$: at each state s the system can represent there is a set of actions $\mathcal{A}$, that is, a selection of maximal cliques from both graphs $\mathcal{G}(s)$ and $\Psi(s)$ that are constructed from the SFM at state s. All actions $a \in \mathcal{A}(s)$ at state s are defined by first selecting a set of maximal cliques $\Gamma_{\mathcal{G}(s)}$ from graph $\mathcal{G}(s)$. Then for all vertices in $\Gamma_{\mathcal{G}(s)}$ that are also a subset of the graphs $\mathcal{K}(s)$ and $\mathcal{L}(s)$ that are to be removed from the graphs $\Psi(s)$ and $\mathcal{G}(s)$. The set of maximal cliques selected in $\Gamma_{\mathcal{G}(s)}$ must result in removing all vertices in $\mathcal{G}(s)$ after this process. Finally, a set of maximal cliques $\Gamma_\Psi(s)$ is selected from the remaining graph $\Psi(s)$, ensuring all users are targeted in the current time epoch.

(3) *State-Action transition probabilities*: in order to define the state-action transitions probability $P_a(s, s')$ for the action a it is useful to introduce two new sets:

$$\begin{aligned}\mathcal{X} &= \left\{i \in \mathcal{T}\left(\Gamma_{\mathcal{G}(s)}\right), \ i \in \mathcal{T}\left(\Gamma_{\Psi(s)}\right) \mid \left|\mathcal{W}_{u_i}\left(s'\right)\right| \right. \\ &\quad \left. < \left|\mathcal{W}_{u_i}(s)\right|\right\} \\ \mathcal{Y} &= \left\{i \in \mathcal{T}\left(\Gamma_{\mathcal{G}(s)}\right), \ i \in \mathcal{T}\left(\Gamma_{\Psi(s)}\right) \mid \left|\mathcal{W}_{u_i}\left(s'\right)\right| \right. \\ &\quad \left. = \left|\mathcal{W}_{u_i}(s)\right|\right\},\end{aligned} \tag{2}$$

where $\mathcal{T}(\Gamma_{\mathcal{G}(s)})$ are the targeted users in the maximal clique that is selected from the graph $\mathcal{G}(s)$.

Also $\mathcal{T}(\Gamma_{\Psi(s)})$ is defined as the targeted users in the maximal clique that is selected from the graph $\Psi(s)$. Therefore, the state-action probability is given as

$$P_a\left(s, s'\right) = \prod_{i \in \mathcal{Y}} q_{ij} \prod_{i \in \mathcal{X}} e_{ij}. \tag{3}$$

(4) *State-action costs*: in the problem formulation the aim is to reduce the amount of transmissions from the servers-to-users. Accordingly, transmissions that occur from using cooperative transmissions will incur no cost. Moreover, for every state any state-action transition will incur a cost of 1 for each orthogonal channel used by the servers. For example, if an action at state s required the servers to transmit on a total of three orthogonal channels collectively, the cost incurred will be 3. Therefore, the amount of maximal cliques that are in the set $\Gamma_{\Psi(s)}$ will directly determine the cost, which is the same as the cardinality of set $\Gamma_{\Psi(s)}$ or equivalently $c(s, a) = |\Gamma_{\Psi(s)}|$, as user-to-user transmissions come at zero cost.

The optimal policy π^* of an SSP is the one that minimizes the cumulative mean cost until completion is reached. This policy is given in

$$\pi^*(s) = \arg\min_{a \in \mathcal{A}(s)} \left\{ c(s, a) + \sum_{s' \in \mathcal{S}(s,a)} P_a\left(s, s'\right) V_{\pi^*}\left(s'\right) \right\}, \tag{4}$$

where $V_{\pi^*}(s)$ is the expected cumulative cost until the system reaches absorption. From the SSP problem formulation, it is clear that the optimal policy requires checking through all possible actions (i.e., maximal cliques), which is proven to be NP-hard [24].

5. The Proposed Greedy Heuristic Algorithm

In this section, a greedy heuristic approach is proposed that can be solved in real-time and aims to efficiently reduce the number of downloads from the servers. The fall back of a greedy heuristic scheme is that it does not in fact guarantee a global optimum, although this scheme will, on average, give a good approximation to it. An attractive feature of the graph-based formulation proposed in Section 3 is that a maximum weighted vertex search under a greedy policy can be directly applied on the graph model.

As discussed already, if an optimal solution is NP-hard there is a requirement for a suboptimal solution for real-time communications. Figures 3 and 4 demonstrate by example the impact that state-action transitions may have depending on the action selected from the current state. Figure 3 shows the case of selecting action a_1, where the associated cost is zero. Here we see that if e_1 and e_2 are large there is a greater chance that the packets will not be received. Therefore, it is desirable that a user that has a poor channel (high probability of erasure) with the server to be targeted with cooperative transmissions. Furthermore, Figure 4 shows a subset of consecutive actions and the resulting state and cost;

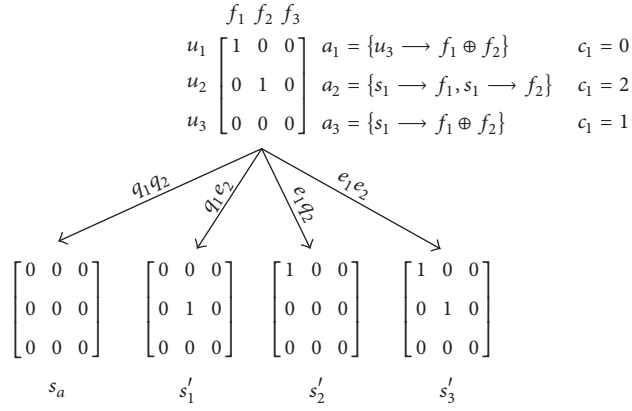


Figure 3: Example SFM at the beginning of the recovery transmissions, shown is a subset of possible actions that can be taken at the current time epoch. SFM transition shows dependence on channel erasure for a given action, in this example that is a_1.

here clearly the shortest (lowest cost) path to absorption is the ones that incur zero cost.

Therefore, it will be desirable for users to serve themselves with the greatest chance of serving as many users simultaneously as possible in each time epoch. To represent this a vertex $v_{ijk}(s)$ in either graph $\mathcal{G}$ or graph Ψ will have a large weight $w_{ijk}(s)$ if it has a large number of adjacent vertices, which themselves have a large number of adjacent vertices. Additionally, as shown in the motivating example it is preferable to place further (higher) weighting on transmissions for users with high probability of erasure with the server.

Therefore, a weight τ is first introduced for each user in $\mathcal{U}$ that is proportional to a users' likely mean completion time (dependent on user-to-server channel). The weight also varies depending on the state the system is in; therefore define the weight for each state s

$$\tau_{u_i}(s) = \frac{\left|\mathcal{W}_i(s)\right|}{1 - e_{ij}}, \tag{5}$$

where (5) is large for a user with a large *Wants* set and a large probability of channel erasure for user-to-server. Now that a weight is defined in (5), at each state s the set of vertices adjacent to $v_{ij}(s)$ is defined as the set of vertices in $\mathcal{N}(s)$. From this the weighted vertex degree $\Delta_{ij}(s)$ can be calculated as

$$\Delta_{ij}(s) = \sum_{\forall v_{kl} \in \mathcal{N}(s)} \tau_{u_k}(s), \tag{6}$$

where (6) defines a value for each vertex that is proportional to how many adjacent vertices it has. Moreover, it describes the potential that the particular vertex has in relative terms of coding potential with other users. Finally, the multiplication of both weights will yield the final weight for each vertex and is defined in

$$w_{ij}(s) = \tau_{u_i}(s)\, \Delta_{ij}(s), \tag{7}$$

where a large weight $w_{ij}(s)$ for the vertex $v_{ij}(s)$ is reflected by a large weight $\tau_{u_i}(s)$ and also has a large number of connected

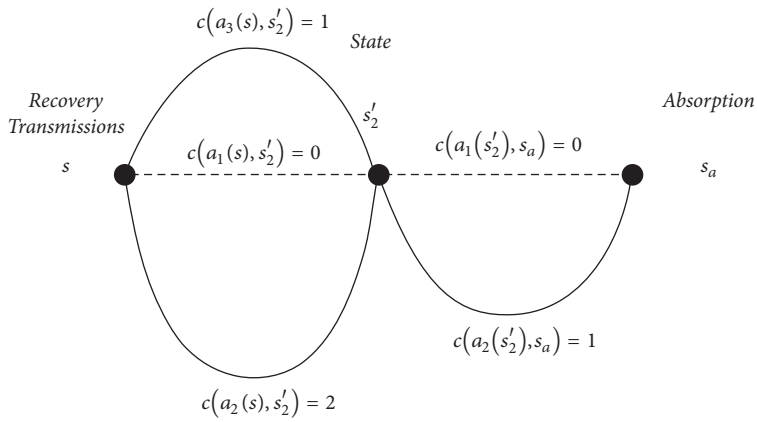


FIGURE 4: Visualization of a SSP, here only state-action probabilities of no erasure are considered. The diagram shows the transition from state s to absorption state s_a that takes multiple paths, where the shortest cost to absorption will be taking the actions of 0 cost at both states s and s_2'.

vertices that are also connected to a large number of vertices represented by $\Delta_{ij}(s)$.

Once the weights are calculated for all the vertices in the graph, the greedy heuristic search will select the vertex v_{ijk} with the largest weighting, or between those with the same largest weight with equal probability. The algorithm then removes all nonadjacent vertices to v_{ijk} and then checks if the vertex v_{ijk} belongs to a subgraph $\mathcal{K}_j$ or $\mathcal{L}_i$ and will remove *all* other vertices that are a member of either subgraph.

Secondly, the algorithm will then update all weights in $\mathcal{G}$ before selecting the next (if any) adjacent vertex in graph $\mathcal{G}$ that forms a clique with all previously selected vertices. The algorithm then continues to iterate these steps until no more vertices can be added to the clique (resulting in a maximal clique). Finally, once a maximal clique listing is found and removed, the whole procedure is iterated until no more vertices are left in the graph $\mathcal{G}$.

At this stage, the algorithm has removed all possible D2D cooperation's available, with the aim of minimizing the amount of downloads from the servers. Therefore, the remaining vertices in graph Ψ need to be served, which were not served locally from D2D cooperation. The exact same procedure can be conducted on the remaining vertices in graph Ψ, where each maximal clique represents one download from a server and this continues until all vertices are removed from the graph. Once all vertices have been removed from the graph Ψ the system will have reached absorption and all users will have received the file in their *Wants* sets.

It is worth mentioning that as users are required to receive multiple files (multiple time epochs required) and there is a chance of channel erasure in transmission, after each sequence of transmissions the SFM needs to be updated. The updated SFM should then be again represented in the graph model where the algorithm will repeat. The process continues until all users have received all wanted files and the system reaches absorption.

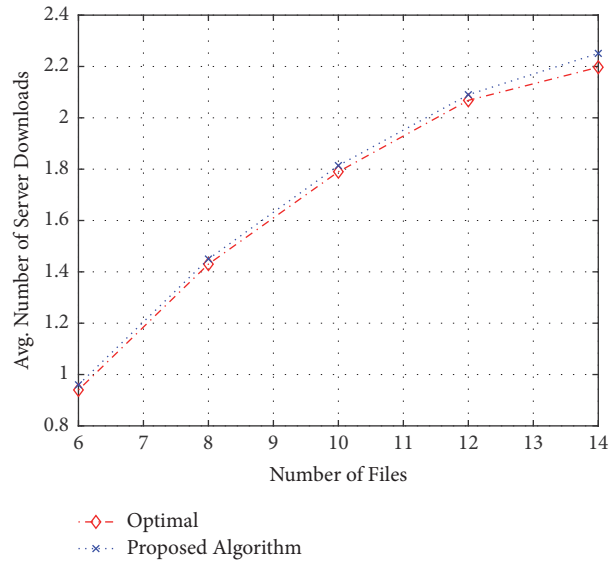


FIGURE 5: Performance of algorithm against the optimal solution.

6. Simulation Results

In this section, the simulation results are presented for the proposed algorithm in a cooperative D2D setting in comparison with a uncooperative decentralized conflict-free IDNC approach that was incorporated in [21]. The aim of both approaches is to reduce the number downloads from the servers. The cooperative approach is a natural next step in the evolution and adoption of F-RAN in the future wireless communications systems to maximize backhaul offloading.

Firstly, in Figure 5 the performance of the proposed heuristic algorithm is compared with the optimal solution. The optimal solution is calculated by a brute force algorithm that checks over all possible IDNC combinations. The simulation results are presented for a small network setting, where there is one server and four users requesting one file with two files in each user's *Has* set. It can be seen from

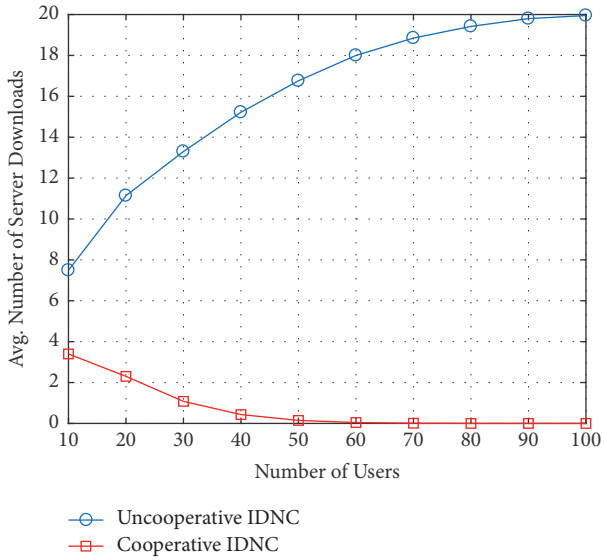


Figure 6: The average number of downloads required from the servers as a function of the number of users (erasure free case).

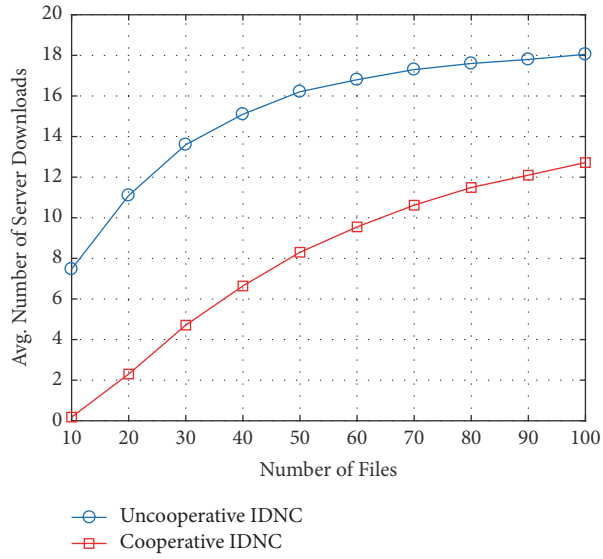


Figure 7: The average number of downloads required from the servers as a function of the number of files (erasure free case).

the results shown in Figure 5 that the proposed algorithm shows a good approximation to the optimal solution, with a small divergence for an increasing amount of files. This divergence from the optimal solution is expected, as the network size increases the divergence from the optimal solution is expected to increase.

It is also worth mentioning that the proposed heuristic algorithm can be solved with a much lower computational complexity and in real-time. However, as expected, there will be a tradeoff here between performance and complexity (nevertheless, the performance of the proposed heuristic is very close to the optimal one).

In the rest of this section, we assume there are two servers available with total coverage of all users in the network, while in the cooperative model a dual network is considered, where the users are split evenly between the two proximity networks p_1 and p_2 (similar to Figure 1). In the simulation results presented in Figures 6 and 7, each user is interested in receiving one file and has two files already received and stored in the *Has* set (when fixed), where the recovery downloads are to be completed in one time epoch. It is also assumed that the transmissions are erasure free. A summary of the data sets used (when fixed) in the simulations can be seen in Table 1.

In Figure 6, the average number of downloads required from the servers for a fixed number of files in the transmission frame of $N_f = 20$ is shown, as a function of the number of users N_u. The result shows that, for the algorithm implemented for a cooperative D2D-enabled setting, as the number of users increase the average number of server downloads tends to monotonically decrease. Intuitively, this result is expected as more users in the network will result in a greater likelihood that the users can serve themselves independently from the servers, as the collective *Has* set of the users in the network will cover the files in the frame $\mathcal{F}$. Additionally, it can be seen that, in comparison to a conventional uncooperative conflict-free IDNC approach, as the network size increases there is significant improvement, where there is an improvement of approximately 550% with only 20 devices in the network setting. Furthermore, approximately no downloads from the servers are required as the number of users approaches 60 in this network setting, that is, 30 users in each D2D-enabled network.

In Figure 7 number of users are fixed to 20, while varying the amount of files per transmission frame. Figure 7 shows the results for cooperative versus uncooperative IDNC transmission schemes. In both cases, it can be seen as the number of files increases, both schemes show a similar increase on the number of downloads required from the servers. Although the two schemes tend to converge if an asymptotic limit is considered, the cooperative scheme still shows reasonable improvement of approximately 50% for up to 100 files. Again, this result is expected as increasing the number of files in a frame reduces the potential to leverage a coded transmission. Additionally, as the number of files increase the likelihood that a user can serve another user in a cooperative D2D-enabled network is reduced.

In the next two simulations presented in Figures 8 and 9, D2D cooperation with channel erasure is considered. In both simulations erasure between user-to-server transmissions is evenly distributed from $[0, 0.03]$, while the transmissions from user-to-users are also evenly distributed from $[0, 0.01]$. In this case, it is assumed that it is more likely that users will be closer geographically and therefore will in general have a better chance for a successful transmission. Firstly in Figure 8, the number of files in the frame is fixed to 20, while all users will demand two random files from the frame. It can be seen in Figure 8 that as the number of users increases the cooperative approach significantly outperforms the uncooperative approach. Moreover, as the number of users increases the improvement also increases. In this case, the model shows a performance increase range of approximately 150–300% from 10 to 100 users.

Table 1: A summary of the data sets used in the simulations, the stated figures are given for the case when the parameter is fixed in the simulation. In all simulations, the number of proximity D2D enabled networks (N_p) is fixed to 2.

Servers (N_s)	Files (N_f)	Users (N_u)	*Has set* $\lvert\mathcal{H}_u\rvert$	*Wants set* $\lvert\mathcal{W}_u\rvert$
2	20	20	2	1

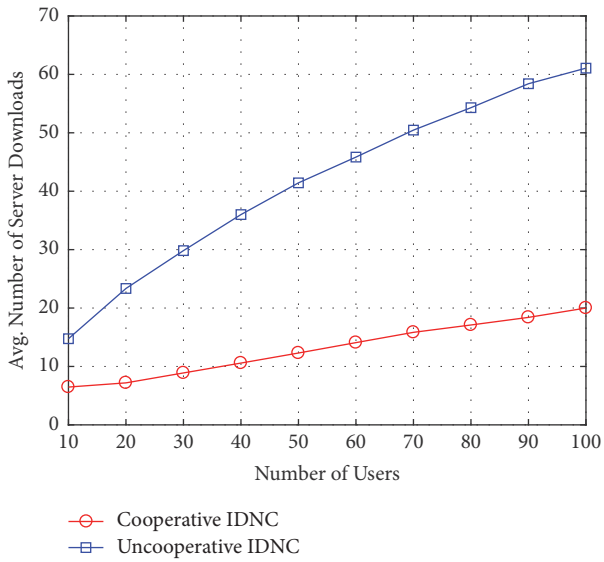


Figure 8: The average number of downloads required from the servers as a function of the number of users in the network (channels are subject to erasure).

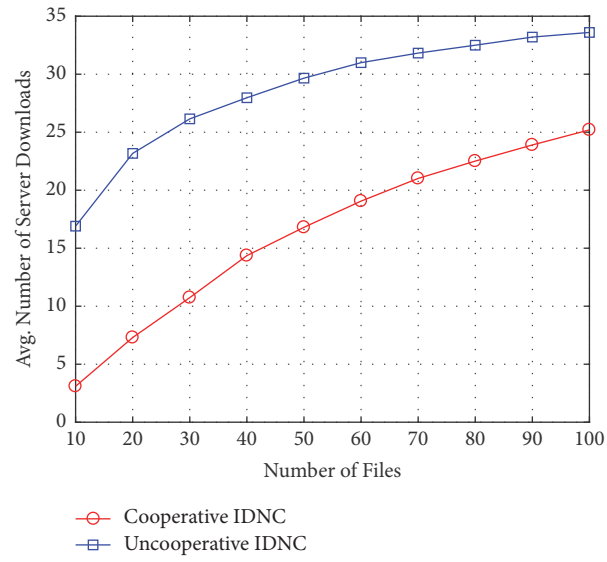


Figure 9: The average number of downloads required from the servers as a function of the number of files (channels are subject to erasure).

In Figure 9, the number of users in the network is fixed at 20 while the number of files in the frame is varied. The results in Figure 9 show that there is significant offloading of the network backhaul, although it appears that the two approaches will tend to converge for a very large number of files. This result is intuitively expected; this is because as the number of files increases with a fixed number of users, the likelihood of the users' ability to diffuse the wanted packets is diminished. Nevertheless, the cooperative approach shows significant ability to reduce the traffic from servers.

7. Complexity Analysis

To gain an appreciation of the performance of the proposed heuristic algorithm in comparison to the optimal solution, we will investigate their worst-case time complexities. Firstly, we will analyze the worst-case time complexity of the optimal solution. As the optimal solution to the problem (the optimal policy) requires checking through all possible actions (i.e., finding all maximal cliques) we would require some algorithm to execute this. One popular method to achieve this, which is generally more efficient than other algorithms, is the Bron-Kerbosch (B-K) algorithm [27]. To find the worst-case time complexity, we need to find the vertex set size of the entire graph. The size of the server graph (Ψ) will be of the order $O(N_u N_f)$, while the size of the graph representing potential cooperative transmissions ($\mathcal{G}$) is of the order of $O(N_s N_u N_f)$. Therefore, by utilizing the B-K algorithm to solve find the optimal solution, the worst-case time complexity would be $O(3^{(O(N_u N_f)+O(N_s N_u N_f))/3})$ [28].

Now we will consider the worst-case time complexity of the proposed heuristic algorithm in Section 5. The first stage of the heuristic algorithm is to scan the entire graph $\mathcal{G}$, while assigning a weight to each vertex, and then to conduct a maximum weighted vertex search. This process is then repeated until the maximum weighted maximal clique is found. There are $N_u N_f$ vertices in graph $\mathcal{G}$, where finding the corresponding vertex with the maximum weight will require $N_u N_f - 1$ operations in the first iteration. The next iteration would require $N_u N_f - 2$ and so on until the final iteration would require $N_u N_f - N_u$ operations. Hence, the total number of operations that is required will be $\sum_{i=1}^{N_u}(N_f N_u - i) = (1/2)(N_f^2 N_u^2 + N_f N_u)$. Therefore, the worse-case time complexity to find the cooperative communications amongst users is $O(N_f^2 N_u^2)$. Once the initial graph has been served (that is, the cooperative transmissions completed), there are still potentially N_{uc} unserved clients that could not be served in a D2D communication in the current time epoch and need to be served by the servers. Again, we proceed with the same algorithm as before, conducting a maximal weighted vertex search on graph Ψ; however the order of the graph is now $O(N_s N_{uc} N_f)$. Since a maximal clique (or maximal independent set) cannot contain more than N_u vertices, the worst-case time complexity requires N_u of the above iterations. Therefore, the computational complexity for the graph

Table 2: Magnitude of worst-case complexity for each graph in the proposed algorithm.

Scenario	S_1	S_2	S_3
Graph $\mathcal{G}$	40000	10^6	1.44×10^6
Graph Ψ	1280	2880	24000

Table 3: Parameters used in each scenario generated in Table 2, where each case has one server.

Scenario	S_1	S_2	S_3
N_f	20	20	60
N_u	10	50	20
N_{uc}	8	12	10

Ψ (i.e., graph representing servers' transmissions to users that are not targeted by the D2D cooperative transmissions) will be $O(N_u[N_f N_{uc} N_s + \log(N_f N_{uc} N_s)]) = O(N_s N_{uc}^2 N_f)$. Thus, the entire algorithm's worst-case time complexity to serve all users will be $O(N_f^2 N_u^2) + O(N_s N_{uc}^2 N_f)$. Given that in reality the number of servers N_s in the network is much smaller than the number of users N_u and also the number of unserved users N_{uc} is much smaller than N_u, the worst-case complexity of the proposed algorithm can be estimated to be of order $O(N_f^2 N_u^2)$. Table 2 shows the numerical values for the magnitude of worst-case complexities of graphs $\mathcal{G}$ and Ψ in the proposed algorithm under the scenarios summarized in Table 3.

From the results shown in Table 2, it can be seen that the worst-case time complexity of solving the proposed algorithm will reduce to just that of graph $\mathcal{G}$, that is, the cooperative transmissions. Therefore, the worst-case time complexity of the proposed algorithm will be $O(N_f^2 N_u^2)$, far more efficient than the B-K algorithm. Moreover, it can be seen that the worst-case time complexity of the proposed algorithm will also be independent of the number of servers in the network, assuming that $N_u \gg N_s$.

8. Conclusion and Future Work

In this paper, the problem of offloading the expensive backhaul of data network servers through a network coded cooperative D2D network model is investigated. The problem is formulated using the IDNC induced subgraph model, where the optimal solution requires finding maximal cliques of multiple graphs. The paper considers the generalized system model where users can demand multiple files and the transmission channels are subject to erasure. Having formulated the optimal solution using the SSP technique, it is found that the optimal solution is intractable and not solvable in real-time. Therefore, a greedy heuristic algorithm is employed using a maximum weighted vertex search approach on the IDNC graph with induced subgraphs. This heuristic when compared with the optimal solution for a small network showed to be a good approximation to the optimal solution. The simulation results show a significant improvement over the conventional method that incorporates IDNC in a distributed fashion without D2D enabled cooperation.

The first recommendation for further work would be to consider a system model with imperfect feedback from the users. In this scenario, we would remove the assumption that the servers receive perfect feedback on the reception status of the transmitted files. In reality, there is the likelihood that the feedback itself may be corrupted; therefore a more efficient online algorithm could consider these uncertainties. Lastly, it would also be beneficial to consider the asymptotic bounds to the problem.

Disclosure

This research paper is an extension of the work in the conference paper "*Offloading of Fog Data Networks with Network Coded Cooperative D2D communications*" [22] that was awarded the "Best Paper Award" in the 9th EAI International Conference on Mobile Networks and Management (EAI MONAMI 2017).

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] Cisco, "Cisco visual networking index: Global mobile data traffic forecast update," Tech. Rep., 2016.

[2] M. Ji, G. Caire, and A. F. Molisch, "Wireless device-to-device caching networks: basic principles and system performance," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 1, pp. 176–189, 2016.

[3] A. G. Dimakis, K. Ramchandran, Y. Wu, and C. Suh, "A Survey on network codes for distributed storage," *Proceedings of the IEEE*, vol. 99, no. 3, pp. 476–489, 2011.

[4] A. G. Dimakis, P. B. Godfrey, Y. Wu, M. J. Wainwright, and K. Ramchandran, "Network coding for distributed storage systems," *IEEE Transactions on Information Theory*, vol. 56, no. 9, pp. 4539–4551, 2010.

[5] D. S. Papailiopoulos, J. Luo, A. G. Dimakis, C. Huang, and J. Li, "Simple regenerating codes: Network coding for cloud storage," in *Proceedings of the IEEE Conference on Computer Communications, INFOCOM 2012*, pp. 2801–2805, USA, March 2012.

[6] A. A. Al-Habob, S. Sorour, N. Aboutorab, and P. Sadeghi, "Conflict free network coding for distributed storage networks," in *Proceedings of the IEEE International Conference on Communications, ICC 2015*, pp. 5517–5522, UK, June 2015.

[7] M. A. Maddah-Ali and U. Niesen, "Fundamental limits of caching," *Institute of Electrical and Electronics Engineers Transactions on Information Theory*, vol. 60, no. 5, pp. 2856–2867, 2014.

[8] 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. 13–15, Finland, August 2012.

[9] R. Tandon and O. Simeone, "Cloud-aided wireless networks with edge caching: Fundamental latency trade-offs in fog Radio Access Networks," in *Proceedings of the 2016 IEEE International*

Symposium on Information Theory, ISIT 2016, pp. 2029–2033, esp, July 2016.

[10] N. Golrezaei, A. Molisch, A. G. Dimakis, and G. Caire, "Femtocaching and device-to-device collaboration: A new architecture for wireless video distribution," *IEEE Communications Magazine*, vol. 51, no. 4, pp. 142–149, 2013.

[11] K. Shanmugam, N. Golrezaei, A. G. Dimakis, A. F. Molisch, and G. Caire, "FemtoCaching: wireless content delivery through distributed caching helpers," *Institute of Electrical and Electronics Engineers Transactions on Information Theory*, vol. 59, no. 12, pp. 8402–8413, 2013.

[12] M. A. Maddah-Ali and U. Niesen, "Decentralized Coded Caching Attains Order-Optimal Memory-Rate Tradeoff," *IEEE/ACM Transactions on Networking*, vol. 23, no. 4, pp. 1029–1040, 2015.

[13] R. Ahlswede, N. Cai, S. R. Li, and R. W. Yeung, "Network information flow," *Institute of Electrical and Electronics Engineers Transactions on Information Theory*, vol. 46, no. 4, pp. 1204–1216, 2000.

[14] S. Sorour and S. Valaee, "On minimizing broadcast completion delay for instantly decodable network coding," in *Proceedings of the IEEE International Conference on Communications (ICC '10)*, pp. 1–5, Cape Town, South Africa, May 2010.

[15] S. Sorour and S. Valaee, "Minimum broadcast decoding delay for generalized instantly decodable network coding," in *Proceedings of the 53rd IEEE Global Communications Conference, GLOBECOM 2010*, USA, December 2010.

[16] E. Drinea, C. Fragouli, and L. Keller, "Delay with network coding and feedback," in *Proceedings of the 2009 IEEE International Symposium on Information Theory, ISIT 2009*, pp. 844–848, Republic of Korea, July 2009.

[17] P. Sadeghi, D. Traskov, and R. Koetter, "Adaptive network coding for broadcast channels," in *Proceedings of the 2009 Workshop on Network Coding, Theory and Applications, NetCod '09*, pp. 80–85, Switzerland, June 2009.

[18] L. Keller, E. Drinea, and C. Fragouli, "Online broadcasting with network coding," in *Proceedings of the 2008 4th Workshop on Network Coding, Theory, and Applications, NetCod 2008*, Hong Kong, January 2008.

[19] S. Sorour and S. Valaee, "An adaptive network coded retransmission scheme for single-hop wireless multicast broadcast services," *IEEE/ACM Transactions on Networking*, vol. 19, no. 3, pp. 869–878, 2011.

[20] P. Baran, "On distributed communications networks," *IEEE Transactions on Communications*, vol. 12, no. 1, pp. 1–9, 1964.

[21] Y. Shnaiwer, S. Sorour, P. Sadeghi, N. Aboutorab, and T. Y. Al-Naffouri, "Network-Coded Macrocell Offloading in Femtocaching-Assisted Cellular Networks," *IEEE Transactions on Vehicular Technology*, 2017.

[22] B. Quinton and N. Aboutorab, "Offloading of fog data networks with network coded cooperative d2d communications," in *Proceedings of the In 9th EAI International Conference on Mobile Networks and Management (EAI Monami 2017)*, pp. 1–13, EAI, 2017.

[23] Y. N. Shnaiwer, S. Sorour, N. Aboutorab, P. Sadeghi, and T. Y. Al-Naffouri, "Network-coded content delivery in femtocaching-assisted cellular networks," in *Proceedings of the 58th IEEE Global Communications Conference, GLOBECOM 2015*, usa, December 2015.

[24] C. S. Edwards and C. H. Elphick, "Lower bounds for the clique and the chromatic numbers of a graph," *Discrete Applied Mathematics*, vol. 5, no. 1, pp. 51–64, 1983.

[25] J. A. Bondy and U. S. R. Murty, *Graph Theory with Applications*, Macmillan Press, New York, NY, USA, 1976.

[26] M. L. Puterman, "Markov decision processes," in *Stochastic Models*, vol. 2 of *Handbooks Oper. Res. Management Sci.*, pp. 331–434, North-Holland, Amsterdam, 1990.

[27] C. Bron and J. Kerbosch, "Algorithm 457: finding all cliques of an undirected graph," *Communications of the ACM*, vol. 16, no. 9, pp. 575–577, 1973.

[28] E. Tomita, A. Tanaka, and H. Takahashi, "The worst-case time complexity for generating all maximal cliques and computational experiments," *Theoretical Computer Science*, vol. 363, no. 1, pp. 28–42, 2006.

Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 4680867, 10 pages
https://doi.org/10.1155/2018/4680867

Research Article

Intrusion Detection System Based on Decision Tree over Big Data in Fog Environment

Kai Peng,[1] **Victor C. M. Leung**,[2] **Lixin Zheng**,[1,3] **Shangguang Wang**,[4] **Chao Huang**,[1] **and Tao Lin**[1]

[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]*State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China*

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

Received 6 December 2017; Accepted 4 February 2018; Published 6 March 2018

Academic Editor: Xuyun Zhang

Fog computing, as the supplement of cloud computing, can provide low-latency services between mobile users and the cloud. However, fog devices may encounter security challenges as a result of the fog nodes being close to the end users and having limited computing ability. Traditional network attacks may destroy the system of fog nodes. Intrusion detection system (IDS) is a proactive security protection technology and can be used in the fog environment. Although IDS in tradition network has been well investigated, unfortunately directly using them in the fog environment may be inappropriate. Fog nodes produce massive amounts of data at all times, and, thus, enabling an IDS system over big data in the fog environment is of paramount importance. In this study, we propose an IDS system based on decision tree. Firstly, we propose a preprocessing algorithm to digitize the strings in the given dataset and then normalize the whole data, to ensure the quality of the input data so as to improve the efficiency of detection. Secondly, we use decision tree method for our IDS system, and then we compare this method with Naïve Bayesian method as well as KNN method. Both the 10% dataset and the full dataset are tested. Our proposed method not only completely detects four kinds of attacks but also enables the detection of twenty-two kinds of attacks. The experimental results show that our IDS system is effective and precise. Above all, our IDS system can be used in fog computing environment over big data.

1. Introduction

Fog computing [1, 2] was defined as a highly virtualized computing platform for migrating cloud computing center tasks to network edge devices. Fog computing provides computing, storage, and networking service between mobile users and traditional Cloud platform, which is complementary to Cloud. The fog computing introduces the middle layer between the cloud and the mobile users, extending the cloud-based network architecture [3–6]. A basic fog framework is shown in Figure 1, each mobile user is connected to one of the fog nodes. Meanwhile, fog nodes could be interconnected with each other and are linked to the Cloud [7]. The fog computing reduces unnecessary multiple communication between the cloud computing center and the mobile users. For instance, when the number of users has increased dramatically, these users can obtain the service by visiting the contents of the cache in the fog servers so as to reduce network delay [8]. And it also significantly reduces the bandwidth of the backbone link load [9, 10]. Unfortunately, the nodes in fog environment are close to the mobile users, and fog computing nodes are usually composed of devices with weak computing ability. Traditional network attacks are widely presented in fog environment; fog devices may face network security challenges. However, Intrusion Detection Systems (IDS) can be used for fog environment [11].

IDS is designed to ensure network security and the main task is detect malicious activities of the host or network

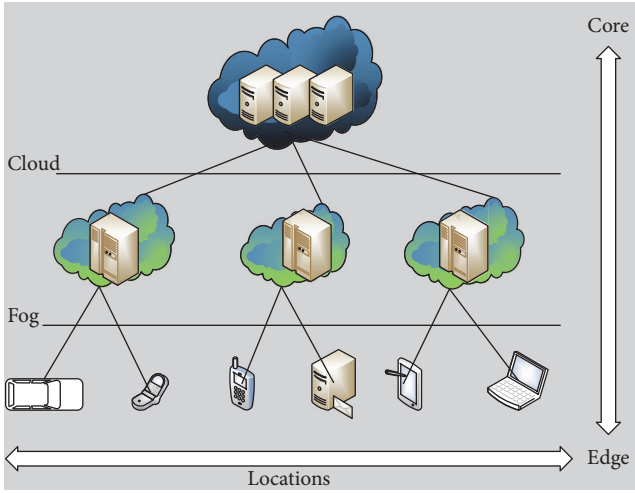


Figure 1: Fog between edge and cloud.

and then respond in a timely manner [12]. The definition of intrusion detection was first formally described in the 1980s [13]. In addition, the concept of real-time anomaly detection was proposed by Denning [14]. Pattern matching algorithm is one of the core technologies of IDS. Misuse detection based on AC, BM, MWM, and other matching algorithms [15] can make IDS have a passive detection of known attacks. However, modern attacks are increasingly inclined to form an unknown intrusion technology by integrating a variety of known intrusion technology. Meanwhile, improved IDS methods usually take proactive protection based on deviation detection and user behavior anomaly detection. For instance, statistical model, Bayesian reasoning, and cluster analysis [16] can make up for the lack of pattern matching, so that the system has a certain detection of unknown attacks. KNN algorithm [17] is widely used in pattern recognition, classification, and regression. Same as KNN, vector automatic classification algorithms, support vector machine [18–20], neural network algorithm [21], Bayesian algorithm [22–24], and K means algorithm are also widely used for IDS [25, 26].

Although the IDS in tradition network has been well investigated, unfortunately directly use them in fog computing environment may not inappropriate. Fog nodes produce massive amounts of data at all times, and, thus, enabling an IDS system over big data in fog environment is of paramount importance. More specifically, the existing researches mainly present the experiments on 10% KDDCUP99 dataset [27]. Although these methods have achieved good results, we cannot judge their efficiency when they are presented in the big data environment, even in the full dataset. In addition, there are four classification methods for network attacks, and also twenty-two classification methods in KDDCUP99. However, the existing research mainly focuses on the detection precision of four attacks but did not consider the detection of twenty-two attacks.

In order to address the above issue, we propose an IDS system based on decision tree over Anaconda [28]. Firstly, we propose a preprocessing algorithm to digitize the strings in the given dataset and then normalize the whole data, to ensure the quality of the input data so as to improve the efficiency of detection. Secondly, we use decision tree method for the detection of network attacks in our proposed IDS system, and then we compare this method with Naïve Bayesian method as well as KNN method. More specifically, three modes of Naïve Bayesian method are compared. And the experiment results show that our proposed IDS system is precise.

Our contributions in this study can be summarized as follows.

(1) For one thing, both the 10% dataset and the full dataset are tested in our IDS system, which proves that our IDS system is effective for big data environment.

(2) For another, we not only complete the detection of four kinds of attacks but also implement the detection of twenty-two kinds of attacks. The results show that our IDS system has a higher detection coverage of network attacks.

(3) In addition, the calculation time of each method is compared. To ensure the detection accuracy, although the calculation time of decision tree is not the best one, the time is also acceptable and can be used for big data environment.

The rest of the paper is organized as follows. In Section 2, the preliminaries are introduced. Section 3 specifies our proposed IDS system. The experimental evaluation is described in Section 4. Section 5 presents the related work. Finally, we conclude our work and describe the future work in Section 6.

2. Preliminaries

In this section, we firstly introduce the problem model and relevant formulas in Section 2.1 and then introduce the evaluation indicators of IDS detection in Section 2.2.

2.1. Problem Model and Relevant Formulas. The object of decision tree is to construct a decision tree model based on a given dataset to enable it to classify the new instances correctly. There are many methods to construct the decision tree, such as ID3 and C4.5 [29] and CART (Classification and Regression Trees) [30, 31]. In this study, we will use CART over Anaconda [28] for our IDS system. The relevant formulas are shown as follows.

For given dataset $T = \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$, where

$$x_i = \left(x_i^{(1)}, x_i^{(2)}, \ldots, x_i^{(n)}\right)^T, \quad i = 1, 2, \ldots, N, \tag{1}$$

x_i is the input instance and represents a network packet record. x_i has n features. N indicates the number of records of the packets contained in the dataset T. $y_i \in \{0, 1, 2, \ldots, K-1\}$ is the class tag which means the result of each detection record.

Let Q represent the data at node m, where X_m is the training data in node m.

For each split $\theta = (j, t_m)$ which consists of a feature j and a threshold t_m, the data is divided into two subsets of $Q_1(\theta)$ and $Q_2(\theta)$:

$$Q_1(\theta) = (x, y) \mid x_j \le t_m,$$
$$Q_2(\theta) = Q \setminus Q_1(\theta). \tag{2}$$

The impurity of m can be obtained by using an impurity function $H()$:

$$G(Q, \theta) = \frac{n_1}{N_m} H(Q_1(\theta)) + \frac{n_2}{N_m} H(Q_2(\theta)). \tag{3}$$

Select the parameter to minimize the impurity:

$$\theta^* = \arg\min_{\theta} G(Q, \theta). \tag{4}$$

Recourse for both $Q_1(\theta^*)$ and $Q_2(\theta^*)$ until N_m reaches the maximum depth and thus $N_m < \min_{\text{samples}}$ or $N_m = 1$.

For the classification of IDS, $y_i \in \{0, 1, 2, \ldots, K-1\}$ for node m represents a region of R_m with instances of N_m.

Assume that p_{mk} is the proportion of class k instance in m and can be obtained by the following formula:

$$p_{mk} = \frac{1}{N_m \sum_{xj \in R_m} I(y_i = k)}. \tag{5}$$

The measure of impurity is generally named as Gini and can be obtained by the following formula:

$$H(X_m) = -\sum_k p_{mk}(1 - p_{mk}). \tag{6}$$

Cross entropy can be obtained by the following formula:

$$H(X_m) = -\sum_k p_{mk} \log(p_{mk}). \tag{7}$$

Misclassification can be obtained by the following formula:

$$H(X_m) = 1 - \max(p_{mk}). \tag{8}$$

2.2. Evaluation Indicators. In this section, we mainly introduce the indicators of IDS.

(1) F1 Score. Assuming that we classify a sample dataset as both normal and abnormal, there are four cases of classification. As shown in Table 1, that is, the True Positive, False Positive, False Negative, and True Negative. True means that the classification is correct while False means that the classification is wrong. Positive means that the classifier is divided into normal (positive samples) and Negative means that the classifier is divided into abnormal (negative samples):

(1) True Positive: normal instance is detected correctly.

(2) False Positive: abnormal instance is incorrectly classified as normal.

(3) False Negative: normal instance is misclassified as abnormal one.

(4) True Negative: abnormal instance is detected correctly.

Table 1: Detection results.

	Relevant	Not relevant
Detected	True positives (TP)	False positives (FP)
Not detected	False negative (FN)	True negatives (TN)

Precision P represents the proportion of relevant instances among the detected instances. P can be obtained by the following formula:

$$P = \frac{\text{TP}}{\text{TP} + \text{FP}}. \tag{9}$$

Recall that R represents the proportion of relevant instances that have been detected over the total amount of relevant instances. R can be obtained by the following formula:

$$R = \frac{\text{TP}}{\text{TP} + \text{FN}}. \tag{10}$$

Actually, indicators of P and R are sometimes contradictory, and thus $F1$ score is the common evaluation indicator. $F1$ score is the weighted average of P and R which can be obtained by the following formula:

$$F1 \text{ Score} = \frac{(\alpha^2 + 1) P * R}{\alpha^2 (P + R)}. \tag{11}$$

More especially, where $\alpha = 1$, $F1$ score will get the new formula, and thus

$$F1 \text{ Score} = \frac{2PR}{P + R}. \tag{12}$$

(2) The Calculation Time. The calculation time t of the IDS detection algorithm. t contains the mode construction time and the detection time of proposed method.

3. A New IDS System for Fog Computing

In this section, a new IDS system for fog computing is presented. The main steps of this system are shown as follows. As shown in Figure 2, our proposed IDS system mainly consists of three steps: Step 1: the data preprocess; Step 2: data normalization; Step 3: decision tree detection. And the main work of each step is shown as follows.

Step 1 (data preprocess). The given dataset is usually composed of numbers and strings. We cannot compare the value of string directly, and thus we need to digitize the string by using string replace operation. The details are shown in Algorithm 1. We firstly traverse the given dataset D and find all the strings S in dataset D and obtain the corresponding columns c by using find () function (Line (1) to Line (3)). Secondly, we call the replace function to replace S with random number m (Line (7) to (9)). Finally the processed dataset D' is retuned. In addition, D' will be the input for Step 2.

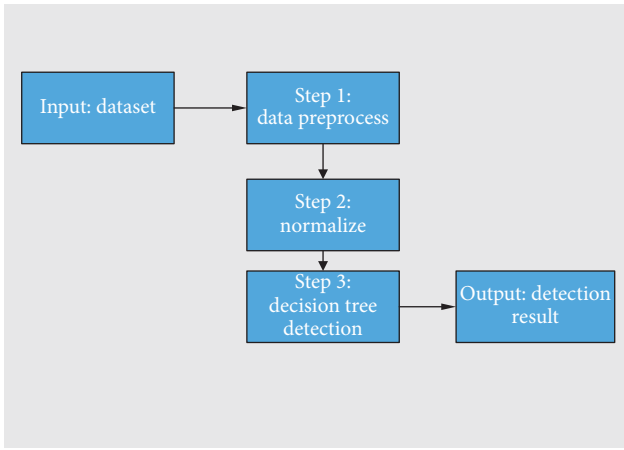


Figure 2: A new IDS detection system for fog computing.

Input: Dataset D
Output: processed Dataset D'
(1) **for** $n = 1$ to N do
(2) **for** $m = 1$ to N
(3) $(S, c) \leftarrow \text{find}(D)$
(4) **end for**
(5) **end for**
(6) **for** $n = 1$ to c do
(7) $D'(S) \leftarrow D(m)$
(8) **end for**
(9) **return** D'

Algorithm 1: String replace method.

Step 2 (data normalization). Notice that the range of numbers in D' may not uniform. That means large numbers of columns will cause the role of small columns to be ignored, and in fact there are some small numbers of columns that may play a very important role. And thus we should perform the normalization process before executing the detection algorithm; the object of normalization is to make the characteristic data shrink [0-1]. The main content is shown in Algorithm 2. Firstly, we randomly select $m\%$ of the training dataset D_1 from D' as the training dataset D_1 and the remaining D_2 equals $(1 - m\%)$ as the testing dataset (Line (1)). Then we obtain the normalization results X_train and X_test by using normalization function (Line (2) to Line (3)). Obviously, X_train and X_test will be the input for Algorithm 3.

Step 3 (decision tree detection method). In this step, we mainly construct the decision tree by using given training dataset X_train and then get the detection result of test dataset X_test. As shown in Algorithm 3, firstly, the decision tree mode mo is established by using CART function according to the related formula (2) to (8) illustrated in Section 2.1 (Line 1). Secondly, the labels in X_test are obtained by using the mode mo (Line 2). Last but not least, we obtain the results of $F1$ score and calculation time t by using statistics () function

Input: Dataset D'
Output: processed Dataset X_train, X_test
(1) $D_1, D_2 \leftarrow$ train_test_split $(D', m\%)$
(2) $X\_train \leftarrow$ normalization (D_1)
(3) $X\_test \leftarrow$ normalization (D_2)
(4) **return** X_train, X_test

Algorithm 2: Data normalization method.

Input: X_train, X_test
Output: $F1$ Score, calculation time t
(1) mo $\leftarrow$ CART(X_train)
(2) $Y2 \leftarrow$ mo(X_test)
(3) ($F1$ Score, t) $\leftarrow$ statistics()
(4) **return** $F1$ Score, t

Algorithm 3: Decision tree detection method.

according to the related formula (9) to (12) illustrated in Section 2.2 (Line 3).

4. Experimental Evaluation

4.1. Experimental Environment. In this section, we evaluate our proposed IDS system on KDDCUP99 dataset. The experiment is implemented by Python on a windows 10 Operating System, where the processor is Inter Core i7 2.7 GHZ, the RAM is 16 GB, and the main software platform is Eclipse and Anaconda 2.7 SCIkit-learn.

4.2. The Introduce of Dataset. For research of IDS, a large number of valid experimental data is needed. Data collection can be obtained through some capture tools, such as TCPdump, Libdump, and Wireshark, and then connection record is generated as the data source for IDS. In this study, we use KDDCUP99 [27] dataset for our test. The dataset is a 9-week network connection data collected from a simulated LAN of US Air Force. The dataset contains two kinds, the former one is 10% dataset named as KDDcup.Data.10.percent.correceted and the latter one is the full dataset named kddcup.data.corrected. Each connection record in KDDCUP99 dataset contains forty-one fixed feature attributes and a class label. Among the forty-one features, nine features are symbolic while the other ones are continuous. As shown in Table 2, the class identifier indicates that the connection record is normal or a specific kind of attack. In addition, we can see that the DOS, Probing, R2L, and U2R have a more detailed division. In this study, the attack kinds are marked as numbers. The corresponding marks are shown in Tables 3 and 4. On the one hand, we will complete the detection of four kinds of attacks; on the other hand, we will complete the detection of twenty-two kinds of attacks. Meanwhile, both the 10% dataset and the full dataset are tested in our experiment. And thus we will perform four group experiments for each method: (1) four kinds of attacks

Table 2: Class identifier of KDDCUP99.

Classification	Meaning	Subclass
Normal	Normal record	normal
DOS	Denial of service attack	Back land Neptune pod smurf teardrop
Probing	Monitoring and other detection activities	Ipsweep nmap portsweep satan
R2L	Illegal access from remote machines	ftp_write guess_passwd imap multihop phf spy warezclient arezmaster
U2R	Unauthorized access of ordinary users' to privileges of administrator	buffer_overflow loadmodule perl rootkit

Table 3: Four kinds of attack classification are marked.

Normal = 0	normal = 0
DOS = 1	back = 1 land = 1 neptune = 1 pod = 1 smurf = 1 teardrop = 1
Probing = 2	ipsweep = 2 nmap = 2 portsweep = 2 satan = 2
R2L = 3	ftp_write = 3 guess_passwd = 3 imap = 3 multihop = 3 phf = 3 spy = 3 warezclient = 3 warezmaster = 3
U2R = 4	buffer_overflow = 4 loadmodule = 4 perl = 4 rootkit = 4

over 10% dataset (2); twenty-two kinds of attacks over 10% dataset; (3) four kinds of attacks over full dataset; (4) twenty-two kinds of attacks over full dataset. Notice that the 10% dataset contains all twenty-two attacks; the full dataset does not contain attacks of No. 17, No. 18, and No. 20.

4.3. Experiment Result and Discussion. We compare the experiment results from the aspects of $F1$ Score and calculation time. In order to cover all the attack kinds and ensure the effectiveness of the test results, we randomly divided the dataset, 60% of which was used as a training dataset and 40% as a test dataset. As a result, the Naïve Bayesian contains three models: MultinomialNB, BernoulliNB, and GaussianNB [32]. And therefore, we firstly test Bayesian method and find the best one for IDS. And then compare it with the other two methods. For each method, we conduct 10 group experiments and then compare their average.

4.3.1. Experiment Result Contrast of Three Modes of Bayesian. Firstly, we test the Bayesian method. The calculation time contrast results are shown in Figure 3. MultinomialNB gets the least calculation time among all the test cases, followed by BernoulliNB, and GaussianNB is the last one. And then we compare the results according to the test results of $F1$ score. Our principle is shown as follows. We firstly see the detection precision of normal class, as in the actual situation, the proportion of normal class is relatively large, and then

Table 4: Twenty-two kinds of attack classification are marked.

normal = 0
buffer_overflow = 1
pod = 2
teardrop = 3
guess_passwd = 4
portsweep = 5
ipsweep = 6
land = 7
back = 8
neptune = 9
smurf = 10
teard = 11
satan = 12
ftp_write = 13
imap = 14
multihop = 15
phf = 16
spy = 17
warezclient = 18
warezmaster = 19
loadmodule = 20
perl = 21
rootkit = 22

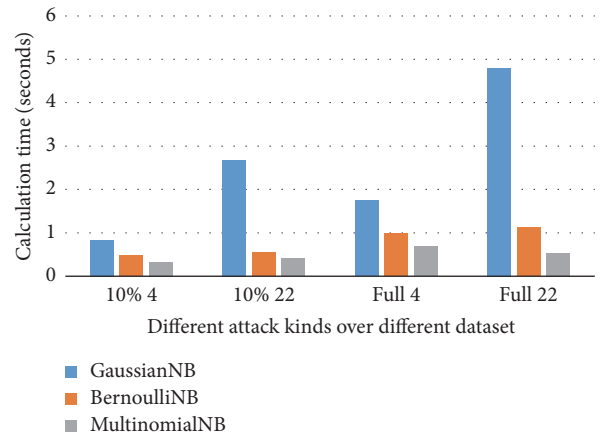


Figure 3: The calculation time contrast of three modes of Bayesian.

see the detection coverage of all attacks. As the attack type is divided into four kinds and twenty-two kinds, so we firstly discuss the detection result of three methods on four kinds of attacks and then discuss the detection result on twenty-two kinds of attacks.

(1) As shown in Figure 4, for 10% dataset, the detection precision on GaussianNB for the normal type is significantly lower than the other two methods. The BernoulliNB method is slightly lower than the MultinomialNB method for the normal type detection. As shown in Figure 5, for full dataset, detection $F1$ Score based on GaussianNB for the normal type has increased, but still lower than the other two. In

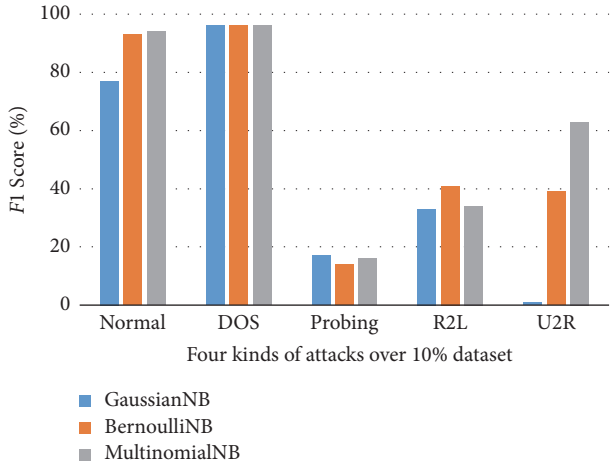


Figure 4: $F1$ score contrast of four kinds of attacks over 10% dataset of three modes.

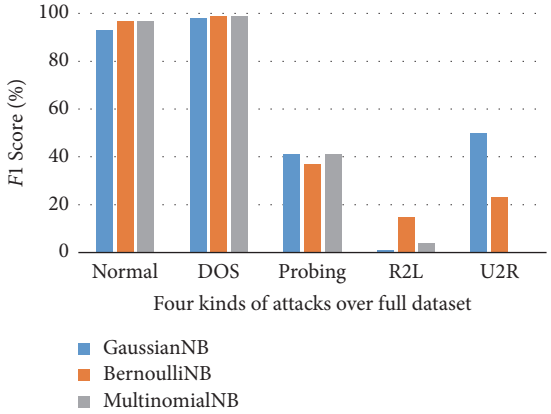


Figure 5: $F1$ score contrast of four kinds of attacks over full dataset of three modes.

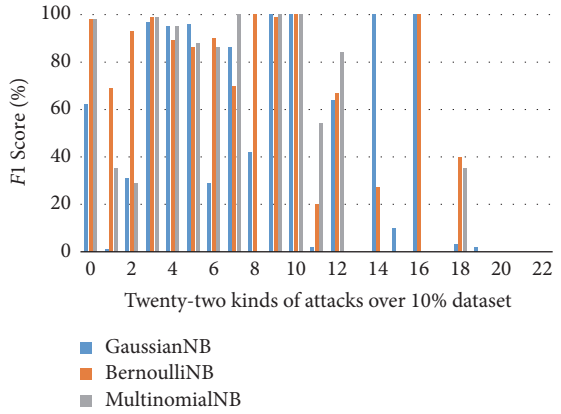


Figure 6: $F1$ score contrast of twenty-two kinds of attacks over 10% dataset of three modes.

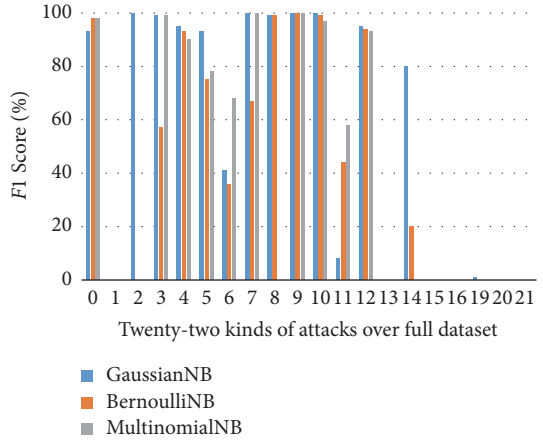


Figure 7: $F1$ score contrast of twenty-two kinds of attacks over full dataset of three modes.

addition, GaussianNB and BernoulliNB can do the detection type coverage. $F1$ Score of U2R based on MultinomialNB is 0%. However BernoulliNB is relatively stable. Although the $F1$ Score of U2R detection by GaussianNB is better than BernoulliNB, the detection $F1$ Score of R2L is much lower than BernoulliNB's, meanwhile, considering $F1$ Score on normal type by GaussianNB is lower than BernoulliNB method. In addition, the calculation time of the former one is much longer than the latter one. And thus, the BernoulliNB method is the best method for IDS.

(2) Next, we discuss the results of the three modes for detecting twenty-two attacks over both datasets. Similarly, we first discuss the normal class of test results. As shown in Figure 6, the same as a result in the above scenario, for 10% dataset, the detection $F1$ Score based on GaussianNB for the normal type is significantly lower than the other two methods. The BernoulliNB method obtains the same precision for normal type detection with the MultinomialNB method. In addition, in view of the detection $F1$ Score of twenty-two kinds of attacks, the BernoulliNB method is the best. As shown in Figure 7, for the full dataset test, the $F1$ Score of the normal class by GaussianNB has improved but is still lower than the other two methods. The analytical method is the same as above. Considering the $F1$ Score of detection for other attacks, the Bernoulli method is the best.

And thus, among the three modes of Bayesian, the BernoulliNB model is the most suitable one for IDS. Next, we will compare it with the other two methods in the next experiment.

4.3.2. Experiment Results Contrast of Three Methods. Next, we will compare BernoulliNB with decision tree and KNN. The calculation time contrast results are shown in Table 5. BernoulliNB gets the least calculation time among all the test cases, followed by decision tree, and KNN is the last one.

As shown in the Table 5, the calculation time of KNN cannot be accepted. Since KNN has the worst performance in time, there is no need for multiple experiments. The $F1$ Score results of the KNN experiment corresponding to the time in the Table 5 are shown in Table 6. KNN method over full dataset is not very good and cannot detect number 4 (U2R) attack.

Table 5: The calculation time contrast for three methods.

Method	Group			
	4 kinds 10%	22 kinds 10%	4 kinds full	22 kinds full
Decision tree	1.455921545 s	1.173570469 s	3.319417967 s	3.143889363 s
BernoulliNB	0.472016006 s	0.552608763 s	0.986687026 s	1.116946133 s
KNN	1962.25 s	/	7372.6833 s	/

Table 6: $F1$ Score of KNN method.

Dataset	Number				
	0	1	2	3	4
10% dataset	100%	100%	99%	94%	52%
Full dataset	100%	100%	100%	85%	0%

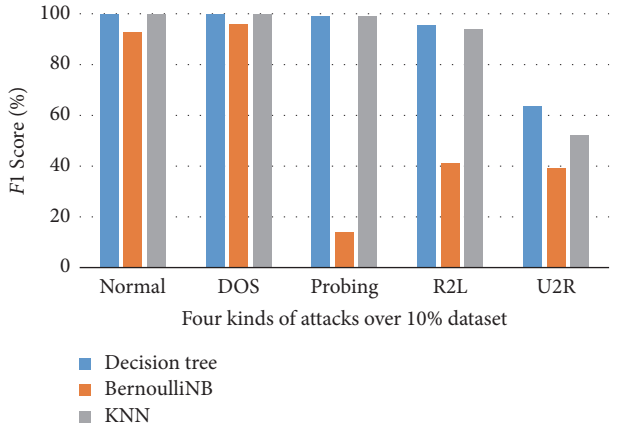


Figure 8: $F1$ score contrast of four kinds of attacks over 10% dataset of three methods.

(1) We first discuss detection results of the two methods of four kinds of attacks and then discuss the situation of twenty-two kinds of attacks. As shown in Figure 8, for each 10% dataset, the $F1$ Score of all attacks based on the decision tree is higher than the BernoulliNB method; as shown in Figure 9, for full dataset detection, all attack detection $F1$ Score on decision tree is higher than BernoulliNB except U2R.

(2) Next, we discuss the results of twenty-two attacks of the two methods over both datasets. Similarly, we firstly discuss results of the normal class. As shown in Figure 10, the decision tree obtains the same precision with BernoulliNB on No. 8 attack, No. 10 attack, and No. 16 attack. The precision of other attacks on decision tree is much higher than BernoulliNB. In particular, BernoulliNB cannot detect No. 13 attack, No. 15 attack, No. 19 attack, No. 21 attack, and No. 22 attack while decision tree methods can do it. As shown in Figure 11, for full dataset, the decision tree method obtains the same $F1$ Score with BernoulliNB on No. 9 attack. In addition, the $F1$ Score of BernoulliNB method is slightly lower than BernoulliNB for No. 4 attack. Moreover, the decision tree method is better than BernoulliNB in all other cases. In addition, the calculation time of the former one is much longer than the latter one. Above all, decision tree

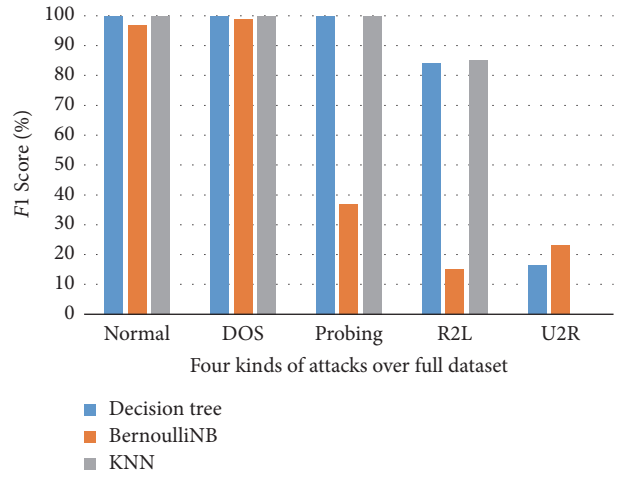


Figure 9: $F1$ score contrast of four kinds of attacks over full dataset of three methods.

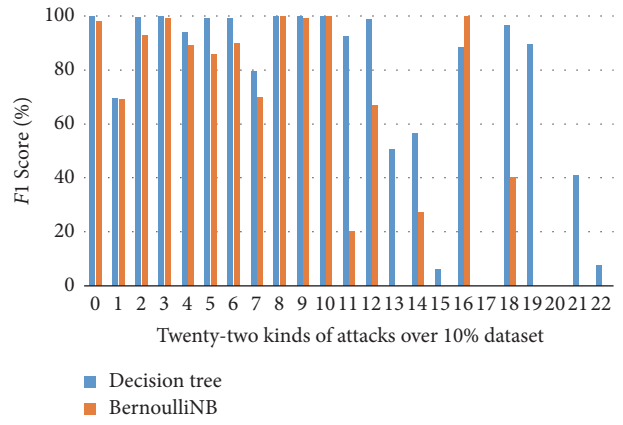


Figure 10: $F1$ score contrast of twenty-two kinds of attacks over 10% dataset.

method is the most suitable one for IDS over big data in fog environment.

4.3.3. Discussion and Performance Analysis. Among the three models of the Bayesian method, BernoulliNB tends to be the best one from the perspective of $F1$ Score. The overall $F1$ Score of the decision tree is the best. From the perspective of the calculation time, the BernoulliNB is the best. The KNN is not suitable for the large dimension although the precision is very high over 10% dataset. For IDS issue, if only the precision is considered, the decision tree algorithm is the best choice; if only the calculation time is considered, the BernoulliNB

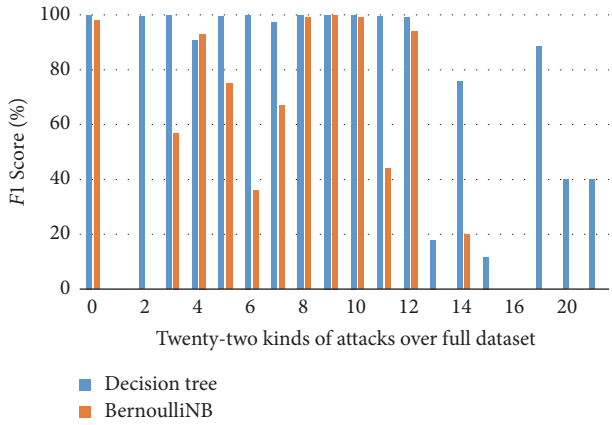


Figure 11: $F1$ score contrast of twenty-two kinds of attacks over full dataset.

algorithm is much better. From the point of view of the calculation time, although is not the best, the calculation time of the decision tree is acceptable. The authors in [24] point out that the calculation time of Naïve Bayesian is generally 7 times faster than that of decision trees by using C4.5. However, in this study, we can conclude that the decision tree based on CART is much faster. The multiple comparison of calculation time is shown as Table 7.

(1) BernoulliNB is 2.364 times faster than decision tree in the case of four kinds of attacks over full dataset. In particular, the time gap is narrowed over the situation of twenty-two kinds of attacks.

(2) In order to make the comparison more comprehensive, we simply look at it with the other two Bayesian cases. Compared with GaussianNB, the decision tree is much faster than GaussianNB over the situation of twenty-two kinds of attacks, even when compared with BernoulliNB which is the most time efficiency mode of Naïve Bayesian, MultinomialNB is only 4.857 times faster than decision tree in the worst situation.

However, taking into account the detection accuracy, as well as the coverage of the attacks, there is no doubt that the decision tree is the best choice for IDS over big data in fog computing.

Above all, our proposed IDS system is efficient and precise. As shown in Figure 1, our proposed system can be deployed in a common node of fog layer without extra requirement. According to the above experiment, we can conclude that the system performance is stable and performs very well in big data environment.

5. Related Work

Fog computing was for the first time proposed by Cisco in 2012 and defined as a highly virtualized computing platform for migrating the tasks of Cloud to network mobile users. The fog computing [4] introduces the middle layer between the cloud and the mobile users, extending the cloud-based network structure, and provides computing, storage, as well as network service between mobile devices and Cloud. The fog computing reduces unnecessary multiple communication between the cloud computing center and the mobile users [8]. It not only reduces the network delay for mobile users but also significantly reduces the link bandwidth backbone [9, 10]. Although there are many advantages of fog computing, some security issues still need to be solved. More specifically, fog computing nodes are usually composed of weak computing power. Traditional network attacks become more common in fog computing environment, such as eavesdrop or hijack the mobile user data and even attempt to destroy the fog system. Fortunately, Intrusion Detection Systems (IDS) can also be applied in fog environment [11].

After decades of development, IDS has become a more successful security technology. IDS which represented by Snort [33] has made an outstanding contribution to network security in recent years. ISS RealSecure is also well known, and it mainly consists of two parts, the engine part and the console part. The former one is responsible for detecting information and generate alarms and the latter one receives the alarm and is a central point for configuring and generating the database report. Pattern matching algorithm is one of the core technologies of IDS products. Misuse detection based on AC, BM, MWM, and other matching algorithms [15] can make IDS have a passive detection of known attacks with wide and obvious characteristics. However, modern attacks are increasingly inclined to form an unknown intrusion technology by integrating a variety of known intrusion technologies. Meanwhile, improved IDS methods usually take proactive protection based on deviation detection and user behavior anomaly detection. Statistical model, Bayesian reasoning, cluster analysis [16], and other excellent algorithms like DB can make up for the lack of pattern matching. KNN algorithm known as k nearest neighbor algorithm [17] is widely used in pattern recognition, classification, and regression [18]. Same as KNN, vector automatic classification algorithms, support vector machine [19, 20], neural network algorithm [21], Bayesian algorithm [22–24], and K means algorithm are also widely used for IDS [25, 26].

Although the IDS in tradition network has been well investigated, unfortunately directly use them in fog computing environment may not inappropriate. More specifically, the existing researches mainly present the experiments on 10% KDDCUP99 dataset. Although these methods have achieved good results, we cannot judge their efficiency when they are presented in big data environment, even in the full dataset of KDDCUP99. In addition, there are four kinds of attacks classification, as well as twenty-two attacks classification in KDDCUP99. However, the existing researchers mainly focus on the detection of four attacks but fail to consider the detection of twenty-two attacks. In order to address the aforementioned problem, we propose an IDS system based on Anaconda, we use decision tree for our IDS detection, and multimethods are compared. Although the author in [24] also uses Bayesian and decision tree methods for IDS. Different from them, we conducted a more adequate experiment. And we compare decision tree with three modes of Naïve Bayesian method, as well as KNN method. More specifically, both the 10% dataset and the full dataset are tested in our IDS system. We not only complete the detection of

Table 7: The multiple comparison of calculation time.

Method	Group			
	4 kinds 10%	22 kinds 10%	4 kinds full	22 kinds full
Decision Tree|GaussianNB	1.757	0.438	1.894	0.655
Decision Tree|BernoulliNB	3.084	2.124	3.364	2.815
Decision Tree|MultinomialNB	4.393	2.882	4.837	5.857

four kinds of attacks but also accomplish the detection of twenty-two kinds of attacks. In addition, the calculation time of each method is compared. The authors in [20] also consider the calculation time of their algorithm; however, they also only present their experiments on 10% dataset, and thus we cannot judge the performance of the algorithm over big data environment. Above all, the experiment results show that our proposed system is effective and precise.

6. Conclusion

Tradition network attacks are widely present in fog computing environment. Although the IDS in tradition network have been well investigated, unfortunately directly use of them in fog computing environment may not inappropriate. In this study, we propose a system based on the decision tree, multimethods are compared with this one, not only the 10% dataset but also the full dataset is tested, and the experiment results show that our system is effective. In addition, we also compared the detection time for each method. In the case of guaranteed accuracy, although the decision tree time is not the best one, the calculation time is also acceptable. Above all, our IDS system can be used in fog computing environment over big data. In our future, we will engage in the research of the IDS for other kinds of attacks.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work is supported by Quanzhou Science and Technology Project (no. 2015Z115), the Scientific Research Foundation of Huaqiao University (no. 14BS316), the Education and Scientific Research Projects of Young and Middle-aged Teachers in Fujian Province (JZ160084), College Students Innovation and Entrepreneurship Project (201710385023), and China Scholarship Council award to Dr. Kai Peng for one year's research abroad at the University of British Columbia.

References

[1] 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 '12*, pp. 13–15, ACM, Helsinki, Finland, August 2012.

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

[3] 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.

[4] T. H. Luan, L. Gao, Z. Li, Y. Xiang, G. Wei, and L. Sun, "Fog computing: focusing on mobile users at the edge," 2015, https://arxiv.org/abs/1502.01815.

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

[6] X. L. 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, Article ID 2376484, 15 pages, 2017.

[7] I. Stojmenovic, S. Wen, X. Huang, and H. Luan, "An overview of Fog computing and its security issues," *Concurrency Computation*, vol. 28, no. 10, pp. 2991–3005, 2016.

[8] G. M. Relations, "Greyhound launches "Blue ?" An exclusive WiFi enabled onboard entertainment system," https://www.greyhound.com/en/about/media/2013/07-08-2013.

[9] W. S. Shi, H. Sun, J. Cao, Q. Zhang, and W. Liu, "Edge computingan emerging computing model for the internet of everything," *Journal of Computer Research and Development*, vol. 54, no. 5, 2017.

[10] Y. Fan and Q. Zhu, "Cloud/Fog computing system architecture and key technologies for south-north water transfer project safety," *Wireless Communications and Mobile Computing*, 2017.

[11] C. Modi, D. Patel, B. Borisaniya, H. Patel, A. Patel, and M. Rajarajan, "A survey of intrusion detection techniques in cloud," *Journal of Network and Computer Applications*, vol. 36, no. 1, pp. 42–57, 2013.

[12] A. Milenkoski, M. Vieira, S. Kounev, A. Avritzer, and B. D. Payne, "Evaluating computer intrusion detection systems: a survey of common practices," *ACM Computing Surveys*, vol. 48, no. 1, pp. 1–41, 2015.

[13] J. P. Anderson, "Computer security threat monitoring and surveillance," 1980.

[14] D. E. Denning, "An intrusion-detection model," *IEEE Transactions on Software Engineering*, vol. SE-13, no. 2, pp. 222–232, 1987.

[15] Y. Li, H. Li, X. Qian, and Y. Zhu, "A review and analysis of outlier detection algorithms," *Analysis of Outlier Detection Algorithms*, vol. 6, p. 5, 2002.

[16] T. Hastie and R. Tibshirani, "Discriminant adaptive nearest neighbor classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 6, pp. 607–616, 1996.

[17] J. Lu, Z. Wu, Y. Wang, and Y. Lu, "Research on abnormal behavior detection based on kNN algorithm," *Computer Engineering*, vol. 7, p. 48, 2007.

[18] K. Li, J. Zhao, H. Hong, S. Tian, and J. Zhao, "One-class support vector machine model for intrusion detection," *China Safety Science Journal*, vol. 13, no. 6, pp. 72–76, 2003.

[19] H. R. Zhang and Z. Z. Han, "An improved sequential minimal optimization learning algorithm for regression support vector machine," *Journal of Software. Ruanjian Xuebao*, vol. 14, no. 12, pp. 2006–2013, 2003.

[20] W. Zhang and J. Fan, "Cloud architecture intrusion detection system based on KKT condition and hyper-sphere incremental SVM algorithm," *Journal of Computer Applications*, vol. 35, no. 10, pp. 2886–2890, 2015.

[21] T. H. Dai, "Intrusive detection based on genetic neural networks," *China Safety Science Journal*, vol. 16, no. 2, pp. 103–108, 2006.

[22] X. L. Shao, Y. W. Liu, M. J. Geng, and J. B. Han, "The parallel Implementation of MapReduce for the Bayesian Algorithm to detect botnets," *CAAI Transactions on Intelligent System*, vol. 1, pp. 26–33, 2014.

[23] S. Wang, H. Zou, Q. Sun, and F. Yang, "Bayesian approach with maximum entropy principle for trusted quality of web service metric in e-commerce applications," *Security and Communication Networks*, vol. 5, no. 10, pp. 1112–1120, 2012.

[24] N. B. Amor, S. Benferhat, and Z. Elouedi, "Naive Bayes vs decision trees in intrusion detection systems," in *Proceedings of the 2004 ACM symposium on applied computing (SAC '04)*, pp. 420–424, March 2004.

[25] H. C. Liu, X. N. Hou, and Z. Yang, "Design of intrusion detection system based on improved K-means algorithm," *Computer Technology and Development*, vol. 1, pp. 101–105, 2016.

[26] W. L. Al-Yaseen, Z. A. Othman, and M. Z. A. Nazri, "Multi-level hybrid support vector machine and extreme learning machine based on modified K-means for intrusion detection system," *Expert Systems with Applications*, vol. 67, pp. 296–303, 2017.

[27] http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.

[28] Scikit-learn, http://scikit-learn.org/stable/index.html.

[29] J. R. Quinlan, *C4.5, Programs for machine learning*, Morgan Kaufmann, San Mateo Ca, 1993.

[30] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees*, Wadsworth and Brooks, California, Calif, USA, 1984.

[31] H. Li. Statistical learning, *method*, Tsinghua University Press, Beijing, 2012.

[32] C. D. Manning, P. Raghavan, and H. Schuetze, *Introduction to Information Retrieval*, Cambridge University Press, Cambridge, UK, 2008.

[33] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21–27, 1967.