Optimal Edge Resource Allocation in IoT-Based Smart Cities

Lei Zhao, Jiadai Wang, Jiajia Liu, and Nei Kato

Abstract

With IoT-based smart cities, massive heterogeneous IoT devices are running diverse advanced services for unprecedented intelligence and efficiency in various domains of city life. Given the exponentially growing number of IoT devices and the large number of smart city services as well as their different QoS requirements, it has been a big challenge for servers to optimally allocate limited resources to all hosted applications for satisfactory performance. Note that by pushing the computing and storage resources to the proximity of end IoT devices, and deploying applications in distributed edge servers, edge computing technology appears to be a promising solution for this challenge. Toward this, we study in this article how to allocate edge resources for average service response time minimization. Besides the proposed algorithms, extensive numerical results are also presented to validate their efficacy.

Introduction

The flourishing Internet of Things (IoT) technology is bringing numerous emerging concepts into reality, among which a striking one is the smart city [1]. With IoT-based smart cities, massive heterogeneous IoT devices run diverse advanced services for unprecedented intelligence and efficiency in various domains of city life [2]. Namely, smart building, intelligent transportation, ubiquitous e-healthcare, smart home, smart energy, and smart factory are going to significantly improve the quality of life of residents in IoT-based smart cities [3].

IoT-based smart cities are characterized by a large number of services running over massive end IoT devices as well as applications hosted in remote servers. Note that these services, varying from virtual reality to precision healthcare, from video-based surveillance to water network monitoring, may have totally different quality of service (QoS) requirements. For example, virtual reality has a very stringent limit on tolerated delay [4]. What's more, the number of IoT devices in smart cities is exponentially growing with disparate characteristics. Stationary IoT devices, as deployed in different geo-locations, may be equipped with powerful processors and huge storage space [5]. On the contrary, the battery life, processing power, and storage capacity of highly mobile IoT devices (e.g., smart wearables [6]) may be very limited.

Given the massive amount of IoT devices and the large number of smart city services as well as their different QoS requirements, it has been a big challenge for servers to optimally allocate limited resources to all hosted applications for satisfactory performance. This issue becomes even more severe as the IoT-based smart city scales up. Obviously, running all applications on the central cloud may easily lead to serious network congestion and performance degradation. Note that by pushing the computing and storage resources to the proximity of end IoT devices, and deploying applications in distributed edge servers, edge computing technology appears to be a promising solution for this challenge.

There have been some pioneering research works toward edge computing in IoT-based smart cities. For example, Zhou *et al.* designed an optimal computing manner for an industrial IoT system based on analyzing the relationship between data processing and energy consumption [7]. Some other works have also been able to provide precious insights into IoT data analytics [8] and content delivery [9]. However, these works have some common limitations. They ignored the huge differences among the resource demands from various applications hosted in edge servers, the huge disparities within the characteristics of detailed service requests from numerous IoT devices, as well as their impacts on edge resource allocation. To the best of our knowledge, edge resource allocation for multiple applications in IoT-based smart cities has rarely been touched upon in the literature. Toward this, in this article we study how to allocate edge resources for average service response time minimization while running multiple smart city services over a large number of IoT devices and satisfying the capacity constraints of edge servers.

The remainder of this article is organized as follows. We present an overview of the networking architecture for IoT-based smart cities. The problem of average service response time minimization (SRTM) with two promising solutions is presented. Then we present experimental results and conclude the whole article.

Overview of the Networking Architecture for IoT-Based Smart Cities

Promising IoT-Based Smart Cities

Smart cities have been closely linked to the IoT technology, in which a large number of interconnected devices are important tools for measurement and data collection. As shown in Fig. 1, smart cities include smart building, smart transportation, smart healthcare, smart home, smart energy, and smart factory. The following discussions illustrate these scenarios.

Digital Object Identifier:
10.1109/MNET.2019.1800221

Lei Zhao, Jiadai Wang, and Jiajia Liu are with Xidian University; Nei Kato is with Tohoku University.

0890-8044/19/$25.00 © 2019 IEEE

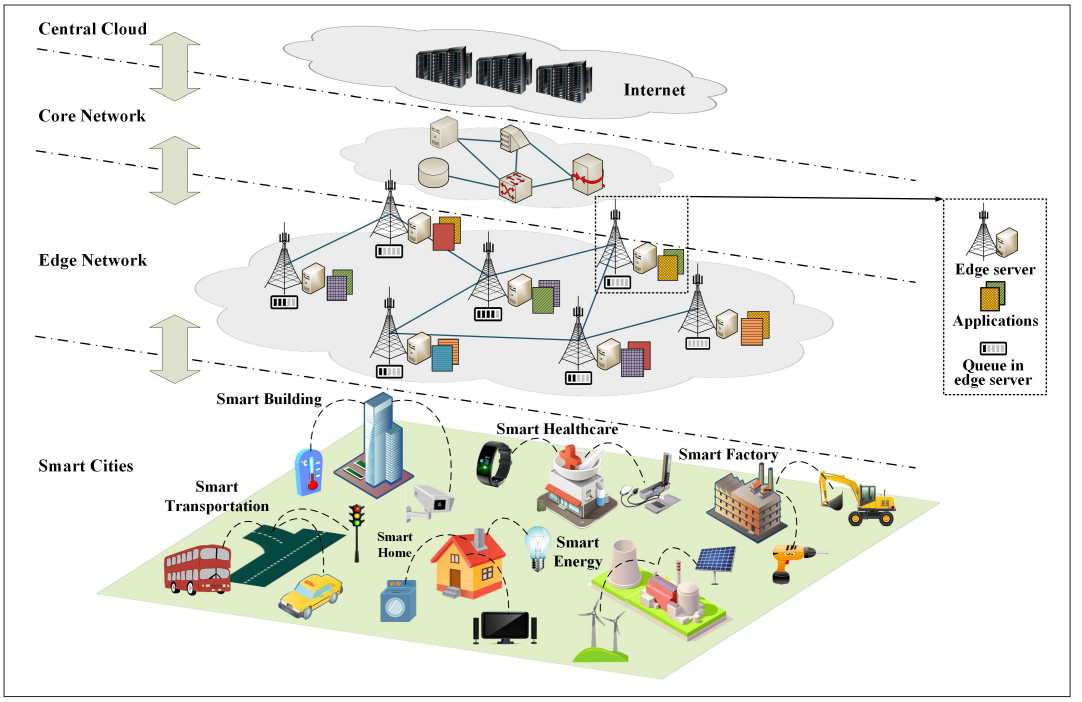

FIGURE 1. The networking architecture for IoT-based smart cities.

Smart Building: A smart building can not only reduce the consumption of unnecessary energy, but also improve the comfort of the occupants. For example, data provided by temperature and humidity sensors can be used to dynamically adjust the internal environment; illumination information can be used to regulate the interior lighting system; and crowd perception and identification are able to be carried out through closed-circuit television (CCTV) cameras.

Smart Transportation: As a fundamental aspect of smart cities, smart transportation can use traffic networks more safely and efficiently. For instance, pneumatic tubes, inductive loops, and cameras can collect data for vehicle detection and road surveillance; and data from carbon dioxide sensors can be used for monitoring vehicle emissions.

Smart Healthcare: People's neglect of health and socially limited medical resources lead to a very urgent need for smart healthcare. Wearable devices carried with patients, such as smart bracelets and blood pressure and sugar detectors, are able to monitor a variety of human health indicators, while doctors can collect real-time raw data through these IoT smart devices. Therefore, doctors and patients will be greatly facilitated with reduced medical cost and enriched experience.

Smart Home: On one hand, IoT devices assembled at home can monitor and perceive the home environment, optimize the energy consumption in the house, and improve the comfort and convenience of people's lives. On the other hand, by networking home devices and appliances, people can use remote access to activate or control part of the smart home [10].

Smart Energy: Intelligent management of energy allocation and consumption such as smart metering, smart lighting, and smart grid has lots of benefits for improving energy efficiency and saving energy cost. Moreover, it can definitely promote the utilization of renewable energy sources (e.g., solar energy and wind energy).

Smart Factory: In order to meet the increasing production demands of the market, smart factory, which can realize machine scheduling and rational assignment of tasks, emerges as the times require. For example, industrial clouds, as proposed in the fourth industrial revolution, can organize a set of geographically distributed factories, flexibly adjust production capacity, and share resources and assets [11].

Edge Resource Allocation for Multiple Applications

Massive data generated by multiple services running on numerous IoT devices needs further processing by applications hosted in remote servers to achieve rich intelligent functions. At this point, the edge network migrates computing and storage capabilities closer to the end IoT devices, and has distributed architecture to balance network traffic. Therefore, it can improve the QoS of the smart city services and avoid traffic peaks. Subsequently, the reasonable allocation of limited edge resources to numerous applications with diverse characteristics and requirements is particularly important. The following discussions give several typical requirements of some IoT applications for edge resource allocation.

Computing Capacity: Applications supporting services that need image and video processing, such as crowd perception and recognition, require superb computing capacity to run various machine learning algorithms and perform complex processing. The services may not have extremely low latency constraints; thus, distant idle edge servers can allocate resources to these applications.

Frequent Interactions: Some services require frequent interaction between IoT devices and applications hosted in servers. For example, smart transportation has the responsibility of managing

and controlling road conditions; therefore, frequent road information interaction with the correlative applications is required. Smart wearable devices monitoring health status should upload and download data in real time. Most renewable energy resources are changeable due to the varying environment, so it is necessary to monitor the enviroment in smart grid and optimally adjust the power generating equipment in real time based on frequent data interactions. Therefore, edge servers with rich communication resource can greatly reduce the communication burden from the frequent interaction requirement of these applications.

Strict Latency Constraint: Lots of smart city services require strict latency constraints. For instance, the scheduling of precision processes in factories is time-critical; timely warning of traffic accidents is extremely sensitive to delay. Consequently, the corresponding applications need to be deployed on edge servers close to the IoT devices for reducing service response delay.

To sum up, in order to achieve satisfactory performance of disparate smart city services, it is necessary to optimally allocate the limited edge resources for applications as needed.

Edge Resource Allocation for Average Service Response Time Minimization

In this section, we first define the problem of minimizing the average service response time by optimal edge resource allocation for applications. Then, we propose the Enumeration-Based Optimal Edge Resource Allocation (EOERA) algorithm and the Clustering-Based Heuristic Edge Resource Allocation (CHERA) algorithm as our schemes.

Problem Definition

We consider a scenario where plentiful edge servers allocate resources to multiple applications for supporting IoT devices running various smart city services. As illustrated earlier, an increasing number of IoT-based smart city services have strict latency constraints, consequently, response time will be critical for the usability of these services. Therefore, we present several analyses on average service response time minimization by optimally allocating edge resources for multiple applications. From Fig. 1, it can be observed clearly that the service response time mainly consists of three parts: the data transmission delay in edge networks, the service processing delay in edge servers, and the network delay to the central cloud for excess requests.

The data transmission delay in edge networks comprises the one-hop wireless transmission delay for uploading requests of IoT devices to their nearby edge servers, and the routing delay of transmitting requests for application $m \in M$ among edge servers. However, the one-hop wireless transmission delay for each IoT device is only determined by the communication environment of the nearby edge server's service region, which is not affected by edge resource allocation schemes. Thus, the influence of the one-hop wireless transmission delay on the estimation of edge resource allocation schemes can be ignored for the sake of simplification.

When all service requests from IoT devices arrive at the nearest edge servers hosting corresponding applications (e.g., $\forall m \in M$), the edge servers assign d_m, $\forall m \in M$ computing resources for each request. The limited capacity of each edge server, $c \in C$, to process service requests is also taken into consideration. A variety of IoT devices send service requests to edge servers hosting application $m \in M$ at various generation rates [12]. Although one edge server may comprise a number of interconnected physical machines to process the incoming service requests, we consider each edge server as an abstract unit to handle these assigned requests. As a result, it is appropriate to model the processing of service requests in each edge server as a queueing model.

When edge servers get overloaded, the service provision mode will switch to traditional cloud computing. Subsequently, excess service requests will be routed to the central cloud via the core network and the Internet. Additionally, we define the average network delay to the central cloud for application $\forall m \in M$ as B_m.

Average Service Response Time Minimization (SRTM): Given the initial information (e.g., the capacity of each edge server), the resource demand of each application, and the topology of the edge network, the SRTM problem is to optimally allocate edge resources for multiple applications. Its resource allocation decision procedure aims to find a feasible solution for request assignment among edge servers hosting corresponding applications so as to minimize the average service response time satisfying the given capacity constraint of edge servers. To conduct this procedure, it needs to partition the resource of each edge server into resource blocks according to the demands of different applications, and allocate these partitioned resource blocks to the corresponding application requests. This resource allocation decision procedure makes the summation of the allocated resource blocks of each edge server be the same to provide services in a balanced way. It can be transformed into an optimization version of the partition problem, which is a well-known NP-complete problem [13]. Thus, the decision procedure of SRTM is NP-complete, and SRTM is NP-hard.

Note that we take the data transmission delay in edge networks and the service request processing delay in edge servers as the metrics for the fitness of edge servers that allocate resources for corresponding applications. Requests from IoT devices will first be assigned to the most suitable edge server. If resources in edge networks run out, excess service requests will be offloaded to the central cloud. Compared to the long network delay to access the central cloud, the processing delay in the powerful cloud can be ignored.

Solutions

In the following, we present two schemes to solve the SRTM problem in IoT-based smart cities. To simplify our algorithm presentation, the main notations are predefined as follows. $G(V, E)$ represents the logical topology of the edge network in IoT-based smart cities, where V and E separately denote the sets of edge servers and the communication links in the edge network; m denotes the set of different applications and C is the set of

the resource demands of these applications; the set of IoT devices is represented by N; α is the set of average service requests rate for all applications; and finally, k is the number of edge servers needed to allocate resources for each application.

Enumeration-Based Optimal Edge Resource Allocation Algorithm: As described in Algorithm 1, an enumeration optimal solution to SRTM mainly contains three steps. First, it enumerates all possible edge resource allocation combinations for multiple applications. Then it calculates the average service response time of each resource allocation case. Finally, an optimal combination of resource allocation for multiple applications can easily be found by comparing the calculated results.

For each resource allocation case, we consider assigning service requests from IoT devices to edge servers in a balanced way. This procedure delivers these requests to the closest edge server hosting the correlative applications and possessing unoccupied resources. At the same time, it also takes the load balancing among each feasible edge server into consideration by assigning requests to available edge servers in turn. What's more, it also decides when and how many service requests should be transmitted to the central cloud based on the current edge network states.

EOERA traverses the whole solution space by enumerating all combinations of edge resource allocation cases for $|M|$ applications and evaluating the average service response time for all these cases. It is obvious that EOERA can find an optimal edge resource allocation case for all applications to get the minimum average service response time.

Require $G(V, E), k, N, C, M, \alpha$
Ensure D_{min}
1: Initialize S denotes the set of all resource allocation cases, $S \leftarrow \Phi$
2: Enumerate all resource allocation combinations for $\forall m \in M$ among edge server set V, and record into set S
3: $D_{min} \leftarrow +\infty$
4: **for** $A \subseteq S$ **do**
5: Evenly assign requests among edge servers and the central cloud based on resource allocation case A
6: Evaluate the average service response time $D_{avg}(M, k)$ for resource allocation case A
7: if $D_{avg}(M, k) \leq D_{min}$ **then**
8: $D_{min} \leftarrow D_{avg}(M, k)$
9: Record the current optimal resource allocation case A_{opt}
10: **end if**
11: **end for**
12: **return** D_{min}, A_{opt}

ALGORITHM 1. Enumeration-Based Optimal Edge Resource Allocation Algorithm (EOERA).

Clustering-Based Heuristic Edge Resource Allocation Algorithm: In order to solve the realistic-scale SRTM problem within a reasonable time range, we propose CHERA, acquiring near-optimal performance with much lower computation complexity than EOERA. CHERA mainly consists of three steps:

1. Finding the resource allocation case for each application heuristically based on the current average service response time
2. Initializing clusters of candidate edge servers for each application based on the result of the heuristic procedure
3. Replacing the current edge server that allocates resources for application $m \in M$ in each cluster heuristically

First, CHERA assumes that all edge servers are no-load and are candidates to allocate resources for each application. Traversing overall candidate edge servers for each application, CHERA selects the edge server that can obtain minimal average service response time under the current edge network situations to allocate resources for the application. Then CHERA updates the load situation of the selected edge server as well as the candidate edge server sets. After processing the candidate edge server selection for all applications iteratively, a heuristic resource allocation result can be acquired.

Second, CHERA adopts a clustering procedure to initialize k clusters (i.e., Y_m) of candidate edge servers for application $m \in M$. In order to cover the edge network evenly, CHERA generates cluster $Y_m^i \in Y_m$ for application $m \in M$ by absorbing candidate edge servers around the selected edge server, which currently allocates resources for

Require $G(V, E), k, N, C, M, \alpha$
Ensure D_{min}
1: Initialize the resource allocation case A of $|M|$ applications heuristically based on the current average service response time
2: $D_{min} \leftarrow +\infty$
3: **for** $m \in M$ **do**
4: Initialize clusters Y_m of candidate edge servers for application m based on A_m
5: **for** $i \leftarrow 1$ *to* k **do**
6: **for** $v \in Y_m^i$ **do**
7: $\tilde{A} \leftarrow A$
8: $\tilde{A}_m^i \leftarrow v$
9: Evaluate the average response time $\tilde{D}$ for resource allocation case $\tilde{A}$
10: if $\tilde{D} < D_{min}$ **then**
11: $D_{min} \leftarrow \tilde{D}$
12: $\hat{A} \leftarrow \tilde{A}$
13: **end if**
14: **end for**
15: Update $A \leftarrow \hat{A}$
16: **end for**
17: **end for**
18: $A_{opt} \leftarrow A$
19: Evenly assign requests among edge servers and the central cloud based on resource allocation case A_{opt}
20: Evaluate the average response time D_{min} for resource allocation case A_{opt}
21: **return** D_{min}, A_{opt}

ALGORITHM 2. Clustering-Based Heuristic Edge Resource Allocation Algorithm (CHERA).

application $m \in M$. In the clustering procedure, all clusters seriatim absorb edge servers from the remaining candidates until all candidate edge servers are within the coverage of all clusters. CHERA selects the candidate edge server that can obtain minimal average data routing delay in cluster $Y_m^i \in Y_m$ to allocate resources for application $m \in M$.

Third, CHERA enumerates each candidate edge server in cluster Y_m^i to replace the current edge server that allocates resources for application $m \in M$. After each replacement process, CHERA reevaluates the average service response time of the overall edge network. Then, if the performance of the replacing result is better than the current one, the resource allocation situation will be updated. By heuristically replacing the edge server in each

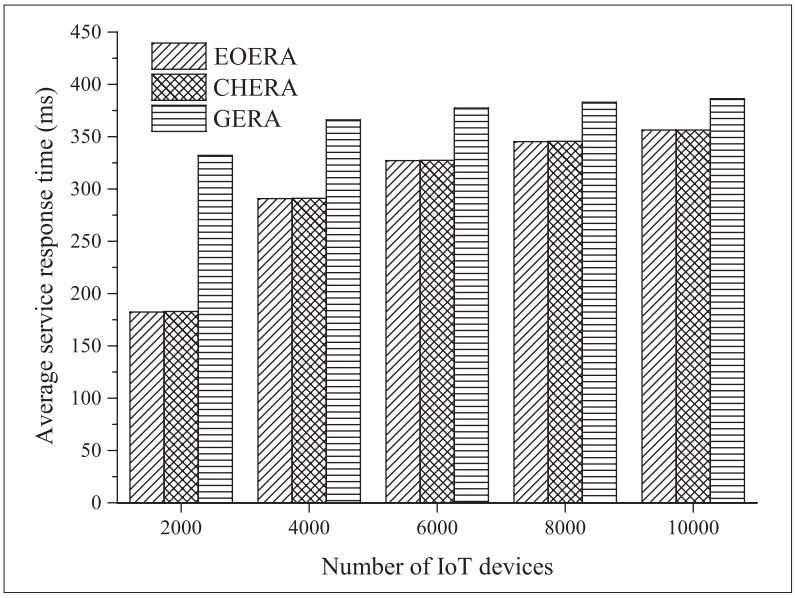

FIGURE 2. Performance comparisons of EOERA, CHERA, and GERA under different number of IoT-devices.

cluster for application $m \in M$ with the optimal candidate edge server, CHERA can get a near-optimal solution to the SRTM problem. The details of CHERA are described in Algorithm 2.

Analysis and Discussions

Obviously, to find an optimal edge resource allocation case for multiple applications of smart cities, EOERA may have to run for a very long time due to its exponential computational complexity. As described in Algorithm 1, the time complexity of EOERA is $O(|M| \cdot |V|^2 \cdot (C^k_{|V|})^{|M|})$. Thus, it can only be implemented on a small scale of edge networks with a few applications deployed in practice. Therefore, EOERA is just proposed as a baseline for our following more efficient scheme.

For CHERA in Algorithm 2, the running time of the initial heuristic procedure is $O(|M| \cdot |V| \cdot k \cdot (|V| + |M|))$. For each application, CHERA first initializes k clusters based on the result of the initial heuristic procedure, which will run for $O(k \cdot |V|^2)$ time. After that, CHERA will enumerate the candidate edge servers to replace the edge server that allocates resources for application $m \in M$ currently in each cluster. In order to take the resource allocation benefit in a global sense, for each alternative resource allocation case, CHERA reevaluates the average service response time of the overall edge network. Finally, CHERA determines whether to replace the edge servers hosting corresponding applications in each cluster according to the performance. The time cost of this evaluation for each application is $O(k \cdot |M| \cdot |V|^3)$. By adding them all up, we get the computational complexity of CHERA in Algorithm 2 with $O(k \cdot |M|^2 \cdot |V|^3)$.

Compared to EOERA, the computational complexity of CHERA in Algorithm 2 is much less. Although CHERA can only give a near optimal solution to the SRTM problem, it has much higher computational efficiency than EOERA. Therefore, EOERA is just presented as a baseline for CHERA. In practice, we are often given the edge resource allocation situations that cannot be changed and asked to find edge servers to further host a few new applications. Therefore, a general Greedy Edge Resource Allocation (GERA) algorithm is also presented to validate the performance of the proposed algorithms. It is well known that if one edge server receives more than half of the overall requests for application $m \in M$, the optimal resource allocation solution for application $m \in M$ will include this edge server. GERA iteratively evaluates each candidate edge server to allocate resources for application $m \in M$ by the average service response time under the condition that more than half of the application requests are routed to the edge server. Then, after greedily selecting edge servers to allocate resources for each application with minimal average service response time, GERA can get a near-optimal solution.

Numerical Results and Discussions

Experimental Settings

From the Topology Zoo [14], a collection of annotated network graphs derived from public network maps, we adopted a real-world online network topology, T-LEX, as the edge network topology of the IoT-based smart city in our experiments. A simulator in Python was developed to simulate the edge resource allocation for multiple applications in IoT-based smart cities. What's more, all algorithms ran on a virtual machine with double 2.10 GHz CPUs, 16.00 GB RAM, and the Ubuntu 64-bit operating system. In the following experiments, we assume a service request ratio from numerous IoT devices to application $m \in M$, following Poisson distribution at rate $\lambda = 0.8$, and the capacity of edge server $v \in V$ (i.e., $c_v = 1000$). The network delay of requests for application $m \in M$ on communication link $e_i \in E$ in edge networks was randomly selected between 5 ms and 50 ms, and the average network delay to the central cloud of service requests for application $m \in M$ (i.e., B_m) was randomly chosen between 100 ms and 400 ms [15].

Numerical Results

From Fig. 2, one can easily observe that with the same number of IoT devices, EOERA can get the best performance on minimizing the average service response time. What's more, the performance of CHERA and EOERA are much closer, and are both better than that of GERA under different numbers of IoT devices. Figure 2 also shows that the growth trend of average service response time slows down with increasing number of IoT devices in all schemes. This is mainly because the usage of edge resources has been saturated, and the extra service requests caused by the increasing IoT devices will be transmitted to the central cloud, which causes a constant network delay for each application in our experiments. It is clearly shown in Fig. 2 that when edge resources dominate the processing of service requests, CHERA can always get much better performance than that of GERA, and is much closer to the optimal solution obtained by EOERA.

Similar conclusions can be obtained from Fig. 3. As we can see, the performance of CHERA in solving the SRTM problem under different number of applications is much closer to that of EOERA and is much better than that of GERA. Figure 3 also shows that with the increasing number of different applications, the average service

response time obtained by EOERA and CHERA both grow slowly. Each edge server has limited service capacity, and CHERA iteratively finds the resource allocation scheme based on the current candidate edge servers for each application, leading to sub-optimal solutions for subsequent applications. Thus, the performance of CHERA is slightly poorer than that of EOERA with a large number of different applications. When edge servers only need to allocate resources for one application, all schemes can get the same performance. However, the result of GERA increases sharply with growing number of applications, which is much higher than that of CHERA.

Conclusions

To minimize the average service response time in IoT-based smart cities, we have presented two schemes, EOERA and CHERA, to optimally allocate edge resources for multiple applications. EOERA adopts an enumeration strategy to find an optimal resource allocation case for all multiple applications. Compared to EOERA, CHERA can not only give a near-optimal solution to the SRTM problem, but also achieve much higher computational efficiency. Numerical results on the performance comparisons of average service response time under different number of IoT devices and applications among EOERA, CHERA, and GERA are provided to show the promise of our proposed schemes. More complex and challenging issues will be considered in our future work. For example, we will further investigate the edge resource allocation problem applying advanced machine learning technology to adapt to the ever changing environment in smart cities.

Acknowledgment

This work was supported by the National Natural Science Foundation of China (61771374, 61771373, and 61601357), in part by China 111 Project (B16037), and in part by the Fundamental Research Fund for the Central Universities (JB171501, JB181506, JB181507, and JB181508).

References

[1] Y. Mehmood *et al.*, "Internet-of-Things-Based Smart Cities: Recent Advances and Challenges," *IEEE Commun. Mag.*, vol. 55, no. 9, Sept. 2017, pp. 16–24.
[2] I. Bisio *et al.*, "Context-Awareness Over Transient Cloud in D2D Networks: Energy Performance Analysis and Evaluation," *Trans. Emerging Telecommun. Technologies*, vol. 28, no. 2, 2017, pp. e3002.
[3] I. Bisio *et al.*, "GPS/HPS-and Wi-Fi Fingerprint-Based Location Recognition for Check-In Applications Over Smartphones in Cloud-Based LBSs," *IEEE Trans. Multimedia*, vol. 15, no. 4, 2013, pp. 858–69.
[4] H. Zhang *et al.*, "Connecting Intelligent Things in Smart Hospitals Using NB-IoT," *IEEE Internet of Things J.*, vol. 5, no. 3, 2018, pp. 1550–60.
[5] M. A. Rahman *et al.*, "Semantic Multimedia Fog Computing and IoT Environment: Sustainability Perspective," *IEEE Commun. Mag.*, vol. 56, no. 5, May 2018, pp. 80–87.
[6] W. Sun, J. Liu, and H. Zhang, "When Smart Wearables Meet Intelligent Vehicles: Challenges and Future Directions," *IEEE Wireless Commun.*, vol. 24, no. 3, June 2017, pp. 58–65.
[7] L. Zhou *et al.*, "When Computation Hugs Intelligence: Content-Aware Data Processing for Industrial IoT," *IEEE Internet of Things J.*, vol. 5, no. 3, 2018, pp. 1657–66.
[8] J. He *et al.*, "Multitier Fog Computing With Large-Scale IoT Data Analytics for Smart Cities," *IEEE Internet of Things J.*, vol. 5, no. 2, 2018, pp. 677–86.
[9] L. Zhou *et al.*, "When Collaboration Hugs Intelligence: Content Delivery over Ultra-Dense Networks," *IEEE Commun. Mag.*, vol. 55, no. 12, Dec. 2017, pp. 91–95.

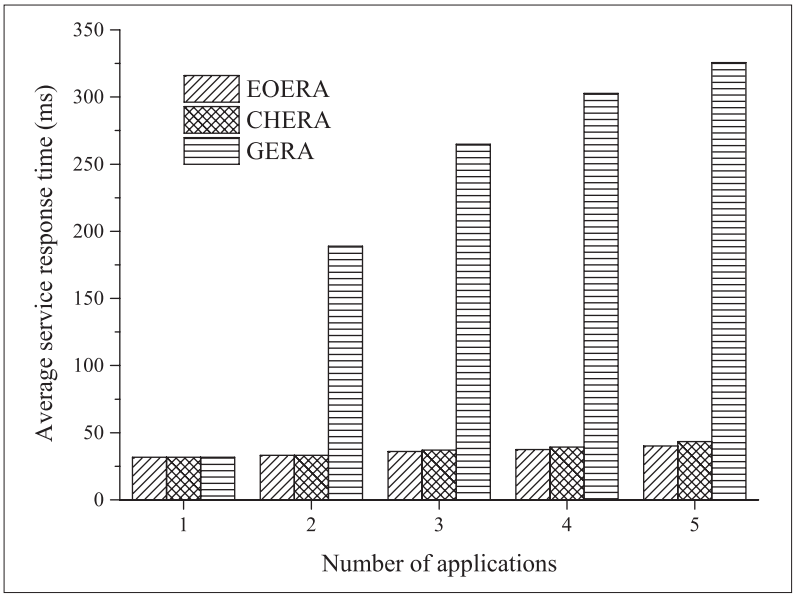

FIGURE 3. Performance comparisons of EOERA, CHERA, and GERA under different number of applications.

[10] S. Wu *et al.*, "Survey on Prediction Algorithms in Smart Homes," *IEEE Internet of Things J.*, vol. 4, no. 3, 2017, pp. 636–44.
[11] M. Wollschlaeger, T. Sauter, and J. Jasperneite, "The Future of Industrial Communication: Automation Networks in the Era of the Internet of Things and Industry 4.0," *Industrial Electronics Mag.*, vol. 11, no. 1, 2017, pp. 17–27.
[12] T. G. Rodrigues *et al.*, "Hybrid Method for Minimizing Service Delay in Edge Cloud Computing Through VM Migration and Transmission Power Control," *IEEE Trans. Computers*, vol. 66, no. 5, 2017, pp. 810–19.
[13] S. Saeednia, "New NP-Complete Partition Problems," *IEEE Trans. Info. Theory*, vol. 48, no. 7, 2002, pp. 2092–94.
[14] S. Knight *et al.*, "The Internet Topology Zoo," *IEEE JSAC*, vol. 29, no. 9, 2011, pp .1765–75.
[15] M. Satyanarayanan *et al.*, "The Case for VM-Based Cloudlets in Mobile Computing," *IEEE Pervasive Computing*, vol. 8, no. 4, 2009, pp. 14–23.

Biographies

Lei Zhao [S'17] received his B.S. degree in computer science and technology from Xidian University in 2015 and his M.S. degree in computer system architecture from Xidian University in 2018. Now he is pursuing a Ph.D. degree in the Department of Electrical and Computer Engineering, University of Victoria. His research interests cover smart city, mobile edge computing, software defined networking, and deep reinforcement learning.

Jiadai Wang [S'18] received her B.S. degree in information security from Qingdao University in 2017. She is currently a Master's student in the School of Cyber Engineering, Xidian University. Her research interests cover the Internet of Things and edge computing.

Jiajia Liu [S'11, M'12, SM'15] (liujiajia@xidian.edu.cn) is currently a full professor at the School of Cyber Engineering, Xidian University. His research interests cover wireless mobile communications, FiWi, IoT, and more. He has published more than 100 peer-reviewed papers in many prestigious IEEE journals and conferences, and currently serves as an Associate Editor for *IEEE Transactions on Communications* and *IEEE Transactions on Vehicular Technology*, an Editor for *IEEE Network*, and a Guest Editor of **IEEE TETC** and the *IEEE Internet of Things Journal*. He is a Distinguished Lecturer of IEEE ComSoc.

Nei Kato [A'03, M'04, SM'05, F'13] (kato@it.is.tohoku.ac.jp) is currently a full professor at GSIS, Tohoku University. He currently serves as the Vice-President (Member & Global Activities) of IEEE ComSoc, Editor-in-Chief of *IEEE Transactions on Vehicular Technology*, and Associate Editor-in-Chief of the *IEEE Internet of Things Journal*. He has also served as the Chair of the SSC and AHSN Technical Committees of IEEE ComSoc. He is a Distinguished Lecturer of IEEE ComSoc and the Vehicular Technology Society. He is a Fellow of IEICE.