Energy- and Reliability-Aware Computation Offloading with Security Constraints in MEC-Enabled Smart Cities

Bohai Zhao[1], Kai Peng[1(✉)], Fangyuan Zhu[2], and Shengjun Xue[3]

[1] College of Engineering, Huaqiao University, Quanzhou, China
kai.peng@hqu.edu.cn
[2] School of Electronic Science and Engineering, Xiamen University, Xiamen, China
[3] School of Computer Science and Technology, Silicon Lake College, Suzhou, China

Abstract. Smart city is a fast-growing system that provides a large number of collaborative services enabled by emerging technologies such as wireless sensor networks and radio frequency identification. Generally, massive heterogeneous smart devices in smart cities are served by limited resources, resulting in low processing efficiency and even insufficient reliability of applications. In this regard, people invented mobile edge computing (MEC) to provide smart devices with more processing capacity. Unfortunately, the computing resources of edge servers in MEC are limited. Consequently, it is of great importance to improve the resource management efficiency of the MEC system. In view of this, we construct a four-layer edge-enabled smart cities model in this paper. Technically, we propose an energy- and reliability-aware multi-objective optimization method which can jointly optimize the energy consumption of smart devices, the resource utilization of edge servers and improve the system's reliability while meeting the privacy constraints. Experimental results prove that our proposed method has good benefits to meet the needs of computation offloading in smart cities.

Keywords: Mobile edge computing · Multi-objective · Energy consumption · Reliability · Privacy protection

1 Introduction

In recent years, with the continuous improvement of city intelligence and the continuous upgrading of mobile device performance, the types of applications show a diversification trend, at the same time, the requirements of mobile users are also becoming more and more widespread [1–3]. The urgency of making the city a more suitable place for quality communication is triggering many initiatives from academia and industry. Benefiting the development of various core

M. R. Khosravi et al. (Eds.): CloudComp 2021, LNICST 430, pp. 207–220, 2022.
https://doi.org/10.1007/978-3-030-99191-3_16

technologies in the Internet of Things (IoT), a new concept came into existence, called smart city [4].

Smart cities contain countless smart devices (SDs), and the number of these SDs continues to grow at a fairly good clip. According to a study by the Cisco Visual Networking Index, the national network traffic network will grow nearly sevenfold from 2017 to 2022, to around 77 exabytes per month, which will undoubtedly be a major challenge for smart cities [5]. Nevertheless, the resources of SDs are often limited, resulting in limited computing speed, massive energy consumption and even inevitable faults, which may affect the further development of smart cities [6,7].

Fortunately, the emergence of mobile edge computing (MEC) brings hope to deal with the above problem [8–11]. The most straightforward idea is offloading the workflow applications to the edge servers for execution [12,13]. At the same time, the edge servers are generally placed close to the edges of the SDs, thus the energy consumption generated during computation migration can also be optimized to some extent.

However, the computing resources of edge servers are also limited, which means the workflow applications can not be offloaded arbitrarily, and proper offloading strategies must be devised for them. Otherwise, unreliable scenarios such as server downtime may occur. Furthermore, the edge servers in smart cities are heterogeneous, and even for the same edge server, the state of resources is set to dynamic for energy-efficient. Furthermore, compared with general applications, workflow applications in smart cities are not so easy to handle because of their complex structure. Besides, the security of tasks served by external servers is also one of the critical concerns in smart cities [14–16].

In view of the above analysis, we investigate the offloading strategy formulation problem for SDs in smart cities. The main contributions of this paper can be summarized as follows.

(1) In order to improve the reliability of the system, we build a four-layer edge-enabled smart city model based on the traditional three-layer MEC structure. Specifically, we combine the lazy shadow scheme with the MEC architecture and add a shadow server layer between the edge servers and core network. Besides, the workflow applications generated by SDs and the edge servers are defined as heterogeneous.
(2) We propose an energy- and reliability-aware multi-objective optimization method with security constraint based on the Non-dominated Sorting Genetic Algorithm II (ERMOS). Both the energy consumption of SDs, the resource utilization of edge servers, as well as the reliability of the edge servers are optimized jointly with security constraints.
(3) We carry on sufficient experiments and analysis to show the advantages of ERMOS.

The rest sections of this paper are described as follows. Section 2 introduces the related work in two directions, i.e., computing offloading for workflow application and computing offloading in smart cities. Then, the system model, workflow applications model, as well as mathematical model of the optimization goals

are described in Sect. 3. Section 4 introduces the details of the proposed method. Then, the comparative experiments and experimental results are described in Sect. 5. Finally, we conclude our work and describe our future work.

2 Related Work

Computation Offloading for Workflow Applications. Xu et al. [17] studied the computation offloading issue in wireless metropolitan area network. They set the computing power of edge servers as a fixed value. Then, they proposed a novel method to optimize the energy consumption and the time consumption of the system. Aiming to reduce the algorithm overhead during offloading strategy formulation, Fan et al. [18] investigated the computation offloading problem by using a DAG-based model, and proposed a shortest-path-based algorithm offloading strategy for mobile-edge workflow applications. Similarly, Zhu et al. [19] devised a network resource allocation algorithm on the basis of deep Q-learning, which effectively reduced the completion time and energy consumption of service workflow applications.

Computation Offloading for Smart City. In order to minimize the energy consumption and execution latency of SDs while improving the quality of service for users in smart cities, Mazza et al. proposed a cluster-based computing offloading method in [20]. Qian et al. [21] focused on the computation offloading in IoT-Based smart cities. They proposed a scalable and sustainable IoT framework, which can reduce the end-to-end offloading time and energy consumption. Esposito et al. [22] propose a new method to cope with data privacy security issues during data packet transmission in cloud-enabled smart cities.

At present, most of the research on edge-enabled systems mainly focuses on multi-objective optimization for SDs. However, the unreliability of the edge server, such as downtime, may lead to workflow execution failure. Especially for smart cities, when facing massive data to be processed, the edge servers may get into a state of collapse at any time, resulting in a remarkable decline in the service benefit. In view of this, inspired by the lazy shadow scheme [23], we built a four-layer edge-enabled smart city model to explore the computation offloading problem for workflow applications generated by SDs. The energy consumption of SDs, the resource utilization of edge servers, the reliability of the system as well as the security of workflow applications during task processing are all taken into consideration.

3 System Model

In this section, the edge-enabled smart city mode is constructed firstly. Next, the computation model of the total energy consumption of SDs, the resource utilization, the reliability of the edge servers as well as the constraint model of privacy protection are described respectively. Finally, the problem formulation and our optimization goals are proposed. Some key variables are described in Table 1.

Table 1. Variables and explanations

Meaning	Notation
Workflow applications generated by SDs	SD
The maximum number of workflow applications	W
The data need to be calculated	S
The relationship between tasks	D
The power of SDs	P
Offloading strategy of tasks	OF
Propagation latency	PE
The maximum number of edge servers	ES
The maximum number of shadow servers	SS
Error probability of servers	RP

3.1 Network System Model

As Fig. 1 shows, the four-layer edge-enabled smart city model is constructed. This model is composed of four layers (i.e., SDs, edge servers, shadow servers and core network). Lower-layer terminals can request upper-layer services through corresponding wireless access nodes. Specifically, SDs in layer 1 will generate a certain number of workflow applications firstly. These workflow applications can directly obtain the services of SDs, but when faced with a huge amount of data, computation offloading, namely, to request upper-layer services, which is undoubtedly a better choice.

As the infrastructure of the second layer, edge servers can provide SDs with high-quality services through local area network (LAN). Specifically, edge servers can also communicate with the core network through wide area network (WAN) directly to ensure the workflow applications can be served by enough computing resources. Namely, the same as the traditional three-layer MEC model.

Particularly, based on the lazy shadow scheme [23], we construct a new layer between edge servers and the core network, defined as the shadow servers layer. Shadow servers are used to undertake some specific tasks from edge servers layer, so as to improve the reliability of workflow applications processing. The principle of shadow servers will be elaborated in Sect. 3.5.

Finally, the core network with almost infinite resources will serve as the uppermost terminal to ensure that the entire architecture has sufficient resources. All lower-layer servers can directly exchange data with it through the WAN.

3.2 Workflow Applications Model

The workflow applications are donated as $SD_w = (S_w, D_w)$ where $S_w = \{s_{1,w}, s_{2,w}, \ldots, s_{m,w}\}$ represents the data need to be calculated in the w-th workflow application while the number of the tasks are denoted as $m \in \{1, 2, \ldots, M\}$,

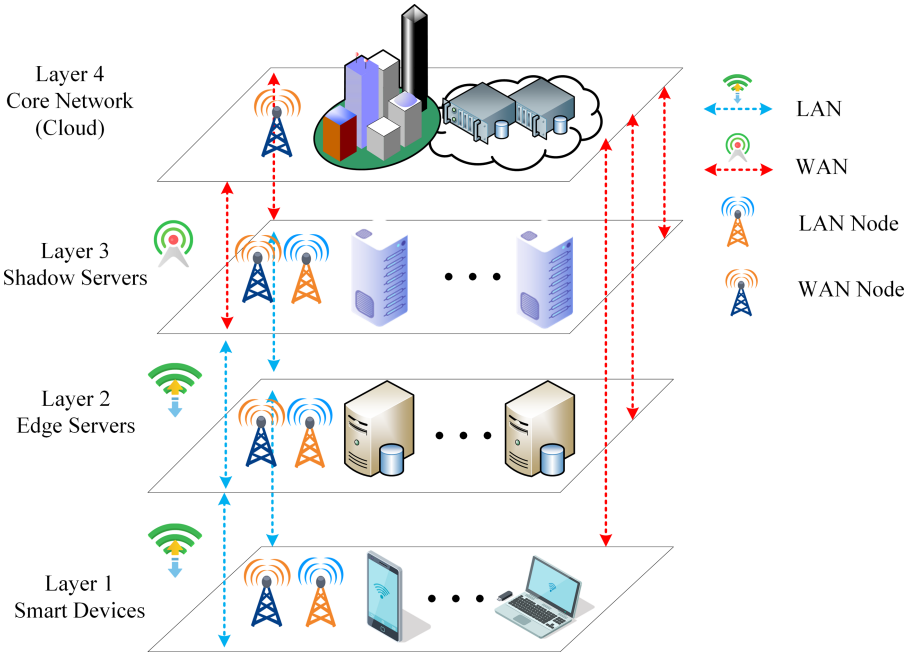

Fig. 1. Four-layer MEC-enabled smart city mode

and $w \in \{1, 2, \ldots, W\}$. Then, D_w represents the relationship between tasks and can be further denoted by a binary array, namely, $D_w = \{R(s_{m,w}, s_{m',w}), E_{m,m'}\}$ where $R(s_{m,w}, s_{m',w})$ represents the relationship between task $s_{m,w}$ and task $s_{m',w}$. In addition, $E_{m,m'}$ represents the transmission data between $s_{m,w}$ and $s_{m',w}$.

3.3 Energy Consumption Model

The total energy consumption mainly consists of three parts, namely, transmission energy, queue waiting energy and computation energy.

Layer 1. First of all, since SDs has certain computing resources, so the workflow applications can still obtain services directly, the energy consumption at this could be calculated and it is given by Eq. (1)

$$E_1(s_{m,w}) = \frac{S_w}{G_{local}} \times P_{work}, \tag{1}$$

where G_{local} indicates the computing resources of the SDs, and P_{work} represents the power cost when SDs work properly.

Layer 2. Then, when task requests for upper service, transmission energy is generated first.

$$E_2^{\alpha}(s_{m,w}, s_{m',w}) = \frac{E_{m,m'}}{W_{m,m'}} \times P_{cors}. \tag{2}$$

As Eq. (2) shows, there is a sequence constraint between two tasks $s_{m,w}$ and $s_{m',w}$, it is necessary to transfer some data to $s_{m',w}$ after the precursor task $s_{m,w}$ has been completed.

Besides, $W_{n,n'}$ represents the bandwidth for transmission. Specially, for layer 2, when two tasks are offloaded to the same edge server (i.e., $OF_{m,w} = OF_{m',w}$), we define $QC = 1$, and the bandwidth $W_{n,n'} = \infty$. However, when $OF_{m,w} \neq OF_{m',w}$, but both $OF_{m',w}$ and $OF_{m',w} \in \{1, 2, \ldots, ES\}$, $QC = 2$. $W_{n,n'}$ is defined as W_{sa}. When one of $OF_{m,w}$ and $OF_{m',w} \in \{1, 2, \ldots, ES\}$ and the other equals 0, at this time, $QC = 3$ and $W_{n,n'} = W_{df}$.

$$W_{n,n'} = \begin{cases} \infty & QC = 1 \\ W_{sa} & QC = 2 \\ W_{df} & QC = 3 \end{cases}. \tag{3}$$

Similarly, when QC gets different values, SDs have two optional states, namely, working state and idle state. The power P_{cor} of SDs in different states follows the principle shown in Eq. (4)

$$P_{cor} = \begin{cases} P_{idle} & QC = 1 \\ P_{idle} & QC = 2 \\ P_{work} & QC = 3 \end{cases}. \tag{4}$$

Similar to our previous work [24], the queue waiting energy consumption E_2^{β} is obtained based on queuing theory [25].

As shown in Eq. (5), the average queue length, the time interval of task arrival and the service time are denoted as L_{ave}, V_{task}, T_{sev}, respectively.

$$E_2^{\beta} = (\frac{L_{ave}}{V_{task}} - \frac{1}{T_{sev}}) \times P_{idle}. \tag{5}$$

Finally, the energy consumption can be obtained by Eq. (6) in which G_{es} represents the processing power of the es-th edge server ($es \in \{1, 2, \ldots, ES\}$). Then, PE_{es} represents the propagation delay during the offloading of $d_{n,a}$ to edge server es.

$$E_2^{\gamma}(s_{m,w}) = (\frac{S_w}{G_{es}} + PE_{es}) \times P_{idle}. \tag{6}$$

Layer 3. In this layer, the energy consumption is mainly generated in the stage of processing the copy of tasks and can be calculated by Eq. (7)

$$E_3 = \begin{cases} (\frac{SF_w}{V^{low}} \times P_{idle}) & FM = 0 \\ (\frac{S_w - SF_w}{V^{fast}} + \frac{SF_w}{V^{low}}) \times P_{idle}) & FM = 1 \end{cases}, \tag{7}$$

where SF_w represents the amount of data that has been computed before the shadow server status changes, and FM represents the system status. When

the value of FM is equal to 0, it means that no fault occurs. After SD_w^τ has been processed by es, its corresponding copy will also be discarded. But when $FM = 1$, which means a failure has occurred. Then, the shadow server ss will replace es to complete the subsequent processing for SD_w^τ. The specific model and related parameters of the lazy shadow scheme will be described in Sect. 3.5.

Layer 4. Different from the edge servers layer, the core network has nearly infinite computing resources, thus we will not consider the queuing waiting energy and the transmission energy caused by the inherent relationship of workflow applications. However, due to the long physical distance between core network and SDs, it is necessary to consider the energy consumption TE_c caused by network latency during offloading. At the same time, the propagation latency PE_c of the workflow application on the core network is also not negligible. In summary, the total energy consumption in layer 4 can be expressed as follows.

$$E_4(s_{m,w}) = TE_c + (\frac{S_w}{G_c} + PE_c) \times P_{idle}. \tag{8}$$

3.4 Resource Utilization Model

Edge servers are often defined as heterogeneous in smart cities, and their resources are extremely precious. The full use of resources also affects the level of the quality of service.

The resource utilization of es can be expressed by Eqs. (9) and (10). Generally, the number of virtual machines that can be accommodated in an edge server determines its resource capacity, namely, VM_{es}.

$$R_{es} = \begin{cases} \frac{K_{es}}{VM_{es}} & K_{es} < VM_{es} \\ 1 & K_{es} \geq VM_{es} \end{cases}, \tag{9}$$

$$K_{es} \in \{0, 1, \ldots, A \times Q\}, \tag{10}$$

where K_{es} represents the number of tasks offloaded to es, and $A \times Q$ represents the capacity of es.

3.5 Reliability Model

When an edge server es provides services for SD_w, it cannot guarantee to be completely accurate. On the contrary, a fatal error may occur in that edge server at any time. Therefore, in this paper, we allocated several shadow servers on the upper layer above edge servers, denoted as $ss \in \{1, 2, \ldots, SS\}$. Then, some tasks with reliability constraints are defined as SD_w^τ. When SD_w^τ is determined need to be offloaded to es, they will be copied to the corresponding ss. Next, when an error occurs while SD_w^τ is being calculated, ss will replace es in charge of processing workflow application SD_w^τ. When there is no error occurs, ss will discard the copy of SD_w^τ. Technically, ss has two states, namely, low-speed V_{low}

and high-speed V_{fast}, to perform the copy tasks. The state switching depends on whether an error occurs in es [23].

As a hardware infrastructure, the error probability of edge servers is very low, i.e. the typical rate is 4%. As a similar facility, shadow server also can not guarantee 100% reliability. Therefore, in this article, we use RP_{es} and RP_{ss} to represent the error probability of edge servers and shadow servers, respectively. Then the possibility that SD_w^τ can be successfully processed in an architecture equipped with n shadow servers can be expressed as

$$PO(s_{m,w}) = 1 - RP_{es} \times RP_{ss}^n. \tag{11}$$

3.6 Privacy Preservation Model

Through computation offloading, users in smart cities can enjoy high-quality services but may also have serious privacy leakage issues. Therefore, whether the privacy of users in smart cities can be guaranteed has aroused strong concern. Fortunately, faced with massive workflow applications, hackers must obtain enough task data that they can pose a threat to user privacy. Thus, privacy constraint has been added in this paper to improve the security of the system.

Specifically, two tasks with privacy conflicts should not be serviced by a single edge server [11]. The privacy constraints between two tasks $s_{m,w}$ and $s_{m',w}$ are redefined as $CO_w(m, m')$

$$CO_w(m, m') = \begin{cases} 1 & Have\ Privacy\ Conflicts \\ 0 & No\ Privacy\ Conflicts \end{cases}. \tag{12}$$

As Eq. (12) shows, when task $s_{m,w}$ and $s_{m',w}$ have privacy conflicts, namely, $CO_w(m, m') = 1$. The corresponding offloading strategy $OF_{m,w}$ can not be equal to $OF_{m',w}$. But when $CO_w(m, m') = 0$, there is no limit. Namely, $OF_{m,w}$ and $OF_{m',w}$ must satisfy the constraint shown by

$$\begin{cases} OF_{m,w} \neq OF_{m',w} & CO_w(m, m') = 1 \\ Unlimited & CO_w(m, m') = 0 \end{cases}. \tag{13}$$

3.7 Problem Formulation

In this paper, the energy consumption of SDs, the resource utilization of edge servers, the reliability of the system and the security of workflow applications are considered. Our optimization problem can be formulated as

$$Min\ \{E_{sum}(d_{m,w})\}, \tag{14}$$

$$Max\ \{R_{es}, PO(s_{m,w})\}, \tag{15}$$

$$OF_{m,w} \neq OF_{m',w}\ \ when\ \ CO_w(m, m') = 1. \tag{16}$$

4 Energy- and Reliability-Aware Multi-objective Optimization Method with Security Constraint (ERMOS)

In this part, faced with the four-layer network model constructed above, we propose a multi-objective optimization method for workflow applications, named Energy- and Reliability-aware Multi-Objective Optimization Method with Security Constraint (ERMOS). Based on NSGA-II [26], ERMOS can be roughly divided into 6 basic steps, namely, initialization, selection, crossover, mutation, calculation of crowding distance, and population update. The details of the method are described as follows.

The pseudo code is shown in Algorithm 1. First, based on genetic algorithms, we need to construct a new parent population PC_p. The size of the parent population TN_p is equal to the total number of tasks performed in SD_w. And the processing of tasks in the population also needs to meet the relational constraints D_w (Lines 1–3).

Then, we need to calculate the energy consumption, resource utilization, and reliability of the current iteration based on Eqs. (1)–(11). Then, we can get a new offspring population PC_o through selection, crossover and mutation based on PC_p. The size of PC_o is the same as PC_p. Specifically, the selection operation will select and sort based on the crowded distance of individuals in PC_p, and the crowding distance $Dis_{m,w}$ can be expressed as Eq. (17). $Dis_{m,w}$ actually depends on our proposed optimization goals, namely, the normalized value of energy consumption, resource utilization, and reliability probability. The smaller $Dis_{m,w}$, the higher the optimization efficiency of the individual under the current offloading strategy, and the greater the possibility that the individual will enter the next generation. In addition, the crossover and mutation operations effectively reduce the probability of local convergence (Lines 5–9).

After generating the offspring population, we need to fuse PC_p and PC_o to generate a mixed population, denoted as PC_{Mix}. Then, the parent population PC_p will be temporarily emptied. After non-dominated sorting from mixed populations, we can get several non-dominated layers NF (Lines 10–12). Finally, the best N individuals will be selected layer by layer to join the new parent population to achieve the purpose of updating the parent population. Particularly, if the last non-dominant layer cannot just fill the parent population. Then again, the better individuals in this layer will be selected to fill the parent population based on the crowding distance (Lines 13–20).

$$Dis_{m,w} = Normalized(E_{sum}, R_{es}, PO(s_{m,w})). \tag{17}$$

5 Experimental Evaluation

In this section, we conduct a detailed experimental evaluation of the method. Specifically, we construct three comparison methods in this paper. Then, the

Algorithm 1. ERMOS

Input: Workflow Applications SD_w, Maximum iteration times IT_{Max},
Output: Optimal offloading strategy $O_{m,w}$

$IT_{cur} = 1$
Initialize parent population PC_p
$TN_p = M \times W$
while $IT_{cur} \leq IT_{Max}$ **do**
  for PC_p **do**
    Calculate crowding distance $Dis_{m,w}$ based on (17)
  end for
  PC_o = selection, crossover and mutation PC_p
  $TN_o = TN_p$
  $TN_{Mix} = TN_p + TN_o$
  $NF = fastnondominatesort(TN_{Mix})$
  $PC_p = \emptyset$
  $s = 0$
  while $len(PC_p) + len(NF[s]) < TN_p$ **do**
    $PC_p = PC_p \cup NF[s]$
    $s = s + 1$
  end while
  Calculate crowding distance of $(NF[s])$ based on (17)
  $PC_p = PC_p \cup$ select $NF[s]$
  $IT_{cur} = IT_{cur} + 1$
end while
return $O_{m,w}$

experimental results are analyzed and the effectiveness of the proposed method is verified accordingly.

5.1 Experimental Setting

First, we construct three comparison methods. Their specific descriptions are shown as follows.

Random Offloading Method (RO): All workflow applications will be offloaded randomly.
Sequential Offloading Method (SO): All workflow applications will be offloaded sequentially.
All Random Offloading to Edge Servers Method (ROE): All workflow applications will be offloaded randomly, but they only consider edge servers and will not be offloaded to other locations.
Our Method (ERMOS): The proposed method. All workflow applications are offloaded based on ERMOS.

We implement these methods by MATLAB on a physical machine with 2 Intel Core i7-7700U 2.80 GHz processors and 16 GB RAM and the operating system is Win10 64.

5.2 Experimental Result and Discussion

In this paper, the number of workflow applications and the maximum available edge servers are selected as experimental variables. Besides, considering that not all tasks require reliability guarantees. Therefore, only half of the edge servers have the ability to communicate with the shadow servers, which can obtain further reliability guarantee services. We have conducted 50 convergence experiments for each target.

The comparison among the four methods in terms of energy consumption are shown in Fig. 2 and Fig. 3. Due to a large number of workflow applications being offloaded to the local or core network, RO and SO will consume more energy than ROE and ERMOS. For ROE, while workflow applications are all offloaded to seemingly superior edge servers, there is a high probability that this random offloading strategy will still face challenges such as tasks exceed the maximum capacity of edge servers resulting in an energy-intensive wait in the queue. In contrast, the proposed method ERMOS can maintain good performance in energy consumption optimization, whether the number of workflow applications or edge servers changes.

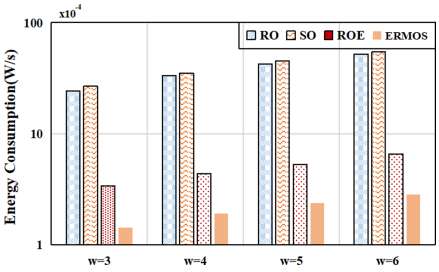

Fig. 2. Comparison of energy consumption when ES = 3

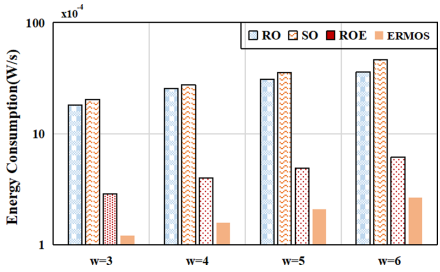

Fig. 3. Comparison of energy consumption when ES = 4

In terms of resource utilization, as shown in Fig. 4 and Fig. 5, whether the number of workflow applications or the number of edge servers changes, ERMOS can still maintain a high resource utilization. At the same time, combined with the above analysis of energy consumption, ERMOS can make a suitable computing offloading strategy to optimize energy consumption and reduce the waste of resources jointly.

Finally, as Fig. 6 shows, by adding several shadow servers to MEC structure, the reliability of the workflow application processing can be effectively improved. At the same time, the comparison between ERMOS and ROE can also prove that ERMOS has a practical optimization results in terms of reliability.

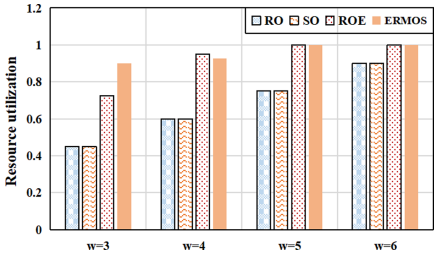

Fig. 4. Comparison of resource utilization when ES = 3

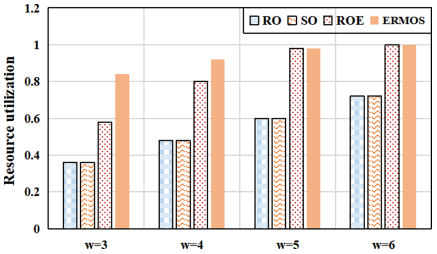

Fig. 5. Comparison of resource utilization when ES = 4

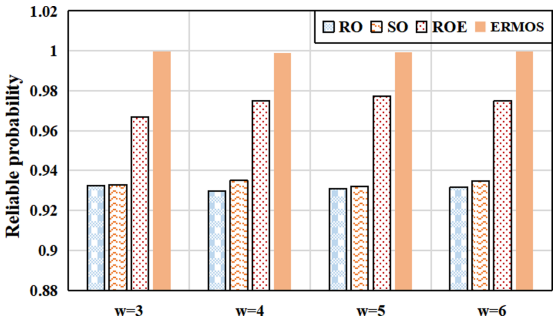

Fig. 6. Comparison of reliability when ES = 5

6 Conclusion

In this study, we investigate computation offloading for SDs in smart city. In particular, considering the reliability requirements of smart cities, we construct a four-layer MEC-enabled smart city model. The model of energy consumption, resource utilization and the reliability guarantee scheme are all described in details. Correspondingly, we propose a multi-objective optimization method, called ERMOS. The energy consumption of SDs, the resource utilization of edge servers, the system's reliability, and the security of workflow applications are taken into consideration jointly. Finally, extensive experiments and detailed discussion have proved the effectiveness and efficiency of our proposed solution.

In the future, we intend to combine the proposed method with blockchain, machine learning and other promising technologies in practical scenarios, such as intelligent warehousing.

Acknowledgment. The authors thank for the National Science Foundation of China (Grant No. 61902133), the Fundamental Research Funds for the Central Universities (ZQN-817), Quanzhou Science and Technology Project (No. 2020C050R) and College Students Innovation and Entrepreneurship Project (202010385011), Huaqiao University 2021 Teacher Teaching Development Reform Project (No. HQJG202120).

References

1. Giang, N.K., et al.: CityFlow: exploiting edge computing for large scale smart city applications, p. 8679234 (2019)
2. Khan, L.U., Yaqoob, I., Tran, N.H., Kazmi, S.M.A., Dang, T.N., Hong, C.S.: Edge computing enabled smart cities: a comprehensive survey. Networking and Internet Architecture (2019). arXiv:1909.08747
3. Mora, O.B., Rivera, R., Larios, V.M., Beltrán-Ramírez, J.R., Maciel, R., Ochoa, A.: A use case in cybersecurity based in blockchain to deal with the security and privacy of citizens and smart cities cyberinfrastructures. In: 2018 IEEE International Smart Cities Conference (ISC2), pp. 1–4. IEEE (2018)
4. Camero, A., Alba, E.: Smart city and information technology: a review. Cities **93**, 84–94 (2019)
5. Cisco Visual Networking Index. Cisco visual networking index: global mobile data traffic forecast update, 2017–2022. Technical report, Cisco, San Jose, CA, USA (2019)
6. Deng, S., Huang, L., Taheri, J., Zomaya, A.Y.: Computation offloading for service workflow in mobile cloud computing. IEEE Trans. Parallel Distrib. Syst. **26**(12), 3317–3329 (2014)
7. Abbas, N., Zhang, Y., Taherkordi, A., Skeie, T.: Mobile edge computing: a survey. IEEE Internet Things J. **5**(1), 450–465 (2017)
8. Peng, Kai, et al.: An energy- and cost-aware computation offloading method for workflow applications in mobile edge computing. EURASIP J. Wirel. Commun. Netw. **2019**(1) (2019). Article number: 207. https://doi.org/10.1186/s13638-019-1526-x
9. Wu, H., Sun, Y., Wolter, K.: Energy-efficient decision making for mobile cloud offloading. IEEE Trans. Cloud Comput. **8**(2), 570–584 (2018)
10. Hu, Y.C., Patel, M., Sabella, D., Sprecher, N., Young, V.: Mobile edge computing-a key technology towards 5G. ETSI White Paper **11**(11), 1–16 (2015)
11. Xu, X., Liu, X., Xu, Z., Wang, C., Wan, S., Yang, X.: Joint optimization of resource utilization and load balance with privacy preservation for edge services in 5G networks. Mob. Netw. Appl. **25**(2), 713–724 (2019). https://doi.org/10.1007/s11036-019-01448-8
12. Meng, T., Wolter, K., Huaming, W., Wang, Q.: A secure and cost-efficient offloading policy for mobile cloud computing against timing attacks. Pervasive Mob. Comput. **45**, 4–18 (2018)
13. Cheng, J., Shi, Y., Bai, B., Chen, W.: Computation offloading in cloud-RAN based mobile cloud computing system. In: 2016 IEEE International Conference on Communications (ICC), pp. 1–6. IEEE (2016)
14. Peng, K., Zhang, Y., Wang, X., Xu, X., Li, X., Leung, V.C.M.: Computation offloading in mobile edge computing. In: Shen, X., Lin, X., Zhang, K. (eds.) Encyclopedia of Wireless Networks, pp. 1–5. Springer, Cham (2019). https://doi.org/10.1007/978-3-319-78262-1_331
15. Shi, W., Cao, J., Zhang, Q., Li, Y., Lanyu, X.: Edge computing: vision and challenges. IEEE Internet Things J. **3**(5), 637–646 (2016)
16. Zhang, W.-L., Guo, B., Shen, Y., Wang, Y., Xiong, W., Duan, L.T.: Computation offloading on intelligent mobile terminal. Chin. J. Comput. **39**(5), 1021–1038 (2016)
17. Xu, X., et al.: An energy-aware computation offloading method for smart edge computing in wireless metropolitan area networks. J. Netw. Comput. Appl. **133**, 75–85 (2019)

18. Fan, L., Liu, X., Li, X., Yuan, D., Xu, J.: Graph4Edge: a graph-based computation offloading strategy for mobile-edge workflow applications. In: 2020 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), pp. 1–4. IEEE (2020)
19. Zhu, A., et al.: Computation offloading for workflow in mobile edge computing based on deep Q-learning. In: 2019 28th Wireless and Optical Communications Conference (WOCC), pp. 1–5. IEEE (2019)
20. Mazza, D., Tarchi, D., Corazza, G.E.: A cluster based computation offloading technique for mobile cloud computing in smart cities. In: 2016 IEEE International Conference on Communications (ICC), pp. 1–6. IEEE (2016)
21. Qian, L.P., Wu, Y., Ji, B., Huang, L., Tsang, D.H.K.: HybridIoT: integration of hierarchical multiple access and computation offloading for IoT-based smart cities. IEEE Netw. **33**(2), 6–13 (2019)
22. Esposito, C., Castiglione, A., Frattini, F., Cinque, M., Yang, Y., Choo, K.-K.R.: On data sovereignty in cloud-based computation offloading for smart cities applications. IEEE Internet Things J. **6**(3), 4521–4535 (2018)
23. Cui, X., Znati, T., Melhem, R.: Adaptive and power-aware resilience for extreme-scale computing. In: 2016 International IEEE Conferences on Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld), pp. 671–679. IEEE (2016)
24. Peng, K., Zhao, B., Xue, S., Huang, Q.: Energy- and resource-aware computation offloading for complex tasks in edge environment. Complexity **2020**, 1–4 (2020). https://doi.org/10.1155/2020/9548262
25. Xu, X., et al.: Multiobjective computation offloading for workflow management in cloudlet-based mobile cloud using NSGA-II. Comput. Intell. **35**, 12 (2018)
26. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. **6**(2), 182–197 (2002)