An Edge Server Placement Method with Cyber-Physical-Social Systems in 5G

Xing Zhang[1], Jielin Jiang[1], Lianyong Qi[2], and Xiaolong Xu[1(✉)]

[1] School of Computer and Software,
Nanjing University of Information Science and Technology, Nanjing, China
[2] School of Information Science and Engineering, Qufu Normal University,
Qufu, China

Abstract. Recently, cyber-physical-social systems (CPSS), an advanced information system, have been introduced to promote the development of the smart society. It makes the control of machines in all walks of life more intelligent in an efficient, convenient, and stable way. Besides, with the maturity of edge computing, the task requests emitted by users in CPSS are tent to be transmitted edge servers for immediate processing. Nevertheless, some problems, i.e., high latency and low utilization, exist in current network placement. It leads to the problem that users in the CPSS are unable to enjoy instant and efficient processing. Given these problems, this paper designs an edge server placement method (ESPM) to alleviate this situation. To be specific, a system model designed according to this scenario is presented firstly. Then, the multi-objective evolutionary algorithm, i.e., improving the strength pareto evolutionary algorithm (SPEA2), is applied in this paper to optimize the access delay and load balance variance with the propose of enhancing the service experience of users. Furthermore, the normalization methods, i.e., the technique for order preference by similarity to an ideal solution (TOPSIS) and multi-criteria decision-making (MCDM) are selected to produce the standard data and optimal strategy. Finally, the experimental results show the effectiveness of ESPM.

Keywords: CPSS · Server placement · Evolutionary algorithm · Edge computing

1 Introduction

Business innovation and industrial intelligence pave the way for intelligent society, smart machines and networking in the future. Also, the persistent improvement of big data, cloud computing, Internet of things (IoT) and so on in recent years has promoted the further integration of traditional physical systems and advanced information technologies as well as the maturity of cyber-physical-social systems (CPSS) [1]. CPSS cover embedded environment perception, dynamic analysis of personnel organization behavior, network communication and network control, which makes the physical system have the functions of

H. Song and D. Jiang (Eds.): SIMUtools 2020, LNICST 370, pp. 127–139, 2021.
https://doi.org/10.1007/978-3-030-72795-6_11

calculation, communication, precise control and remote cooperation [2]. CPSS will be applied in many fields, such as intelligent enterprise, intelligent transportation, intelligent home, intelligent medical treatment and so on [3]. CPSS enable the physical system to have efficient computing, instant communication and other functions. This makes machine system more evolved and human organizations operate machines more stably and immediately through cyberspace. Besides, CPSS also lead to the rapid development of IoT technology and the more widespread utilization in a variety of intelligent scenarios [4].

However, it is precise because handheld devices, IoT devices and various intelligent scenarios are becoming more and more intelligent, more users have higher standards and stricter requirements for these application scenarios [5]. Besides, when the physical system combines network information and social information, the mass data transmission also puts forward higher requirements for network communication (i.e., bandwidth and speed) [6]. This also reflects people's inability to enjoy high-quality and low latency network services. Therefore, how to deal with these massive data in CPSS timely and effectively is an urgent problem to be solved [7]. There is no doubt that it is meaningless to consider the quality of service outside the network performance which signifies the basis of the quality of service [8]. The introduction of 5G network to accelerate the evolution of intelligent application scenarios can not only improve the data transmission rate and reduce the delay but also make the intelligent scenarios more advanced [9].

To provide timely and efficient feedback for users in CPSS, it is undoubtedly necessary to make full use of edge computing, so that users are able to experience high-quality service applications in real-time [10]. Technically, edge computing has rich computing resources. It could effectively reduce the task offloading delay by placing the execution task on the computing node close to the terminal device. It enables users to be geographically close to the servers that are processing resources, thus reducing the delay of offloading tasks and obtaining a higher quality of service [11]. Specifically, in CPSS, base stations are processed to evolve into edge computing nodes for serving users covered by them. In addition to the advantages of edge computing in offloading tasks, the proximity of distance also reduces the possibility of the harm to users caused by traditional information interception, thus increasing the security of users' privacy.

In general, placing edge server is an infrastructure requirement for edge computing. Unfortunately, most of today's researches on edge computing are only for theory, and there are few studies on practical applications such as edge server placement. In this paper, we focus on the edge server placement in a mobile edge computing environment that provides wireless internet coverage for mobile users.

In the edge server (ES) placement strategy, each ES node has limited computing resources. Most of them cannot handle the massive tasks at the same time [12]. This will lead to produce some services delay in the calculation process, affect service execution and reduce service efficiency. It is unacceptable for users in the 5G scenarios. Therefore, we need to significantly reduce the overall delay. Besides, we need to consider how to improve the resource utilization as much as

possible owing to the limited computing resources of each node. In addition, we need to trade-off all nodes to ensure the stability of each node for performance maximization.

Given these facts, achieving a reasonable ES placement strategy by reducing transmission latency and ensuring load balance to improve overall performance is a huge challenge. In this paper, an edge server placement method, namely ESPM, is designed for the placement of edge devices in 5G network.

Specifically, the pivotal motivations and contributions of this paper are shown below:

- Few studies research on the edge server placement method while pursuing the minimum access delay and load balance variance in CPSS scenario. So a unique task offload method in CPSS based on edge computing is designed in this paper.
- The evolutionary algorithm and normalization method are collectively deployed in this paper to obtain the feasible offloading strategies and select the optimal strategy.

2 Related Work

The characteristics of edge computing, such as large storage capacity and strong computing power, are particularly prominent in CPSS. In the previous literature, CPSS and its outstanding advantages have been studied in depth.

CPSS have the advantages of high immediacy, efficiency, reliability and so on. It is widely used in various aspects. Wang et al. fully described how CPSS is transformed into CPSS, and also introduced the definition, classification and application of CPSS, the contribution and significance of CPSS [13]. Han et al. proposed to introduce dynamic and diverse human behaviors into the vehicle network to make it become a CPSS system and evolve it into a parallel vehicle network to achieve efficient traffic state and low data communication delay between vehicles [2]. Wang et al. proposed a new unified method of CPSS framework based on cloud parallel driving with the purpose of realizing collaborative online automatic driving and carried out parallel testing, learning and reinforcement learning for the framework [14].

Given the massive data in the CPSS scene, it is difficult for traditional methods to deal with these data effectively and timely. Therefore, we apply edge computing to CPSS to solve the above problems. Edge computing introduces the advantages of cloud computing to the network edge cloud in various scenes close to CPSS to provide efficient services [15].

Wang et al. studied the edge server layout in the mobile edge computing environment of smart cities, which is described as a multi-objective (i.e., minimizing the access delay and achieving the load balance) constrained optimization problem [16]. Li et al. studied the energy-aware edge server layout problem, tried to find a more efficient and low-energy layout scheme, and designed an energy-aware edge server layout algorithm based on particle swarm optimization to find the

optimal solution [17]. Zeng et al. studied how to efficient and economic deployment in the wireless metropolitan area network edge server problem, and put forward a kind of based on greedy algorithm as well as a simulated annealing based global optimization method to solve the problem [18].

It is undeniable that there are indeed many achievements and developments in edge computing. However, as far as we know that there is little research have investigated the ES placement strategies. Most studies of this respect have focus on the traffic prediction, load balancing and minimum latency under the conditions of determining certain placement strategy. Compared with the previous works, this paper first presents a task offloading framework based on CPSS. Then server placement method based on CPSS, whose purpose is offering better experience for users in the condition of minimum time consumption and load balance variance, has been designed accordingly. At last, the proof experiments are given in the paper to prove the effectiveness of our method.

3 System Model

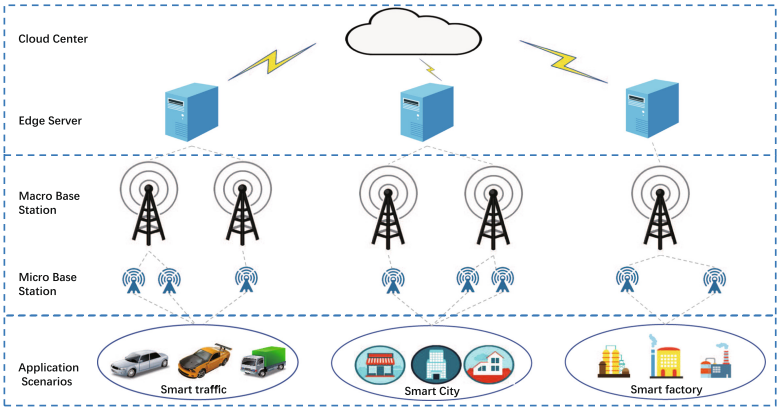

Fig. 1. An edge server placement framework based on CPSS with edge.

The framework in Fig. 1 is composed of user layer, base station layer and the service layer. Three typical scenes are presented in user layer, include smart factory, city and traffic. These scenes tend to generate lots of data which require low latency and high reliability. Such high requirements are difficult to realize in traditional 4G environments. For example, automatic driving in IoV scenario requires time response to reach millisecond level. Thus, the gradual maturation of 5G technology provides a good opportunity for these requirements. All task requests are defined as different tasks which are denoted as $T = \{t_1, t_2, ..., t_{Num}\}$.

Base station layer is divided into micro base station layer and macro base station layer. The former usually receive the data of users and send them to

macro base station in real time. The collection of macro base stations is denoted as $BS = \{bs_1, bs_2, ..., bs_N\}$, and micro base stations collection is defined as $BS^i = \{bs_1^i, bs_2^i, ..., bs_O^i\}$. In 5G scene, the tasks produced from users will transmit to the nearest micro base station, then to macro base station. At last, the tasks will be coped with edge server. Besides, we define that an edge server e_i covers K macro base stations in any location. All the stations are denoted as a collection $\mathrm{B}_{cb}^m = \{b_1^{e^m}, b_2^{e^m}, ..., b_K^{e^m}\}$.

The ES collection is denoted as $E = \{e_1, e_2, ..., e_M\}$. This paper is studying the placement strategies of ES, so a unique collection $PS = \{ps_1, ps_2, ..., ps_U\}$ is defined to store different placement methods. In edge servers, the computing resources are measured by virtual machines (VMs) which are denoted as $V = \{V_m^1, V_m^2, ..., V_m^D\}$. The collection means that the m-th edge server has D VMs.

3.1 Delay Model

This paper assumes that the access delay includes three segments in all. AB_m^n is defined to judge whether the n-th ($n = 1, 2, ..., N$) base station bs_n is combined with the m-th ($i = 1, 2, ..., M$) edge server es_m. N and M represent the largest number of base stations placed and edge servers placed respectively.

$$AB_m^n = \begin{cases} 1, \text{ if } bs_n \text{ combines with } es_m, \\ 0, \qquad\qquad \text{otherwise.} \end{cases} \tag{1}$$

The first segment is transmission delay for transmitting task from the base station to the target edge server, which is defined by

$$TT_n(X) = (1 - AB_m^n) \cdot \frac{ts_n}{tr} \cdot ap_n, \tag{2}$$

where ts_n represents the size of t_n coming from bs_n in the coverage of es_m. Besides, ap_n signifies the amount of the passing base stations in the process of task propagation, and tr on behalf of the propagation rate between base stations.

The second segment is processing time for executing ts_n coming from bs_n, which is calculated by

$$TP_n(X) = \frac{ts_n}{NV_n \cdot \lambda}, \tag{3}$$

where NV_n represents the amount of resource units demanded by the task of ts_n, and λ represents the executing power of the unit in VM.

The third segment is the feedback time of processed result coming from es_m, which is calculated by

$$TF_n(X) = \frac{ts_n'}{tr}, \tag{4}$$

where ts_n' represents the task size of the results unloaded from bs_n.

The total access delay for answering t_n is calculated by

$$OD_n(X) = TT_n(X) + TP_n(X) + TF_n(X). \tag{5}$$

The average access delay for overall ES placement strategy is calculated by

$$AAD = \frac{1}{Num} \cdot \sum_{n=1}^{Num} OD_n(X). \tag{6}$$

3.2 Load Balance Model

In the load balance model, we define the data forwarded by the base station as the tasks that the edge server needs to process [19]. In this section, we assume that each edge server covers the same number of base stations. The task collection in one certain edge server is defined by

$$T_e^i = \sum_{k=1}^{K} t_k, \tag{7}$$

where t_k represent the task number of k-th station covered by i-th server.

The capacity of ES and the computing resource requirement of data are measured by the VMs number. In this scene, the number of VMs required for a certain server is calculated by

$$VM_i = \frac{T_e^i}{R}, \tag{8}$$

where R represents a coefficient calculated jointly by N and M.

The usage of the computing resources in the m-th edge server e_m is measured by

$$U_{cal}^m = \frac{\sum_{r=1}^{7} VM_r}{C}, \tag{9}$$

where C on behalf of the capacity of each edge server.

The average usage of the computing resources in ES is calculated by

$$\mathrm{U_{ave}} = \frac{\sum_{i=1}^{M} U_{cal}^i}{M}. \tag{10}$$

The average load balance variance for the whole ES is calculated by

$$U_{bal} = \frac{\sum_{i=1}^{M} \left(U_{cal}^i - U_{ave}\right)^2}{M}. \tag{11}$$

3.3 Problems Formulation

The aim in this paper is to achieve the minimum delay and to realize the load balance when placing ES around base stations. The multi-objective optimization problem is given as

$$\begin{cases} \min D_{ave}, \\ \min U_{bal}. \end{cases} \tag{12}$$

A few constrains are formalized as follows for acquiring better results by

$$s.t.\ M \leq N, \tag{13}$$

$$\sum_{i=1}^{N} t_{bs_i} \leq C \times M. \tag{14}$$

The constraint (13) means that the number of the ES is less than the number of edge servers. The constraint (14) represents that each active base station has its own server which responsible for handling forwarded tasks, and the sum of the processing power of all servers is greater than the tasks need to be processed.

4 The Design of ESPM

In this section, the elaborative process of designing ESPM is fully presented. Firstly, the process of selecting placement strategies with SPEA2 is designed. Then, the process of identifying placement strategy by TOPSIS and MCDM is presented. Finally, the overview of how to design ESPM is summarized in the last.

4.1 Generating Optimal Placement Strategy

There exists a fact that evolutionary algorithms are widely applied in multi-objective problem. And different algorithms are suitable for different multi-objective scenes. SPEA2 adopts a reasonable fitness allocation strategy and integrates the nearest neighbor density estimation technology, which makes the search process more accurate. In view of this, SPEA2 is used to solve the above problems. This process consists of four stages, i.e., encoding the relevant objective function, transforming the objective function into fitness function and adding constraints, using selection, crossover, mutation and other operators to mutate the algorithm, and finally, generating all the solutions.

Encoding. The objective functions, i.e., to minimize the access delay and load balance variance, to be optimized needs to be mapped into a mathematical problem. Coding is to transform the objective functions into different forms. The purpose of coding is that genetic algorithm can operate the functions easily. There are many ways of coding, different ways determine the efficiency of genetic evolution. In this method, binary coding method is proposed, which is easy to code and decode, and its crossover, mutation and other genetic operations are easy to realize.

Fitness Functions and Constraints. The fitness function is also called the evaluation function, which is a criterion for distinguishing the good or bad of individuals in a group according to the objective function. The specific process includes three processes, i.e., obtain the individual's phenotype according to the processing of the coding part, calculate the objective function value of the individual through the individual's phenotype, obtain the individual's fitness from the objective function value according to the principle of minimization. Besides, the constraint functions are set to constrain the algorithm to a reasonable range. Specific constraints have been shown in (13) and (14).

Selection Operator. Selection operation is utilized to determine how to select those individuals from the parent population in a certain way so as to inherit them to the next generation population. The selection operation is used to determine the recombination or cross individuals, and how many offspring will be generated by the selected individuals. In this method, the expectation selection operator is selected, and the random selection operation is performed according to the survival expectation of each individual in the next generation group.

Crossover Operator. Cross operation refers to the exchange of some genes between two paired chromosomes in a certain way, so as to form two new individuals. In this method, we choose the uniform crossover mode, which can make the genes on each locus of two matched individuals exchange with the same crossover probability, thus forming two new individuals.

Mutation Operator. Mutation operation refers to the replacement of gene values at some points in the coding string of individual chromosomes with other alleles at that point, so as to form a new individual. In this method, the mutation operation is performed on a bit or bits in the individual coding string which are randomly assigned by the mutation probability only because of the value on the seat.

Selecting Optimal Strategy by TOPSIS and MCAM. After generating placement strategies, the TOPSIS and MCAM owing to be selected to generate the optimal strategy. By detecting the distance between the evaluation object and the optimal solution as well as the worst solution, if the evaluation object is the closest to the optimal solution while the farthest from the worst solution, it is the best. Otherwise, it is not the best. Each index value of the optimal solution reaches the optimal value of each evaluation index. Each index value of the worst solution reaches the worst value of each evaluation index.

4.2 The Overview of ESPM

ESPM aims to realize the optimization of the objective functions (6) and (11). The overview of ESPM has showed in Algorithm 1. In this algorithm, the num-

ber of the population is A, the maximum amount of inheritance is B, and the exportation of ESPM is the optimal strategy OS.

Algorithm 1. Obtaining the best strategy by utilizing ESPM

Require: R
Ensure: OS
1: **for** $a = 1$ to A **do**
2: $b = 1$
3: **while** $b <= B$ **do**
4: **for** individuals in population **do**
5: Calculate access dalay time by (6)
6: Calculate load balance variance by (11)
7: Execute the selection, crossover and mutation operators to gengrate better offspring
8: **end for**
9: $b = b + 1$
10: **end while**
11: Obtain the optimal solution by TOPSIS and MCDM
12: **end for**
13: **return** OS

5 Experimental Analysis

In this section, we will compare Benchmark, FFD, BFD and ESPM in the same experimental environment to verify the effectiveness of our method.

5.1 Performance Evaluations on ESPM

Experimental Results on the Average Access Delay. Correspondingly, the average access delay of tasks is calculated. As is shown in the Fig. 2, the average access delay keeps growing as the number of tasks increases. The picture describes that the delay calculated by ESPM is lower than Benchmark, FFD and BFD visually. The average access delay of ESPM are 0.15, 0.25, 0.43, 0.55 and 0.70 (s) when the scale of the tasks are set to 50, 100, 150, 200 and 250.

Experimental Results on the Average Resource Utilization. The average resource utilization, an important indicator, reflects the overall utilization of the system. This indicator directly represents the degree of usage of cells in the virtual machine, and expects to be a high degree. We can see that the Fig. 3 shows the average resource utilization calculated by ESPM and the other three classical methods intuitively. The average resource utilization are 0.74, 0.80, 0.85, 0.88 and 0.92 (s) when the scale of the tasks equal 50, 100, 150, 200 and 250.

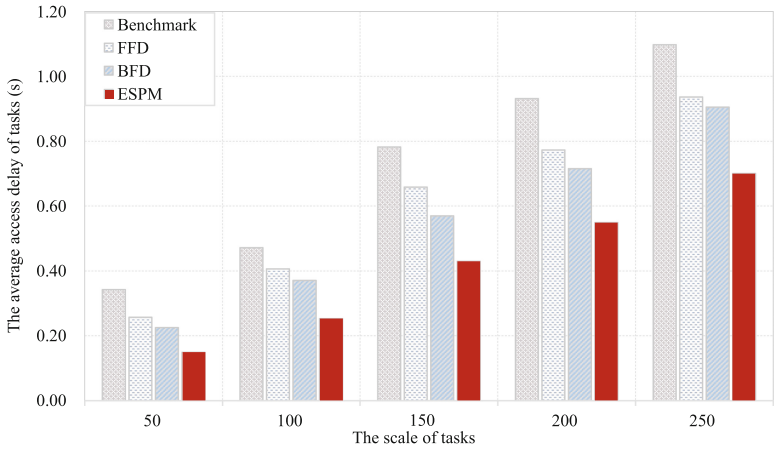

Fig. 2. Experimental results on the average access delay.

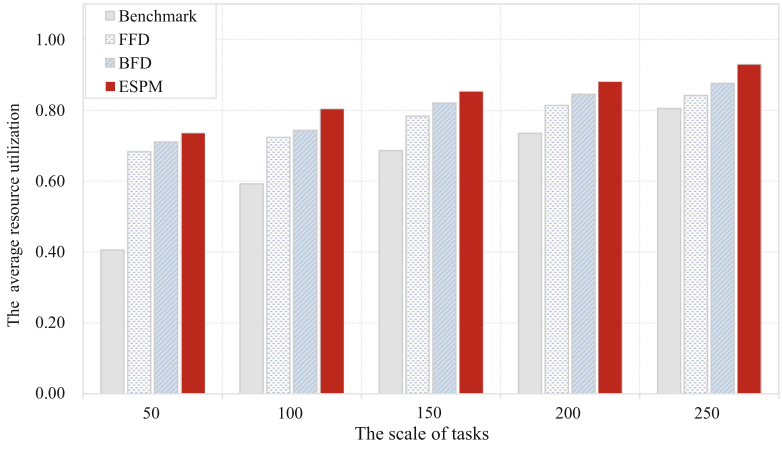

Fig. 3. Experimental results on the average resource utilization.

Experimental Results on the Load Balance Variance. Load balance variance is our main objective function in our paper. Figure 4 shows that the variance calculated by ESPM is lower than the other methods and the variance keeps growing as the number of tasks increases. Besides, the performance of ESPM keeps better than the other methods no matter the number of the tasks. After the detailed statistics, the load balance variance of ESPM are 0.09, 0.31, 0.43, 0.55, and 0.69 when the amount of the tasks are 50, 100, 150, 200 and 250.

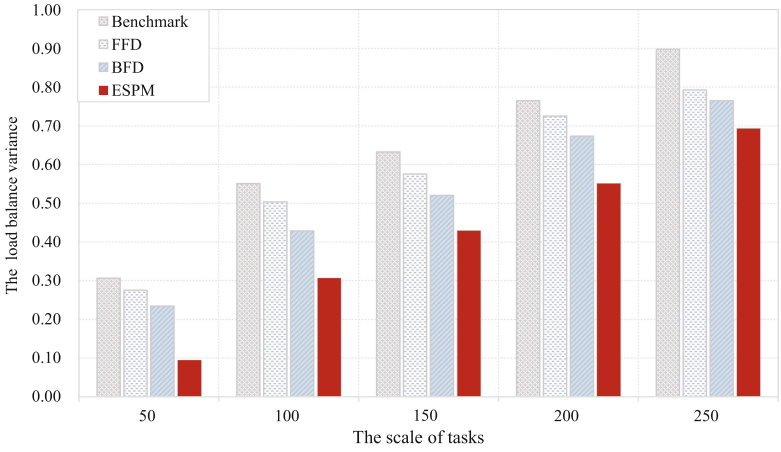

Fig. 4. Experimental results on the load balance variance.

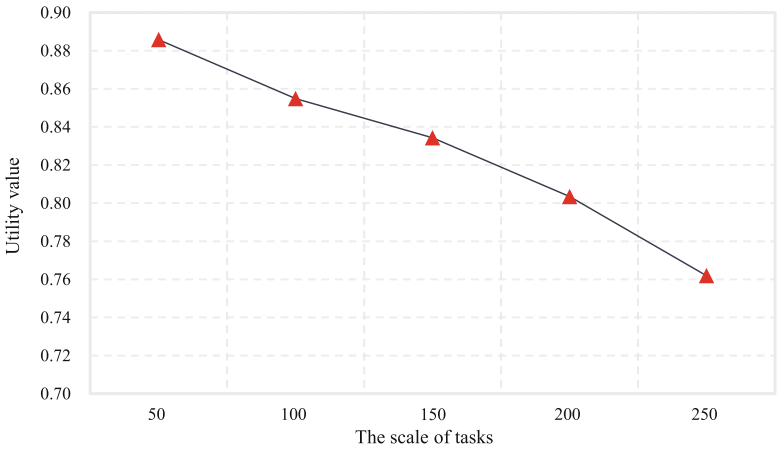

Fig. 5. Experimental results on the utility value.

Experimental Results on the Utility Value. Among all the unload strategies, the one with the highest utility value is the optimal strategy we want to obtain. Figure 5 shows that the highest utility value in different scale will decrease with the increasing of the task scale. Besides, when the scale of the tasks equals 50, the utility value achieves the highest value in all the solutions. The utility value of ESPM are 0.89, 0.85, 0.83, 0.80 and 0.79 when the amount of the tasks are set to 50, 100, 150, 200 and 250.

6 Conclusion

We focus on the problem of edge server placement based on CPSS, where edge computing technique has been applied in it. The placement problem is addressed as an optimization problem with the purpose of minimizing access delay and achieving load balance. Then, the process of designing ESPM is showed in this paper. Besides, the normalization techniques, i.e., TOPSIS and MCDM, are also combined to acquire standardized data. The comparison of experimental results on different dimensions shows the high efficiency of ESPM. For our future work, we intend to use ESPM to compare with the other method in the other literatures to discuss the applicability in practice.

Acknowledgement. This work is supported in part by the National Natural Science Foundation of China under Grant 61601235 and 61872219, in part by the Natural Science Foundation of Jiangsu Province of China under Grant BK20160972.

References

1. Guo, W., Zhang, Y., Li, L.: The integration of CPS, CPSS, and ITS: a focus on data. Tsinghua Sci. Technol. **20**(4), 327–335 (2015)
2. Han, S., Wang, X., Zhang, J.J., Cao, D., Wang, F.Y.: Parallel vehicular networks: a CPSS-based approach via multimodal big data in IoV. IEEE Internet Things J. **6**(1), 1079–1089 (2018)
3. Wang, P., Yang, L.T., Li, J.: An edge cloud-assisted CPSS framework for smart city. IEEE Cloud Comput. **5**(5), 37–46 (2018)
4. Xu, X., et al.: A computation offloading method over big data for IoT-enabled cloud-edge computing. Futur. Gener. Comput. Syst. **95**, 522–533 (2019)
5. Khan, M.A., Salah, K.: IoT security: review, blockchain solutions, and open challenges. Futur. Gener. Comput. Syst. **82**, 395–411 (2018)
6. Xu, X., He, C., Xu, Z., Qi, L., Wan, S., Bhuiyan, M.Z.A.: Joint optimization of offloading utility and privacy for edge computing enabled IoT. IEEE Internet Things J. **7**, 2622–2629 (2019)
7. Wang, F.Y., Yuan, Y., Rong, C., Zhang, J.J.: Parallel blockchain: an architecture for CPSS-based smart societies. IEEE Trans. Comput. Soc. Syst. **5**(2), 303–310 (2018)
8. Karakus, M., Durresi, A.: Quality of service (QoS) in software defined networking (SDN): a survey. J. Netw. Comput. Appl. **80**, 200–218 (2017)
9. Xu, X., et al.: An edge computing-enabled computation offloading method with privacy preservation for internet of connected vehicles. Futur. Gener. Comput. Syst. **96**, 89–100 (2019)
10. Mach, P., Becvar, Z.: Mobile edge computing: a survey on architecture and computation offloading. IEEE Commun. Surv. Tutor. **19**(3), 1628–1656 (2017)
11. 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)
12. Mao, Y., You, C., Zhang, J., Huang, K., Letaief, K.B.: A survey on mobile edge computing: the communication perspective. IEEE Commun. Surv. Tutor. **19**(4), 2322–2358 (2017)

13. Wang, F.: The emergence of intelligent enterprises: from CPS to CPSS. IEEE Intell. Syst. **25**(4), 85–88 (2010)
14. Wang, F.Y., Zheng, N.N., Cao, D., Martinez, C.M., Li, L., Liu, T.: Parallel driving in CPSS: a unified approach for transport automation and vehicle intelligence. IEEE/CAA J. Automatica Sinica **4**(4), 577–587 (2017)
15. Xu, X., et al.: An IoT-oriented data placement method with privacy preservation in cloud environment. J. Netw. Comput. Appl. **124**, 148–157 (2018)
16. Wang, S., Zhao, Y., Xu, J., Yuan, J., Hsu, C.H.: Edge server placement in mobile edge computing. J. Parallel Distrib. Comput. **127**, 160–168 (2019)
17. Li, Y., Wang, S.: An energy-aware edge server placement algorithm in mobile edge computing. In: 2018 IEEE International Conference on Edge Computing (EDGE), pp. 66–73. IEEE (2018)
18. Zeng, F., Ren, Y., Deng, X., Li, W.: Cost-effective edge server placement in wireless metropolitan area networks. Sensors **19**(1), 32 (2019)
19. Xu, Z., Liu, X., Jiang, G., Tang, B.: A time-efficient data offloading method with privacy preservation for intelligent sensors in edge computing. EURASIP J. Wirel. Commun. Netw. **2019**(1), 1–12 (2019). https://doi.org/10.1186/s13638-019-1560-8