# An Enhanced PROMOT Algorithm with D2D and Robust for Mobile Edge Computing

Jin Wang[1,2], Wenbing Wu[1], Zhuofan Liao[1], Yeon-Woong Jung[3], Jeong-Uk Kim[3]

[1] School of Computer & Communication Engineering, Changsha University of Science & Technology, China
[2] School of Information Science and Engineering, Fujian University of Technology, China
[3] Department of Energy-grid, Sangmyung University, South Korea
jinwang@csust.edu.cn, wu_wenbing@163.com, zfliao@csust.edu.cn, jyw@afm.co.kr, jukim@smu.ac.kr

## Abstract

With the development of the fifth-generation network (5G), the base station is becoming denser and denser, which leads to the user device is covered by multiple edge servers in mobile edge computing. Based on this, user devices have more options to decide on which server to offload the tasks. However, there may be still a few user devices located on the edge of a region that is not covered by any edge servers. The user device can only execute tasks locally resulting in excessive latency or energy consumption. To solve the problem, in this paper, an Enhanced PRiori Offloading Mechanism with joint Offloading proportion and Transmission (EPROMOT) power algorithm is proposed. Firstly, a mobile edge computing (MEC) model with device-to-device (D2D) technology is proposed. Then a tradeoff problem consists of the overhead of latency and energy consumption is formulated. Next, a Genetic algorithm is adopted to resolve the tradeoff problem. Besides a prevention mechanism is proposed to increase the robust when the edge server is shut down during the offloading time slot. Finally, experiments have performed to show the outperformance of the EPROMOT algorithm.

**Keywords:** Device-to-device (D2D) technology, Genetic algorithm, Prevention mechanism

## 1 Introduction

In the era of the fifth-generation network (5G) and the Internet of Things (IoT), user devices provide convenience in all aspects of our lives [1-2]. Many advanced applications have also evolved. Nevertheless, limited computing resources, power battery, and storage capacity for most devices cannot meet the demands of computation-intensive and delay-intensive applications. In that case, the quality of experience (QoE) is hard to satisfy.

To solve the above problem, mobile edge computing (MEC) is developed rapidly in those years. By adopting MEC. MEC offers Cloud capabilities for the user device through the backbone. In this case, those computation-intensive [3-4] and delay-intensive tasks [5] can be offloaded to the MEC server in binary or partially [6-8] and cached in the edge server as well [9]. Based on this, a lot of applications and technology have been further developed [10-15].

The base station is becoming denser and denser due to the 5G era, which leads to the user device is multi-covered in mobile edge computing. Since the user device can offload tasks to the edge server, the user devices have more options to decide on which server to offload the tasks. Based on this, the user device will waste lots of workloads to decide which server to offload the task in the task offloading decision phase. The previous work [16] proposed a distributed PROMOT (PRiori Offloading Mechanism with joint Offloading proportion and Transmission) power algorithm to reduce the workload during the task offloading decision phase.

However, the previous work [16] hasn't considered the situation that there may be still a few user devices located on the edge of the region that is not covered by any edge server under the multi-cover scenario. The user device has to execute tasks locally if it is not covered, which can lead to excessive and unaccepted latency and energy consumption. Besides, if there is an edge server shuts down during the offloading time slot, the tasks of the user device can be all failed.

In the paper, an enhanced PROMOT (EPROMOT) algorithm is proposed to solve the mentioned above problems. In the algorithm, the device-to-device (D2D) technology is adopted. If the user device is not covered, the user device will offload tasks to the D2D device by an appropriate offloading proportion and transmission power. Besides, a prevention mechanism is proposed to improve the robust considering that the edge server may be shut down during the time slot. The main contribution is as follows:

- We conduct a D2D-enabled edge computing system, in which most user devices are covered by several edge servers and the rest are not covered by any

*Corresponding Author: Jeong-Uk Kim; E-mail: jukim@smu.ac.kr
DOI: 10.3966/160792642020092105018

edge servers. D2D technology is adopted to improve the performance of the uncovered user devices.

- A tradeoff problem consists of the overhead of latency and energy consumption is formulated. To deal with the problem, An Enhanced PRiori Offloading Mechanism with joint Offloading proportion and Transmission (EPROMOT) power algorithm is proposed to improve the system performance.
- A prevention mechanism is proposed to further improve the system performance when the edge server shuts down during the offloading procedure.
- Simulations are carried out to evaluate the performance which proved efficient of EPROMOT algorithm.

## 2 System Model

The system model is depicted in Figure 1, we assume that there is a region that consists of M edge server and N user devices, and the location of the user device in the region is random. Most user devices are covered by several edge servers and the rest are not covered by any edge servers. The user device is connected to a nearby device, called D2D device, through a D2D connection. A user device is paired with only one D2D device. We take in our consideration that every user device submits their tasks with the arrival rate $\lambda_i$ as stated by the process of Poisson [17]. A quad $<S_i, C_i, T_i^{\max}, E_i^{\max}>$ is used to represent the computation task, where $S_i$ denotes the size of task *i*, $C_i$ denotes the necessary needed Central Processing Unit (CPU) cycles to complete the task, $T_i^{\max}$ and $E_i^{\max}$ denotes the completion time and energy of task *i* that user device can be tolerated, respectively. Task can be split into several parts and offloaded due to adopted partial offloading model.

The edge server adopts the heterogeneity M/M/1 queue model with different service rates $SR_j$.

### 2.1 Local Execution Model

We first introduce the local execution model. We use $Pro_{i,l}$ denotes the proportion that task *i* left in local user devices for executing. Let assume that the $f_{i,l}$ denotes the capacity of local user device *i*. The latency of task *i* executed in the local user device can be defined as follows:

$$t_{i,l}^{exe} = \frac{Pro_{i,l} C_i}{f_{i,l}} \tag{1}$$

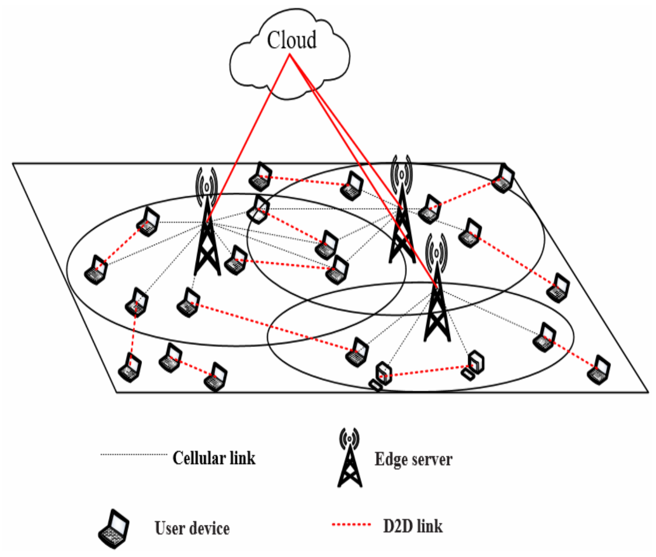


**Figure 1.** The system model

The widely used model as $\varepsilon = \kappa f^2$ [18] is adopted to calculate the local energy consumption locally, where $\kappa$ is the energy coefficient and $f$ denotes the frequency of CPU. Therefore, the energy consumption of task *i* executed locally can be obtained as follows:

$$e_{i,l}^{comp} = \kappa f_{i,l}^2 Pro_{i,l} C_i \tag{2}$$

### 2.2 Remote Execution Model

The remote execution model includes the D2D execution model and the edge computing model. All overhead is generated by the transmission and execution procedure. But the overhead of computation result feedback can be neglect, as the data size is much smaller than the original size [19].

To calculate the overhead during the transmission phase, the transmission rate needs to be obtained first. The Shannon formula [20] with device and background noise interference is adopted. Then the transmission rate of user device *i* transfer tasks to can be estimated as follows

$$R_{i,j} = B\log_2\left(1 + \frac{Po_i h_{i,j}}{N_0 + \Sigma_{i' \in k \setminus i} Po_{i'} h_{i',j}}\right) \tag{3}$$

where $B$ is the bandwidth of the channel and $N_0$ is the noise interference of background. $Po_i$ ($0 < Po_i \le Po_{\max}$) denoted as the transmission power of the user device *i* within the maximum transmission power $Po_{\max}$, $h_{i,j}$ denoted as the gain of the channel between the user device *i* and the offloading carrier (the edge server or the D2D user device). The remaining items in the denominator are interference between user devices

#### 2.2.1 D2D Execution Model

Then we calculate the overhead when a part of task *i* is offloaded to the D2D device, the notation $Pro_{i,d}$ is denoted as the proportion of the task *i* offloaded to the D2D device. The transmission latency can be obtained as follows:

$$t_{i,d}^{trans} = \frac{Pro_{i,d}S_i}{R_{i,d}} \quad \textbf{(4)}$$

And the energy consumption when task *i* transferred to the nearby user device is also obtained as follows

$$e_{i,d}^{trans} = t_{i,d}^{trans}Po_i = Po_i\frac{Pro_{i,d}S_i}{R_{i,d}} \quad \textbf{(5)}$$

When the transmission procedure of the subtask of task *i* to the D2D device is completed, the subtask of the task *i* will execute in the D2D device. Thus, we can obtain the latency when the subtask executed in D2D device as follows

$$t_{i,d}^{comp} = \frac{Pro_{i,d}C_i}{f_{i,d}} \quad \textbf{(6)}$$

where $f_{i,d}$ is the computing capacity of the D2D device of user device *i*.

And the energy consumption of task *i* can be obtained when task offloaded to the D2D device as the follows

$$e_{i,d}^{com} = p_{i,d}^{com}t_{i,d}^{trans} \quad \textbf{(7)}$$

where $p_{i,d}^{com}$ represents the energy consumption level [21-22]. We assume that the D2D devices can receive the offloaded task for having far enough capacity to execute their task.

### 2.2.2 Edge Server Execution Model

The user device can offload its tasks to the connected MEC server to increase the performance. However, the user device at the edge of the region maybe not covered by any MEC server when the edge server location is fixed. Thus, a binary variable is been used to denote the user device *i* is covered by a MEC server or not as follows

$$X_i = \begin{cases} 1, & \textit{The user device is coverd} \\ 0, & \textit{Otherwise} \end{cases} \quad \textbf{(8)}$$

To improve the efficiency of the task offloading decision making, the mathematical expectation is used in this paper. For each connected edge server of the user device, the offloading probability when user device *i* offload its tasks can be calculated as follows

$$prob_{i,j} = X_i\frac{SR_j}{\sum_k SR_k} \quad \textbf{(9)}$$

where $SR_j$ denotes the service rate of the MEC server *j*, the total number of MEC servers that user device *i* connected is k.

When the probability has been calculated, the expectation of transmission latency can be calculated. Let's assume that the proportion of the task i offloaded to the edge server is $Pro_{i,e}$, then the transmission time expectation of task i can be estimated as follow

$$t_{i,e}^{trans} = X_i\sum_k t_{i,k}^{trans} = X_i\sum_k prob_{i,k}\frac{Pro_{i,e}S_i}{R_{i,k}} \quad \textbf{(10)}$$

After the transmission time expectation of the task *i* offloaded to the MEC server is been calculated, the energy consumption can be obtained as well. Let's notate that the energy consumption expectation for transmitting the data from the user device *i* to the edge server as $e_{i,e}^{trans}$ follows:

$$e_{i,e}^{trans} = X_i\sum_k prob_{i,k}Po_i t_{i,k}^{trans} \quad \textbf{(11)}$$

Then the widely used queuing theory is applied in edge computing [16-17, 23], and each edge server upholds a heterogeneity M/M/1 queue. As shown in Figure 2.

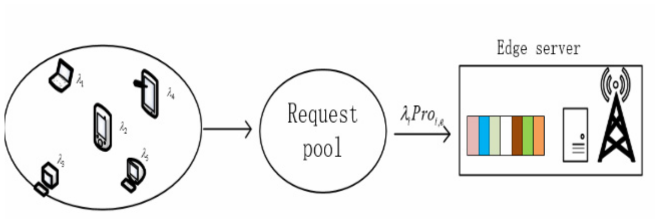


**Figure 2.** The conceptual structure of the offloading procedure in the edge server

According to the M/M/1 queuing model, the latency of task *i* executed in MEC server *j* can be found in the following equation:

$$t_{i,j}^{exe} = X_i\frac{1}{Sr_j - Pro_{i,e}\lambda_i} \quad \textbf{(12)}$$

The execution time of task *i* is uncertain until the user device decides which edge server to offload tasks to. Thus, we use the expectation value as the execution time of task *i*. We can obtain the expected time of task executing as follow

$$t_{i,e}^{exe} = X_i\sum_k prob_{i,k}t_{i,k}^{exe} = X_i\sum_k prob_{i,k}\frac{1}{Sr_k - Pro_{i,e}\lambda_i} \quad \textbf{(13)}$$

Then, the expectation value of execution energy consumption is shown in the following equation

$$e_{i,e}^{comp} = \varpi t_{i,e}^{exe} \quad \textbf{(14)}$$

where $\varpi$ is the energy coefficient of edge computing server.

## 2.3 Problem Formulation

Through the above analysis, we can know that the latency of completing a task is included three-part: local execution latency, D2D execution latency, and the edge server execution latency. However, the offloading procedure is distributed, which means all three-part are synchronized. The total latency of executing the whole task is the largest time consumption within the above three parts. The total delay of task *i* when offloading its task in a distributed way is specified by

$$T_i^{total} = \max\{t_{i,l}^{exe}, t_{i,d}^{trans} + t_{i,d}^{exe}, t_{i,e}^{trans} + t_{i,e}^{exe}\} \quad \textbf{(15)}$$

Although the above three parts are running simultaneously, the energy consumption of executing the entire task *i* is the sum of the three parts of energy consumption. The energy consumption of task *i* can be described as follows

$$E_i^{total} = e_{i,l}^{comp} + e_{i,d}^{trans} + e_{i,d}^{comp} + e_{i,e}^{trans} + e_{i,e}^{comp} \quad \textbf{(16)}$$

The energy consumption and delay is usually used as the criteria of users' Quality of Experience (QoE), thus a tradeoff offloading utility consist of energy consumption and latency is designed as follow

$$\begin{aligned} &utility_i(Pro_{i,l}, Pro_{i,d}, Pro_{i,e}, Po_i) \\ &= \alpha\frac{T_i^{\max} - T_i^{total}}{T_i^{\max}} + (1-\alpha)\frac{E_i^{\max} - E_i^{total}}{E_i^{\max}} \end{aligned} \quad \textbf{(17)}$$

where $0 < \alpha < 1$ is the user's preference on latency and energy consumption, the network operator can set the value of $\alpha$ according to the needs.

In this paper, the maximization system utility of the considered D2D-enabled MEC system is our goal, in which the system consists of covered and uncovered user devices. As the system utility is a tradeoff between latency and energy consumption, we can obtain the optimization problem as follow

$$\max_{Pro_{i,l}, Po_i, Pro_{i,d}, Pro_{i,e}} U = \sum_{i=1}^{M} utility_i(Pro_{i,l}, Pro_{i,d}, Pro_{i,e}, Po_i) \quad \textbf{(18)}$$

$$\text{subject to } Pro_{i,l} + Pro_{i,d} + Pro_{i,e} = 1 \quad \textbf{(18a)}$$

$$0 < Pro_{i,l}, Pro_{i,d}, Pro_{i,e} < 1 \quad \textbf{(18b)}$$

$$0 < Po_i \le Po_{\max} \quad \textbf{(18c)}$$

$$E_i^{total} < E_i^{\max} \quad \textbf{(18d)}$$

$$T_i^{total} < T_i^{\max} \quad \textbf{(18e)}$$

$$X_i = \{0,1\} \quad \textbf{(18f)}$$

The constrictions in the formulation above can be clarified as given: constraint (18a) and (18b) implies that the whole task divides into three sub-task and the proportion must greater than 0 and less than 1, constraint (18c) indicate that the transmission power that adopted during the offloading procedure must within the maximum transmission power of the user device, constraint (18d) and constraint (18e) indicate that the energy consumption and latency cannot exceed the maximum tolerate energy consumption latency, constraint (18f) indicate that the user device is covered or not, there is not the third situation comes out.

## 3 Algorithm Design

### 3.1 EPROMOT Algorithm

Through the analysis of the system model, the above optimization problems can be divided into two problems in different situations.

Situation 1: the user device is not covered with any edge servers. For that case, the binary variable X is equal to 0. The offloading proportion of edge server $Pro_{i,e}$ is equal to 0 due to the user device cannot offload tasks to the edge server. And the user device can only offload tasks to the connected D2D device. The optimization problem can be formulated as follows:

$$\max_{Pro_{i,l}^0, Po_i^0, Pro_{i,d}^0} U = \sum_{i=1}^{Un} utility_i(Pro_{i,l}^0, Pro_{i,d}^0, Po_i^0) \quad \textbf{(19)}$$

where the variable *Un* is denoted as the number of uncovered user devices. The superscript of the notation 0 indicates the optimization offloading proportion and transmission power of situation 1.

Situation 2: the user device is covered with several edge servers. For that case, the binary variable X is equal to 1. It means that the user device can offload tasks not only to the D2D device but also to the connected edge server. The optimization problem can be formulated as follows:

$$\max_{\substack{Pro_{i,l}^1, Po_i^1 \\ Pro_{i,d}^1, Pro_{i,e}^1}} U = \sum_{i=1}^{M-Un} utility_i(Pro_{i,l}^1, Pro_{i,d}^1, Pro_{i,e}^1, Po_i^1) \quad \textbf{(20)}$$

where the superscript of the notation 1 indicates the optimization offloading proportion and transmission power of the situation 2.

The above problem is a multi-objective optimization problem, and we proposed EPROMOT power algorithm to solve the problem. The widely used Genetic Algorithm (GA) is adopted [24] in the EPROMOT algorithm. We first initialize the parameters in the GA algorithm, then encode the optimization problem and evaluate the fitness value. Through iterative selection, crossover, and mutation operation, an optimal solution is finally obtained, which is the best transmission power and offloading proportion of the EPROMOT algorithm.

After the optimization solution is obtained, the user device will generate several intervals according to the offloading probability of each connected edge server [16]. There is no need to generate intervals if the user device is not covered. Then the user device can offload tasks according to the optimization solution.

The flowchart is shown in Figure 3 and the pseudocode is shown in Algorithm 1.

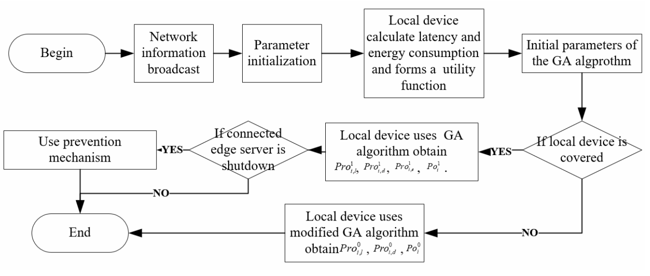

Begin
Network information broadcast
Parameter initialization
Local device calculate latency and energy consumption and forms a utility function
Initial parameters of the GA algprothm
If local device is covered
YES
Local device uses GA algorithm obtain $Pro_{i,l}^1$, $Pro_{i,d}^1$, $Pro_{i,e}^1$, $Po_i^1$.
If connected edge server is shutdown
YES
Use prevention mechanism
NO
NO
Local device uses modified GA algorithm obtain $Pro_{i,l}^0$, $Pro_{i,d}^0$, $Po_i^0$
End

**Figure 3.** The flowchart of the EPROMOT algorithm

**Algorithm 1.** Proposed EPROMOT algorithm

**Input:** local device information, D2D device information, edge server information, network information, the initial parameters for the GA algorithm.

**Output:** $Pro_{i,l}$, $Pro_{i,d}$, $Pro_{i,e}$, $Po_i$

```
1.  Initialize the local devices, D2D devices, edge servers.
2.  Broadcast the information between local devices, D2D devices, edge servers.
3.  The local device i calculate the task overhead and formulate an optimization problem
4.  Initialize the parameters for the GA algorithm.
5.  If local device i connect edge server, then
6.  Use the optimization problem as fitness function and adopt the GA algorithm to obtain $Pro_{i,l}^1$, $Pro_{i,d}^1$, $Pro_{i,e}^1$, $Po_i^1$ as the output.
7.  The user device generates different intervals between 0 and 1 according to the offloading probability. Each interval denotes the select interval of an edge server.
8.  The user device generates a random number, and select an edge server offloads task based on the interval of 0-1 where the random number belongs.
9.  If the local device isn't connected with any edge servers, then
10. Use the optimization problem as fitness function and adopt the modified GA algorithm to obtain $Pro_{i,l}^0$, $Pro_{i,d}^0$, $Po_i^0$ as the output.
11. User device i offloads tasks according to the output.
```

## 3.2 Prevention Mechanism

Notice that EPROMOT algorithm is continuously run at the start of a time slot, this means that the transmission power and offloading proportion keep stable during the offloading time slot. When the edge server is shut down during the time slot, we divide this situation into two cases:

Case 1: the user device is only connected with one edge server which is shut down. It means that the user device can only offload tasks to the connected D2D device. Then the task will be offloaded to the D2D device according to the output of EPROMOT which the superscript is 0. This the reason that we save the offloading proportion and transmitted power when only the D2D device is the offloading carrier even though the edge server is connected.

Case 2: the user device is connected with multiple edge servers. When the connected servers are shut down, the user device can choose another edge server to offload tasks. We use the following rule to choose the offloading edge server:

$$.\max_{k-1} utility_i(Pro_{i,l}^1, Pro_{i,d}^1, Pro_{i,e}^1, Po_i^1) - utility_i(Pro_{i,l}^0, Pro_{i,d}^0, Po_i^0) \quad \textbf{(21)}$$

For a given best transmission power and offloading proportion, we calculate the difference value in system utility between offloading the task directly to the D2D device and offloading it to the remaining $k-1$ edge servers. If the difference value is less than 0, then we offload tasks only on D2D devices, because offloading to D2D devices has a higher system utility. If the difference value is greater than 0, we select the edge server with the largest difference value as our backup offloading server. The following Figure 4 is the flowchart of the prevention mechanism.

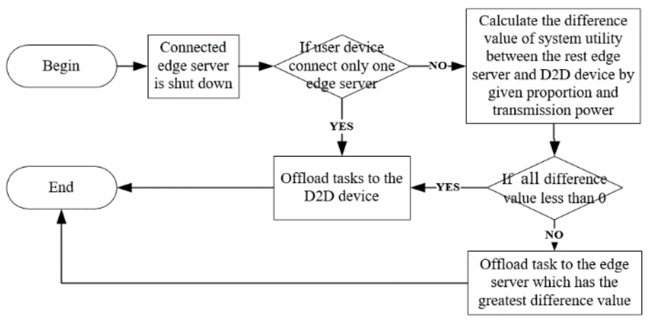

Begin
Connected edge server is shut down
If user device connect only one edge server
NO
Calculate the difference value of system utility between the rest edge server and D2D device by given proportion and transmission power
YES
If all difference value less than 0
YES
Offload tasks to the D2D device
NO
Offload task to the edge server which has the greatest difference value
End

**Figure 4.** The flowchart of the prevention mechanism

## 4 Performance Evaluation

Numerical results are introduced to measure the performance of the proposed EPROMOT, as compared to the following two benchmark systems.

- The Offloading Proportion optimization and the Queue theory adopted with Random power allocation (OPQR) algorithm: this scheme only optimizes the offloading proportion, the transmission power is randomly chosen under the maximum transmission power. And the queue theory is adopted.
- The previous PROMOT scheme [16]: this scheme optimized the transmission power and offloading proportion, the queue theory is been also adopted. However, the uncovered user device located at the edge of the region is not considered into system optimized.

We usually focus on the offloading proportion when adopting the partial offloading model, the transmission power allocation is omitted. We can analyze the impact of the transmission power on the system utility by comparing scheme 1. And we can further analyze the impact of the D2D device on the uncovered user device by comparing the scheme 2.

Those several edge servers and user devices are installed in a range of 50m × 50m region. Edge servers, user devices, and D2D devices can communicate by an antenna. The tolerate latency and energy consumption we assumed is 15J and 2s respectively [16]. The main parameters used in this paper can be shown in Table 1. A windows operating system that contains CPU with dual-core, eight gigabytes (GB) of memory and one terabyte of second storage memory is been used as the simulation environment.

**Table 1.** Simulation parameters

| Parameter Name | Value |
|---|---|
| $\lambda_i$ | 1.5 MIPS |
| $f_{i,l}$ | 1GHz |
| $\kappa$ | $5\times10^{-27}$ |
| $\varpi$ | 16 [16] |
| $p_{i,d}^{com}$ | 36dBm [21] |
| $B$ | 20MHz |
| $N_0$ | -110dBm |
| $Po_{\max}$ | 20dBm |
| $h_{i,j}$ | $140.7+36.7\log_{10} d$ [25] |
| $SR$ | 10~15MIPS |

We now evaluate the average system utility performance under the different number of user devices. As shown in Figure 5, we compare the number of user devices from 10 to 100. The EPROMOT algorithm always performs best than the other two schemes. This because the other two schemes didn't consider the situation that there will be some user device located at the edge of a region that may not be covered, which causes the user device to generate excessive overhead.

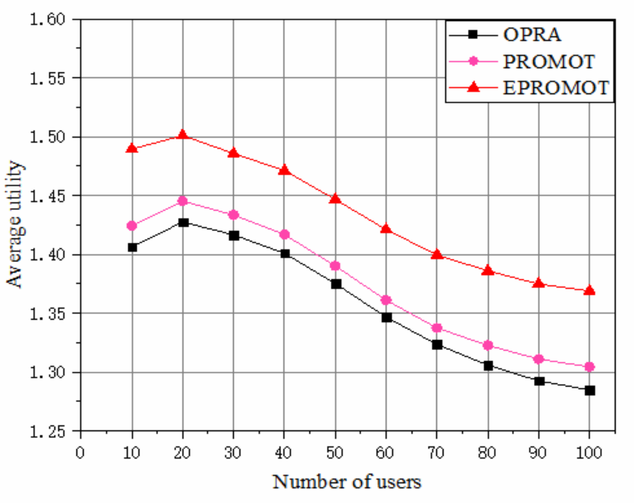


**Figure 5.** Comparison of average system utility against the different number of users

Then we discuss the impact of the D2D device capacity on the system utility. Observe from Figure 6, the trend of EPROMOT algorithm curve is upward, and the curve of the other two schemes, in general, keeps stable. This because the EPROMOT algorithm adopted the D2D technology to solve the user device which not covered by edge server located at the edge of the region for improving the system utility. The D2D devices have high capacity mean those devices have high processing probability.

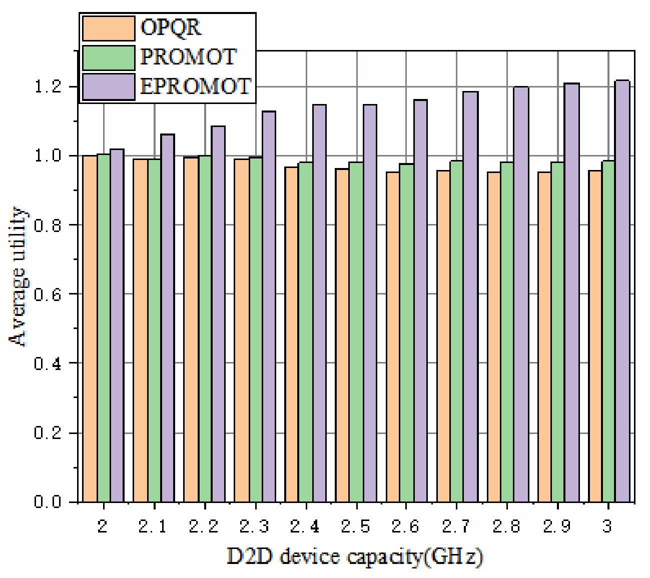


**Figure 6.** Comparison of average system utility against the different capacity of the D2D device

Figure 7 shows the impact of the increasing number of uncovering user devices on average system utility. In Figure 7, the gap between those three curves becomes larger and larger, and the general trend is downward. This because when the user device cannot offload the tasks to the edge server, the EPROMOT algorithm can lead the user device offloads task to the D2D device. In that case, the user device can go on offloading tasks.

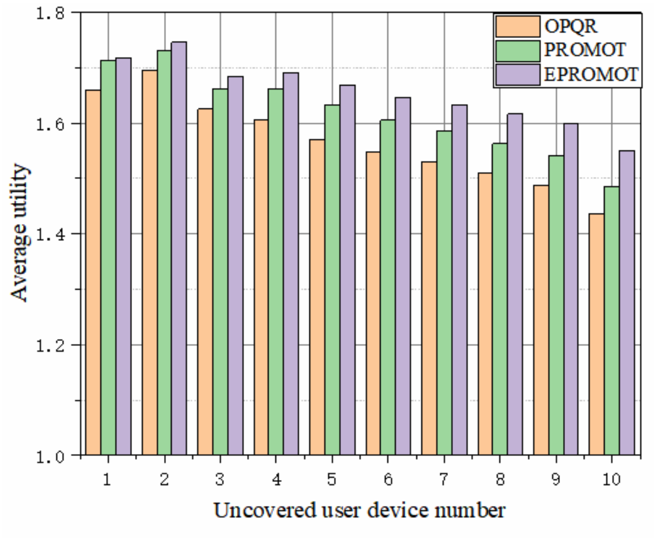


**Figure 7.** Comparison of average system utility against different number of the uncovered user device

We discuss the impact of the edge server keeps increasing shut down on average system utility in Figure 8, the number of user devices we simulate is 20. As can be observed, the average utility keeps down with the decreasing of the edge server shut down. However, the average utility of EPROMOT algorithm is always having a better outperformance than the other two schemes. Based on this, it is sufficient to prove that the prevention mechanism we proposed is of good usability.

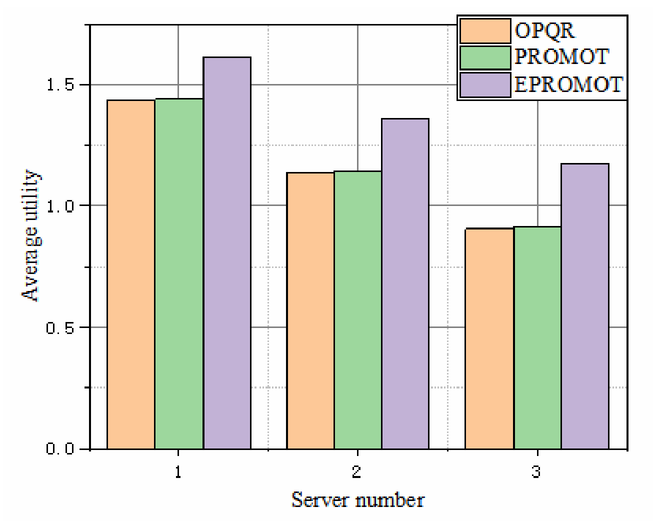
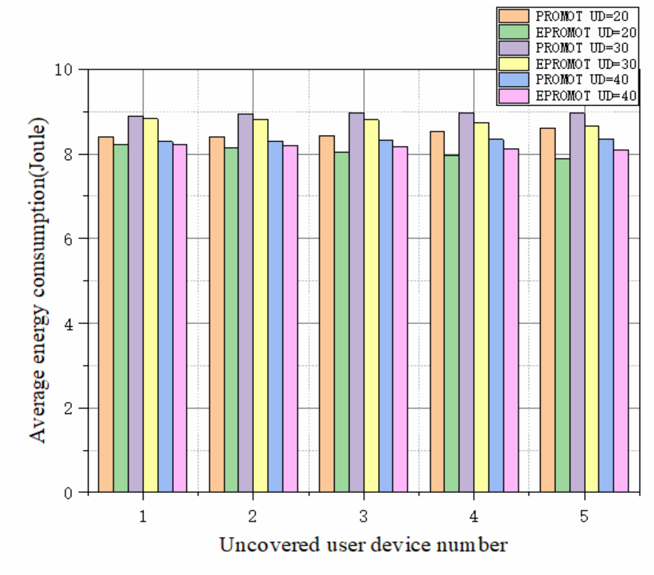
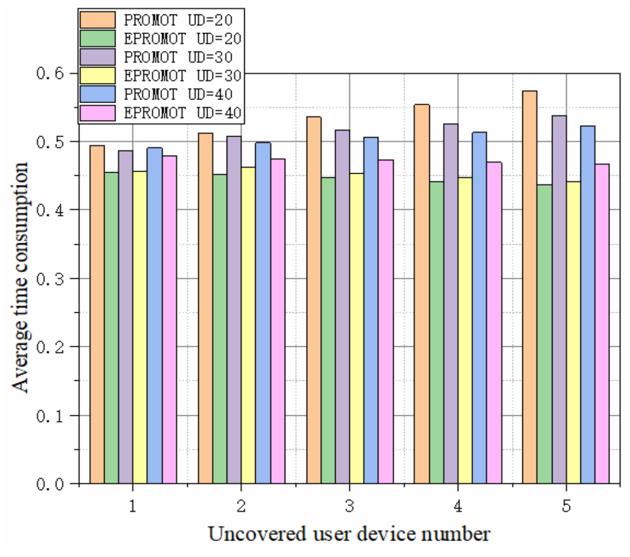


**Figure 8.** Comparison of average system utility against different number of the edge server shut down

We now evaluate the impact of the increasing number of uncovered user devices on latency and energy consumption. Figure 9(a, b) show the average time and energy consumption when we vary the uncovered users device under the PROMOT algorithm and EPROMOT algorithm. From Figure 9(a), we can see that the general trend is upward, which means the more uncovered user device, the more energy consumption. Although tasks can be offloaded to the D2D device in the EPROMOT algorithm, the energy consumption is increasing with the increasing of uncovered user devices. It can also observe that the gap of energy consumption between the RPOMOT algorithm and the EPROMOT is larger and larger with the increasing of uncovered user devices. Observe from Figure 9(b), the gap of latency between the RPOMOT algorithm and the EPROMOT is larger and larger with the increasing of uncovered user devices. The reason for that in the two algorithms is that the processing ability of the D2D device is lower than the edge server.



(a) Average energy consumption



(b) Average time consumption

**Figure 9.** Comparison of average system utility against the different numbers of the uncovered user device, evaluated on the different number of the user device of PROMOT algorithm and EPROMOT algorithm

## 5 Conclusion

To solve the user devices located on the edge of a region which may not be covered and improve the system utility, we proposed an Enhanced PRiori Offloading Mechanism with joint Offloading proportion and Transmission (EPROMOT) power algorithm. In the EPROMOT algorithm, we adopted the D2D technology to improve the performance of the uncovered user device. And a Genetic algorithm is used to find the optimization offloading proportion and transmission power between the local device, MEC server, and D2D device. To improve the system utility when the edge server is shut down during the offloading procedure, a prevent mechanism is proposed, in which the user can choose another edge server to offload tasks or just offload tasks to the D2D device. The simulation result shows the outperformance compare to other benchmarks from several aspects.

## Acknowledgments

This work was supported by the National Natural Science Foundation of China (61772454), and by the 'Human Resources Program in Energy Technology' of the Korea Institute of Energy Technology Evaluation and Planning (KETEP), granted financial resource from the Ministry of Trade, Industry and Energy, Republic of Korea (No. 20164030300230). Prof. Jeong-Uk Kim is the corresponding author.

## References

[1] P. Porambage, J. Okwuibe, M. Liyanage, M. Ylianttila, T. Taleb, Survey on Multi-Access Edge Computing for Internet of Things Realization, *IEEE Communications Surveys & Tutorials*, Vol. 20, No. 4, pp. 2961-2991, Fourthquarter, 2018.

[2] J. Wang, Y. Yang, T. Wang, R. Sherratt, J. Zhang, Big Data Service Architecture: A Survey, *Journal of Internet Technology*, Vol. 21, No. 2, pp. 393-405, March, 2020.

[3] J. Wang, Y. Zou, P. Lei, R. Sherratt, L. Wang, Research on Recurrent Neural Network based Crack Opening Prediction of Concrete Dam, *Journal of Internet Technology*, Vol. 21, No. 4, July, 2020.

[4] J. Wang, Y. Tang, S. He, C. Zhao, P. K. Sharma, O. Alfarraj, A. Tolba, LogEvent2vec: LogEvent-to-Vector based Anomaly Detection for Large-Scale Logs in Internet of Things, *Sensors*, Vol. 20, No. 9, 2451, May, 2020.

[5] J. Wang, Y. Gao, C. Zhou, R. Sherratt, L. Wang, Optimal Coverage Multi-Path Scheduling Scheme with Multiple Mobile Sinks for WSNs, *Computers, Materials & Continua*, Vol. 62, No. 2, pp. 695-711, 2020.

[6] Y. Mao, C. You, J. Zhang, K. Huang, K. B. Letaief, A Survey on Mobile Edge Computing: The Communication Perspective, *IEEE Communications Surveys & Tutorials*, Vol. 19, No. 4, pp. 2322-2358, Fourthquarter, 2017.

[7] P. Mach, Z. Becvar, Mobile Edge Computing: A Survey on Architecture and Computation Offloading, *IEEE Communications Surveys & Tutorials*, Vol. 19, No. 3, pp. 1628-1656, Thirdquarter, 2017.

[8] N. Abbas, Y. Zhang, A. Taherkordi, T. Skeie, Mobile Edge Computing: A Survey, *IEEE Internet of Things Journal*, Vol. 5, No. 1, pp. 450-465, February, 2018.

[9] J. Yao, T. Han, N. Ansari, On Mobile Edge Caching, *IEEE Communications Surveys & Tutorials*, Vol. 21, No. 3, pp. 2525-2553, Thirdquarter, 2019.

[10] J. Zhang, S. Zhong, T. Wang, H. C. Chao, J. Wang, Blockchain-Based Systems and Applications: A Survey, *Journal of Internet Technology*, Vo. 21, No. 1, pp. 1-14, January, 2020.

[11] S. He, K. Xie, X. Zhou, T. Semong, J. Wang, Multi-Source Reliable Multicast Routing with QoS Constraints of NFV in Edge Computing, *Electronics*, Vol. 8, No. 10, 1106, October, 2019.

[12] D. Cao, Y. Jiang, J. Wang, B. Ji, O. Alfarraj, A. Tolba, X. Ma, Y. Liu, ARNS: Adaptive Relay-Node Selection Method for Message Broadcasting in the Internet of Vehicles, *Sensors*, Vol. 20, No. 5, 1338, March, 2020.

[13] J. Zhang, C. Wu, D. Yang, Y. Chen, X. Meng, L. Xu, M. Guo, HSCS: A Hybrid Shared Cache Scheduling Scheme for Multiprogrammed Workloads, *Frontiers of Computer Science*, Vol. 12, No. 6, pp. 1090 -1104, December, 2018.

[14] W. Li, H. Xu, H. Li, Y. Yang, P. K. Sharma, J. Wang, S. Singh, Complexity and Algorithms for Superposed Data Uploading Problem in Networks with Smart Devices, *IEEE Internet of Things Journal*, Vol. 7, No. 7, pp. 5882- 5891, July, 2020.

[15] Z. Liao, J. Peng, Y. Chen, J. Zhang, J. Wang, A Fast Q-Learning Based Data Storage Optimization for Low Latency in Data Center Networks, *IEEE Access*, Vol. 8, pp. 90630-90639, May, 2020.

[16] J. Wang, W. Wu, Z. Liao, R. Sherratt, G. Kim, O. Alfarraj, A. Alzubi, A. Tolba, A Probability Preferred Priori Offloading Mechanism in Mobile Edge Computing, *IEEE Access*, Vol. 8, pp. 39758-39767, February, 2020.

[17] L. Q. Liu, Z. Chang, X. J. Guo, S. W. Mao, T. Ristaniemi, Multiobjective Optimization for Computation Offloading in Fog Computing, *IEEE Internet of Things Journal*, Vol. 5, No. 1, pp. 283-294, February, 2018.

[18] Q. Tang, H. M. Lyu, G. J. Han, J. Wang, K. Z. Wang, Partial Offloading Strategy for Mobile Edge Computing Considering Mixed Overhead of Time and Energy, *Neural Computing and Applications*, August, 2019, Doi: 10.1007/s00521-019-04401-8

[19] X. Chen, L. Jiao, W. Z. Li, X. M. Fu, Efficient Multi-User Computation Offloading for Mobile-Edge Cloud Computing, *IEEE/ACM Transactions on Networking*, Vol. 24, No. 5, pp. 2795-2808, October, 2016.

[20] T. S. Rappaport, *Wireless Communications: Principles and Practice*, Prentice-Hall, 1996.

[21] B. Gu, Y. P. Chen, H. J. Liao, Z. Y. Zhou, D. Zhang, A Distributed and Context-Aware Task Assignment Mechanism

for Collaborative Mobile Edge Computing, *Sensors*, Vol. 18, No. 8, 2423, August, 2018.

[22] J. Wang, W. Wu, Z. Liao, A. K. Sangaiah, R. Sherratt, An Energy-efficient Off-loading Scheme for Low Latency in Collaborative Edge Computing, *IEEE Access*, Vol. 7, pp. 149182-149190, October, 2019.

[23] K. Li, Computation Offloading Strategy Optimization with Multiple Heterogeneous Servers in Mobile Edge Computing, *IEEE Transactions on Sustainable Computing*, March, 2019, Early Access.

[24] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II, *IEEE Transactions on Evolutionary Computation*, Vol. 6, No. 2, pp. 182-197, April, 2002.

[25] T. Tran, D. Pompili, Joint Task Offloading and Resource Allocation for Multi-Server Mobile-Edge Computing Networks, *IEEE Transactions on Vehicular Technology*, Vol. 68, No. 1, pp. 856-868, January, 2019.

## Biographies

**Jin Wang** (Senior Member, IEEE) received the M.S. degrees from the Nanjing University of Posts and Telecommunications, China, and the Ph.D. degree from Kyung Hee University, South Korea. He is currently a Professor with the Changsha University of Science and Technology. His research interests include wireless sensor networks, and network performance analysis and optimization.

**Wenbing Wu** received the B.S. degree in Hunan Agricultural University, China, in 2018. He is a postgraduate student in Changsha University of Science and Technology, Changsha, China. His research interests include mobile edge computing, fog computing, machine learning. He is good at Python and C++.

**Zhuofan Liao** received the Ph.D. degree in Computer Science from Central South University, China, in 2012. She is an Assistant Professor in the School of Computer and Communication Engineering at Changsha University of Science and Technology, China. She is a member of the IEEE and China Computer Federation.

**Yeon-Woong Jung** is a graduate student in Sangmyung University in Seoul. He is also running his own company and he mainly serves the system solutions. His research interests mainly include BIM, facility management system, and system integration.

**Jeong-Uk Kim** received the M.S and Ph.D. degree from the Korea Advanced Institute of Science and Technology. He is currently a Professor of Electrical Engineering and a Dean of the graduate school. His research interests include network algorithm, automation system, energy performance analysis.