# Energy Minimization and Offloading Number Maximization in Wireless Mobile Edge Computing

Peifeng Li*, Yuansheng Luo†, Kezhi Wang‡ and Kun Yang§*
* School of Information and Communications Engineering, University of Electronic Science and Technology of China, Chengdu, China
† School of Computer and Communications Engineering, Changsha University of Science and Technology, Changsha, China
‡ Department of Computer and Information Sciences, Northumbria University, Newcastle upon Tyne, UK
§ School of Computer Sciences and Electrical Engineering, University of Essex, Colchester, UK

***Abstract*—With the fast development of mobile edge computing (MEC), user equipments (UEs) can enjoy much higher experience than before by offloading the tasks to its close edge cloud. In this paper, we assume there are several edge clouds, each of which has limited resource. We aim to maximize the number of offloaded tasks and minimize the energy consumption of all the UEs and edge clouds, by selecting the best edge cloud for each UE to offload. We formulate the problem as a mixed-integer non-convex optimization, which is difficult to solve in general. By transforming this problem into a minimum-cost maximum-flow (MCMF) problem, we can solve it efficiently. The simulation shows that our proposed algorithm has better performance and lower complexity than the conventional solutions.**

***Index Terms*—Energy Minimization; Offloading Number Maximization; Mobile Edge Computing; Minimum-Cost-Maximum-Flow**

## I. Introduction

Nowadays, mobile edge computing (MEC) has attracted more and more attention from not only academia, but also industry. With the help of edge cloud, user equipment (UE) has the potential to enjoy much higher experience than before, as the UE can offload its task to the edge cloud, with the benefit of saving its own battery life and increasing the computing capacity. Unlike the traditional mobile cloud which is normally far from the user, edge cloud can be just around the UE. For instance, our laptop, road unit, street lamp can all be the edge cloud, and in [1], PC is used as an edge cloud. Compared to normal cloud, edge cloud can fast respond to the UE's request, which is very important for the latency-sensitive tasks, such as virtual reality service. However, compared to the normal cloud, edge cloud may be resource-limited and therefore, it may not be able to accept and conduct all the tasks offloaded from all the UEs. In this case, access control policy may be enforced by edge cloud to the offloaded tasks. Also, from the side of the UE, it may favor selecting the closer edge cloud to offload the task. Because with the increase of the distance between UE and edge cloud, it may take UE much more time and energy in offloading the data to the edge cloud. Therefore, it is crucial for UEs to select the best place to offload the tasks, by considering its own capacity, and the resource availability of the edge clouds.

Some previous works have attempted to reduce the energy consumption of offloading computations in MEC [2], where Luo *et al.* studied a multi-user computing problem in wireless interference environment, and developed an energy-efficient Algorithm. Zhang *et al.* [3] proposed a Near-Far Computing Enhanced C-RAN (NFC-RAN) architecture, which can better meet the QoS and increase the tasks offloading successful rate. However, in above papers, they did not consider the offloading place section.

In this paper, we assume there are several edge clouds, each of which has limited resource. Also, we assume there are a number of UEs, each of which has a computation-intensive task to be conducted. We aim to study the best place for each UE to offload, by considering the minimization of energy consumption for all the UEs and edge clouds and at the same time considering the maximization of the number of accepted tasks from UEs, with the constraints of limited computing resource in each edge cloud. We formulate this problem as a mixed-integer non-convex problem, which is difficult to solve in general. By transforming this optimization to a minimum-cost-maximum-flow problem, we are able to solve it properly. The simulation shows that our proposed algorithm has better performance than the conventional solutions.

The remainder of this paper is organized as follows. Section II presents the system model. The solution is presented in Section III. In Section IV, we present the simulation results, followed by the conclusion in section V.

## II. System Model

### A. System Architecture

In this paper, we assume there are serval edge clouds, randomly distributed in an area, where there are a number of UEs, trying to offload their tasks to the edge clouds, as shown in Fig. 1. We assume all the UEs intend to offload their tasks and no tasks will be executed locally. This is realistic for the tasks with high computations required, which may not be possible executed locally. We assume the edge clouds have limited resource and are not able to accept all the offloaded tasks and therefore access control will be conducted in each edge cloud. In Fig. 1, the dashed line means a possible/potential communication channel which the UE can use to offload the data to the corresponding edge cloud, while the solid line denotes the actual communication channel that UE selects to offload the data.

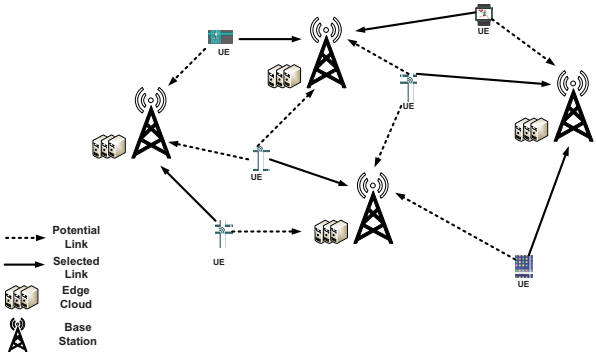


Fig. 1: System Architecture.

Consider that there are $\mathcal{I} = \{1, 2, ..., I\}$ UEs and $\mathcal{J} = \{1, 2, ..., J\}$ base stations, each of which has an edge cloud. Also, assume each UE $i$ has a task $U_i$ to be offloaded.

$$U_i = (D_i, B_i, T_i^{max}), \forall i \in \mathcal{I} \tag{1}$$

where $D_i$ denotes the total number of the CPU cycles required to accomplish the task measured in Gigacycle, $B_i$ describes the data size of task $i$ to be offloaded. $T_i^{max}$ is the QoS requirement of this task.

Let $E_{ij}^{tr}$ denotes the transmission energy consumption of task $U_i$, then, one can have

$$E_{ij}^{tr} = P_i^{tr} T_{ij}^{tr}, \forall i \in \mathcal{I}, \forall j \in \mathcal{J} \tag{2}$$

where $P_i^{tr}$ is the transmitting power, which will be fixed in transmission such as in [5] [6] [7], $T_{ij}^{tr}$ is the time for uploading the data to the $j$-th edge cloud, given by

$$T_{ij}^{tr} = \frac{B_i}{R_{ij}}, \forall i \in \mathcal{I}, \forall j \in \mathcal{J} \tag{3}$$

in which $R_{ij}$ is the uplink data rate from the $i$-th UE to the $j$-th edge cloud. Then, one can have $R_{ij}$ as [4]

$$R_{ij} = W \log_2(1 + \frac{P_i^{tr} H_{ij}}{w_n + \sum_{u=1}^{I} P_u^{tr} H_{uj}}), \forall i \in \mathcal{I}, \forall j \in \mathcal{J} \tag{4}$$

where $W$ is the channel bandwidth, $w_n$ is the noise power. $H_{ij}$ is the channel gain between the $i$-th UE to the $j$-th edge cloud, given by [5]

$$H_{ij} = d_{ij}^{-\alpha}, \forall i \in \mathcal{I}, \forall j \in \mathcal{J} \tag{5}$$

in which $d_{ij}$ is the distance between the $i$-th UE to the $j$-th edge cloud. We assume the path loss factor as $\alpha = -4$ [5].

Moreover, assume the computing energy in edge cloud is given as $E_{ij}^C$. Then, one can have

$$E_{ij}^C = P_j^C T_{ij}^C, \forall i \in \mathcal{I}, \forall j \in \mathcal{J} \tag{6}$$

where $T_{ij}^C$ denotes the time of executing the task $i$ in the $j$-th edge cloud. One can have $T_{ij}^C$ as

$$T_{ij}^C = \frac{D_i}{f_j^C}, \forall i \in \mathcal{I}, \forall j \in \mathcal{J} \tag{7}$$

where $f_j^C$ is the computing capacity that edge cloud allocates to UE measured in Gigahertz(Ghz). Also, in (6), $P_{ij}^C$ denotes the computing power consumption in edge cloud, which can be modeled as [8]

$$P_j^C = \kappa^C (f_j^C)^{\nu^C}, \forall j \in \mathcal{J} \tag{8}$$

where $\kappa^C \geq 0$ is the effective switched capacitance and $\nu^C \geq 1$ is the positive constant [9] to meet the test. We set $\kappa^C = 1$ and $\nu^C = 3$ in this paper.

Furthermore, we assume $x$ is a collection of all the $x_{ij}$, $\forall i \in \mathcal{I}, \forall j \in \mathcal{J}$, where $x_{ij} \in \{0, 1\}$ denotes whether the $i$-th UE selects the $j$-th edge cloud to offload, i.e., $x_{ij} = 1$ means the $i$-th UE selecting the $j$-th edge cloud, while $x_{ij} = 0$, otherwise. Then the total energy consumption can be given as

$$E(x) = \sum_{i=1}^{I} \sum_{j=1}^{J} (E_{ij}^{tr} + E_{ij}^C) x_{ij} \tag{9}$$

Also, assume the available computation resource in $j$-th edge cloud as $F_j$. Then, one can have

$$\sum_{i=1}^{I} x_{ij} f_j^C \leq F_j, \forall j \in \mathcal{J} \tag{10}$$

Moreover, assume that the number of the whole accepted tasks by all the edge cloud is $N$, then one can have

$$N(x) = \sum_{i=1}^{I} \sum_{j=1}^{J} x_{ij} \tag{11}$$

Then, assume the task from UEs only need to be conducted in one edge cloud or got rejected. Thus, one can have

$$\sum_{j=1}^{J} x_{ij} \leq 1, \forall i \in \mathcal{I} \tag{12}$$

To guarantee the QoS requirement for each UE, one can also have

$$\sum_{j=1}^{J} x_{ij} (T_{ij}^C + T_{ij}^{tr}) \leq T_i^{max}, \forall i \in \mathcal{I} \tag{13}$$

where $T_i^{max}$ is the maximum time in which the task has to be completed. Next, we will give the problem formulation.

### *B. Problem Formulation*

We aim to maximize the number of offloaded tasks and minimize the energy consumption of all the UEs and edge clouds.

To this end, the optimization problem can be formulated as follows

$$\begin{aligned}
\mathcal{P}: \quad & \underset{x}{\text{minimize}} \quad \{E(x), -N(x)\} \\
& \text{subject to} : (1) \quad \sum_{j=1}^{J} x_{ij} \leq 1, \forall i \in \mathcal{I} \\
& (2) \quad \sum_{i=1}^{I} x_{ij} f_j^C \leq F_j, \forall j \in \mathcal{J} \\
& (3) \quad \sum_{j=1}^{J} x_{ij}(T_{ij}^C + T_{ij}^{tr}) \leq T_i^{max}, \forall i \in \mathcal{I} \\
& (4) \quad x_{ij} \in \{0,1\}, \forall i \in \mathcal{I}, \forall j \in \mathcal{J}
\end{aligned} \tag{14}$$

One can see that we have two objectives in $\mathcal{P}$, where **minimize** $\{E(x)\}$ is the minimization of the energy consumption for all the UEs and edge clouds, while **minimize** $\{-N(x)\}$ can be seen as **maximize** $\{N(x)\}$, i.e., the maximization of the number of offloaded tasks. One can also see that $\mathcal{P}$ is a multiple-objective-optimization problem (MOOP) with two objective functions. It is also a mixed-integer programming problem. This kind of problem is usually an NP-hard problem. Next, we will analyze the characteristics of this problem and transform it into a relaxed problem which can be easily solved.

## III. Minimum-Cost-Maximum-Flow (MCMF) based Solution

### A. Reformulating Constraint

At first, the constraint (2) can be transformed as follows:

$$\sum_{i=1}^{I} x_{ij} f_j^C \leq F_j \Rightarrow \sum_{i=1}^{I} x_{ij} \leq \frac{F_j}{f_j^C} \tag{15}$$

Since $x_{ij}$ is an integer, let $Q_j = \lfloor \frac{F_j}{f_j^C} \rfloor$, (15) can be re-written as:

$$\sum_{i=1}^{I} x_{ij} \leq Q_j \tag{16}$$

where $\lfloor \bullet \rfloor$ means the operation to get the nearest integer less than or equal to the value.

### B. Pruning Strategy

We can get the lower bound of uplink data rate $R_{ij}^{min}$ as:

$$T_{ij}^C + T_{ij}^{tr} \leq T_i^{max} \Rightarrow R_{ij} \geq \frac{B_i}{T_i^{max} - \frac{D_i}{f_j^C}} = R_{ij}^{min} \tag{17}$$

where $R_{ij}^{min}$ is the lower bound of $R_{ij}$. Each candidate link between the $i$-th UE and the $j$-th edge cloud in Fig. 1 represents that the uplink data rate $R_{ij}$ satisfies (17). Since $R_{ij}$ can be calculated by (4), we can prune to remove the infeasible candidate offloading choice $x_{ij}$, which violate the (17) when $x_{ij} = 1$, from the candidate set. Thus, the remaining $x_{ij}$ in the candidate set must conform to (17) and the constraint (3) can be removed from the formulation (14).

Now we can transform the original problem $\mathcal{P}$ to the following problem $\mathcal{P}'$:

$$\begin{aligned}
\mathcal{P}': \quad & \underset{x'}{\text{minimize}} \quad \{E(x'), -N(x')\} \\
& \text{subject to} : (1) \quad \sum_{j=1}^{J'} x'_{ij} \leq 1, \forall i \in \mathcal{I}' \\
& (2) \quad \sum_{i=1}^{I'} x'_{ij} \leq Q_j, \forall j \in \mathcal{J}' \\
& (3) \quad x'_{ij} \in \{0,1\}, \forall i \in \mathcal{I}', \forall j \in \mathcal{J}'
\end{aligned} \tag{18}$$

### C. Problem Transformation

As illustrated in Fig. 2, by adding a dummy node $M$, a source node $S$ and a terminal node $T$ in the topology graph of problem $P$, the original problem is transformed to a new problem. In Fig. 2, there are two kinds of edges. One is the solid line connecting UE nodes to edge cloud nodes, which means there are real communication channels between UEs and edge clouds. The other one is the dashed line from the source node $S$ to UE nodes, or from UE nodes to the dummy node $M$ and then to the terminal node $T$. These dashed lines are virtual connections and just used to construct the new problem. We can transform the problem $P'$ into the Minimum-Cost-Maximum-Flow (MCMF) problem. We assume that $(s, e)$ denotes the edge from node $s$ to node $e$ in Fig. 2, $\mathcal{A}$ is the collection of edges.

In Fig. 2, if node $s$ tries to deliver its task to node $e$, there will be a "flow" going through the edge $(s, e)$, and the flow through the edge will cause cost $a_{se}x''_{se}$. Where $a_{se}$ is the cost of the edge $(s, e)$, and $x''_{se}$ denotes the number of tasks flow through the edge$(s, e)$. The energy consumption in (18) will be transformed to the cost of each edge between UE nodes and edge cloud nodes in Fig. 2.

The constraint (1) in $\mathcal{P}'$ can be transformed to as follows

$$\begin{aligned}
& (19a) \;\; 0 \leq x''_{se} \leq 1, s = S, \forall e \in \mathcal{I}' \\
& (19b) \;\; \sum_{\{e \in \mathcal{I}', s = S\}} x''_{se} - \sum_{\{e \in \mathcal{I}', s = S\}} x''_{es} = |I'| \\
& (19c) 0 \leq x''_{se} \leq 1, \forall s \in \mathcal{I}', \forall e \in \mathcal{J}' \bigcup M
\end{aligned} \tag{19}$$

As shown in Fig. 2, there is an edge between each UE node and S node, thus (19) guarantees that each UE node gets one and only one task delivered by S node.

The constraints (2) in $\mathcal{P}'$ can be transformed to as follows

$$\begin{aligned}
& (20a) \;\; 0 \leq x''_{se} \leq Q_s, \forall s \in \mathcal{J}', e = T \\
& (20b) \;\; \sum_{\{e = T, s \in \mathcal{J}'\}} x''_{se} - \sum_{\{e \in \mathcal{I}', s \in \mathcal{J}'\}} x''_{es} = 0
\end{aligned} \tag{20}$$

Moreover, the number of in and out flow for all nodes except S and T node must be equal. All flow start at S node will finally arrive at T node, and if one UE node delivers its task flow to one of the edge cloud nodes and finally reach T node, it means that the UE offloads successfully. If one node fails to offload, then the flow will go through it to M node and

finally reach T node. Thus, we can solve original problem $\mathcal{P}$ by solving $\mathcal{P}''$:

$$\begin{aligned}\mathcal{P}'': \quad & \underset{x''}{\text{minimize}} \quad \sum_{(s,e)\in\mathcal{A}} a_{se}x''_{se} \\ & \text{subject to} : (1)\ \ 0 \le x''_{se} \le w_{se}, \forall (s,e) \in \mathcal{A} \\ & (2) \sum_{\{e|(s,e)\in\mathcal{A}\}} x''_{se} - \sum_{\{s|(e,s)\in\mathcal{A}\}} x''_{es} \\ & = \begin{cases} |I'|, s = S \\ 0, \forall s \in \mathcal{I}', s \neq S, T \\ -|I'|, s = T \end{cases}\end{aligned} \tag{21}$$

where $w_{se}$ is the weight of the edge $(s, e)$ with the value given by (22)

$$w_{se} = \begin{cases} Q_s, \forall s \in \mathcal{J}', e = T \\ 1, \forall s \in \mathcal{I}', \forall e \in \mathcal{J}' \bigcup M \\ 1, s = S, \forall e \in \mathcal{I}' \\ |I'|, s = M, e = T \end{cases} \tag{22}$$

Also, in $\mathcal{P}''$, $a_{se}$ is the cost of edge $(s, e)$ with the value given by (23):

$$a_{se} = \begin{cases} E^{tr}_{se} + E^{C}_{se}, \forall s \in \mathcal{I}', \forall e \in \mathcal{J}' \\ G, \forall s \in \mathcal{I}', e = M \\ 0, \forall s \in \mathcal{J}' \bigcup M, e = T \\ 0, s = S, \forall e \in \mathcal{I}' \end{cases} \tag{23}$$

where $G$ can be a very large value. For example, one can assign $G$ as a value which is larger than the sum of energy consumption of all the UEs and edge clouds.

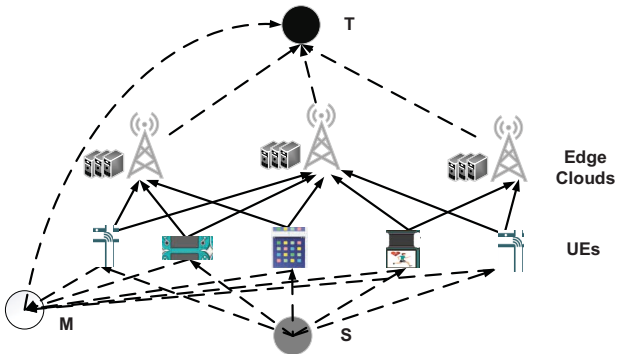


Fig. 2: Topology of Problem $\mathcal{P}''$

### D. Optimal Solution

*1) Integer Solution:* The $\mathcal{P}''$ is actually a LP relaxation to the original problem $\mathcal{P}$, thus it can be solved by the simplex algorithm. But general the solution to LP is not guaranteed to be integral. To guarantee the result returned by the simplex algorithm is integral, all entries of $P''$ should be integers, i.e., the $a_{se}$, $\mathcal{I}'$, $w_{se}$ should be integers. $\mathcal{I}'$, $w_{se}$ are integers as aforementioned. Part of the $a_{se}$ is the energy consumption, which may be not integral. To address this issue, we can adjust the magnitude of the energy unit to a large order to make it as integer values. For example, if the energy consumption is 1.2 Joule, we can adjust it to 1200 Millijoule. If $a_{se}$ is too small, the fractional part can be omitted. By making the entries integral, one can get the integer solutions of the MCMF by the simplex algorithm.

*2) Optimality Analysis:* Since the original problem is an MOOP, we use the Pareto Optimality to discuss the optimality.

**Definition 1**. *Pareto Optimality: Assume that $f_1(x) = E(x)$ and $f_2(x) = -N(x)$, a decision variable $x^*$ is Pareto optimal, if there does not exist another feasible decision $x$ subject to $f_m(x) \le f_m(x^*)$ for all $m \in \{1, 2\}$ and at least one of the inequalities is strict, i.e., $\exists k \in \{1, 2\}, f_k(x) < f_k(x^*)$.*

Then, we can have following Theorem according to the Definition 1.

**Theorem 1**. *If $x^*$ is the optimal solution of problem $\mathcal{P}''$, then it is the Pareto optimal solution of both problem $\mathcal{P}$ and $\mathcal{P}''$.*

**Proof:** The proof of theorem 1 is omitted here due to the page limitation.

## IV. Simulation Results

In this section, we present the simulation results to show the performance of our proposed method. The configuration of the simulation is listed in Table I. The total available CPU frequency $F^{total}$ is assigned to all the edge cloud nodes randomly and $\sum_{j=1}^{J} F_j = F^{total}$. In the simulation, we allocate the UE nodes and the edge cloud nodes in the simulation area randomly.

TABLE I: The Simulation Parameters

| Parameter Name | Parameter Value (unit) |
|---|---|
| Simulation Area | $3000(m) \times 3000(m)$ |
| Bandwidth $W$ | $1.00 \times 10^6(Hz)$ |
| CPU Frequency of VMs $f^C_j$ | $\{0.2, 0.4, 0.8, 1\}(Ghz)$ |
| Total CPU Frequency $F^{total}$ | $100(Ghz)$ |
| Task CPU Cycles $D_i$ | $0.1(Gigacycle) - 1(Gigacycle)$ |
| Task Data Size $B_i$ | $\{0.01, 0.04, 0.06, 0.1, 0.4\} \times 10^5(Bit)$ |
| The QoS Requirement $T^{max}_i$ | $\{1.4, 1.8, 2.2, 2.6\}(s)$ |
| Transmission Power $P^{tr}_i$ | $0.4(W), i \in \mathcal{I}$ |
| Background Noise $w_n$ | 0(dBm) |

We compare our solutions with the following two benchmark algorithms, i.e., Greedy Algorithm and the Distance First Algorithm.

We input the distance data, time consumption and QoS into the three algorithms. The Greedy Algorithm and Distance First Algorithm are described in Algorithm 1&2:

The average energy consumption (AEC) is used in comparison, which is:

$$AEC = \frac{\text{Total Energy Consumption}}{\text{Number Of Offloaded Tasks}} \tag{24}$$

We fixed the number of edge clouds and changed the number of UEs at first, and fixed the number of UEs while changed the number of edge clouds after that.

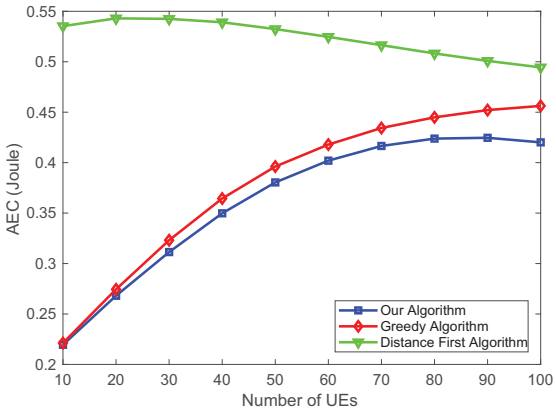


Fig. 3: Average energy consumption vs. the number of UEs where the number of edge clouds=18

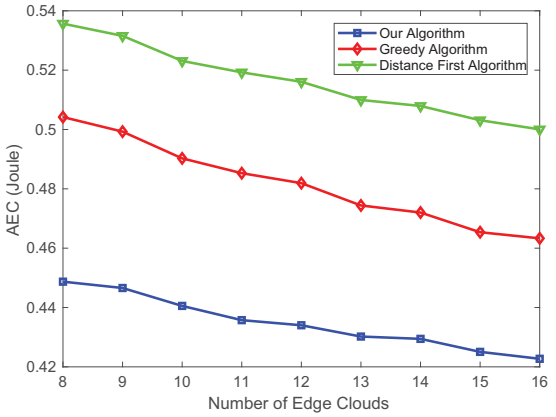


Fig. 4: Average energy consumption vs. the number of edge cloud where the number of UEs=100

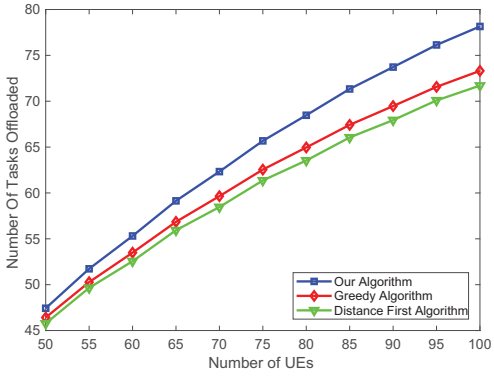


Fig. 5: Number of tasks offloaded successfully vs. the number of UE where the number of edge clouds=18

**Algorithm 1** Greedy Algorithm

**Choose** $i$-th UE from the UEs that have not offloaded their tasks randomly to offload its task ;
**There** are $J$ edge clouds the $i$-th UE can choose to offload, sort the energy consumption $i$-th UE to each of the edge cloud from low to high, and reorder the edge clouds according to the energy comsumption order. Let $j = 1$;
**If** the CPU frequency of $j$-th edge cloud is enough and the time consumption to $j$-th edge cloud meet the QoS $T_i^{max}$
  **choose** the $j$-th edge cloud to offload, go to step 13;
  **Else**
    $j = j + 1$;
    **If** $j \leq J$
      **back** to step 3;
    **Else**
      **Offload** fail, and go to step 13;
    **Endif**
  **Endif**
**If** all UEs have been chosen
  **Output** the average energy consumption, the number of tasks offloaded successfully. **End** algorithm;
  **Else**
  **Back** to step 1;
**EndIf**

**Algorithm 2** Distance First Algorithm

**Choose** $i$-th UE from the UEs that have not offloaded their tasks randomly to offload its task ;
**There** are $J$ edge clouds the $i$-th UE can choose to offload, sort the distance between $i$-th UE and each of the edge cloud from short to long, and reorder the edge clouds according to the distance order. Let $j = 1$;
**If** the CPU frequency of $j$-th edge cloud is enough and the time consumption to $j$-th edge cloud meet the QoS $T_i^{max}$:
  **choose** the $j$-th edge cloud to offload, go to step 13;
  **Else**
    $j = j + 1$;
    **If** $j \leq J$
      **back** to step 3;
    **Else**
      **offload** fail, and go to step 13;
    **Endif**
  **Endif**
**If** all UEs have been chosen
  **Output** the average energy consumption, the number of tasks offloaded successfully. **End** algorithm;
  **Else**
  **Back** to step 1;
**EndIf**

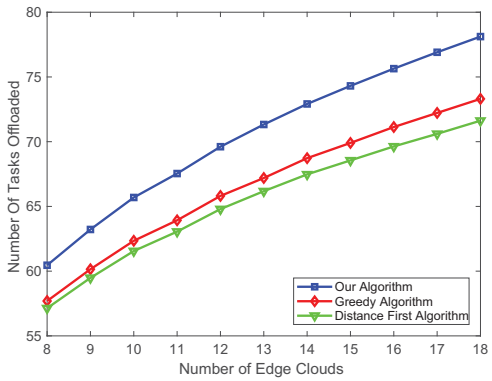


Fig. 6: Number of tasks offloaded successfully vs. the number of edge clouds where the number of UEs=100

As depicted in Fig. 3, the proposed MCMF based method outperforms the Greedy Algorithm and Distance Algorithm in average energy consumption. The number of the edge clouds is set to 18. The proposed method can achieve up to $8.78\%$ and $17.82\%$ energy consumption reduction over the solutions achieved by the Greedy Algorithm and Distance First Algorithm respectively when the number of UEs equal to 100.

As shown in Fig.4, with the increase of the number of edge clouds, the AEC becomes smaller and smaller because there are more computation and communication resources provided with the increasing of edge clouds.

In Fig.5, the impact of the number of edge clouds on the number of the offloaded tasks is shown. The proposed algorithm achieves up to $6.56\%$ and $8.62\%$ more number of offloaded tasks than Greedy Algorithm and Distance First Algorithm.

As depicted in Fig.6, with the increasing of number of edge clouds, more and more tasks can be offloaded. More edge clouds mean more choices for the UEs, and more likely exist the feasible solutions.

## V. Conclusion

In this paper, we propose the energy minimization and offloading number maximization problem in wireless mobile edge computing. This problem can be seen as an MOOP optimization with two objectives. By transforming it to a MCMF problem, we can solve it effectively. We also show that the solution we obtain is a Pareto optimal solution. Simulation results show that the proposed method outperform the traditional solutions.

## Acknowledgment

This work is supported by the Natural Science Foundation of China (Grant No.61620106011, 61572389 and 61303043), UK EPSRC Project NIRVANA (Grant No.EP/L026031/1).

## References

[1] K. Yang, S. Ou and H. Chen, "On effective offloading services for resource-constrained mobile devices running heavier mobile Internet applications," in IEEE Communications Magazine, vol. 46, no. 1, pp. 56-63, January 2008.

[2] Luo C, Salinas S, Li M, et al. Energy-Efficient Autonomic Offloading in Mobile Edge Computing[C]. Dependable, Autonomic and Secure Computing, 15th Intl Conf on Pervasive Intelligence & Computing, 3rd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress (DASC/PiCom/DataCom/CyberSciTech), 2017 IEEE 15th Intl. IEEE, 2017: 581-588.

[3] Zhang L, Wang K, Xuan D, et al. Optimal Task Allocation in Near-Far Computing Enhanced C-RAN for Wireless Big Data Processing[J]. IEEE Wireless Communications, 2018, 25(1): 50-55.

[4] T. S. Rappaport. *Wireless communications: principles and practice*[M]. New Jersey: prentice hall PTR, 1996.

[5] Xiao M, Shroff N B, Chong E K P.A utility-based power-control scheme in wireless cellular systems[J]. IEEE/ACM Transactions on Networking (TON), 2003, 11(2): 210-221.

[6] Saraydar C U, Mandayam N B, Goodman D J. Efficient power control via pricing in wireless data networks[J]. IEEE transactions on Communications, 2002, 50(2): 291-303.

[7] Chen X. Decentralized computation offloading game for mobile cloud computing[J]. IEEE Transactions on Parallel and Distributed Systems, 2015, 26(4): 974-983.

[8] Wang K, Yang K. Power-minimization computing resource allocation in mobile cloud-radio access network[C]. Computer and Information Technology (CIT), 2016 IEEE International Conference on. IEEE, 2016: 667-672.

[9] Tang J, Tay W P, Wen Y. Dynamic request redirection and elastic service scaling in cloud-centric media networks[J]. IEEE Transactions on Multimedia, 2014, 16(5): 1434-1445.

[10] Bertsekas D P. *Network optimization : continuous and discrete models*[M]. Athena Scientific, 1998.

[11] Ehrgott M, Gandibleux X. Multiple Criteria Optimization: State of the Art Annotated Bibliographic Surveys[J]. Kluwer Academic Publishers Boston Ma, 2010, 52:xxii,496.