SMETO: Stable Matching for Energy-Minimized Task Offloading in Cloud-Fog Networks

Yijun Zu*, Fei Shen†‡, Feng Yan*, Lianfeng Shen*, Fei Qin§, Rong Yang¶
*National Mobile Communications Research Laboratory, Southeast University, Nanjing, 210096, China
†Key Lab of Wireless Sensor network and Communication, Shanghai Institute of Microsystem and Information Technology, Chinese Academy of Sciences, Shanghai, 200050, China
‡Shanghai Research Center for Wireless Communications, Shanghai, 201210, China
§The School of Electronic and Electrical Communication Engineering, University of Chinese Academy of Sciences, Beijing, 100049, China
¶College of Machatronics and Control Engineering, Shenzhen University, Shenzhen, 518060, China
Email: zyj@seu.edu.cn, fei.shen@mail.sim.ac.cn, feng.yan@seu.edu.cn

***Abstract*—In order to minimize the total energy consumption of a cloud-fog network, one of the most essential challenges is the assignment of subtasks from the task node (TN) to suitable fog nodes (FNs). In this paper, we apply a many-to-one matching to deal with this problem. Specifically, we first introduce two concepts, Service Efficiency (SE) and Energy Efficiency (EE), as the indexes of the preference list (PL) of TNs and FNs, respectively. Then a stable matching algorithm for energy-minimized task offloading (SMETO) is proposed, which is comprised of two key components: (i) Deferred Acceptance Algorithm Based On Energy Efficiency (EEDA) and (ii) Energy-Minimized Task Allocation (EMTA). Algorithm (i) is an iterative procedure that matches TNs and helpers based on PLs. Algorithm (ii) minimizes the energy consumption in the network by allocating the subtasks to helpers according to the result of matching. Finally, numerical simulations indicate the stability and energy-minimization of our proposed SMETO.**

***Index Terms*—fog computing, task offloading, matching theory, QoS requirement**

I. Introduction

With the rapid development of internet of things (IoT), there exist billions of smart devices connected to the network, such as smart phones and wearable devices. In addition, an increasing number of new mobile applications, such as online games and augmented reality, generates a large amount of data. To relieve the pressure of cloud computing on link delay and congestion, the promising fog computing is proposed to help offload tasks from nearby task nodes (TNs). However, processing massive data consumes a lot of energy in the cloud-fog network [1]. Considering the finite battery lifetime of some fog nodes (FNs), it is essential to focus on reducing the overall energy consumption in the network.

This work is supported in part by the National Natural Science Foundation of China (No. 61871370, No. 61601122 and No. 61773266), the Natural Science Foundation of Shanghai, China (No. 18ZR1437500 and No. 19ZR1423100), the Hundred Talent Program of Chinese Academy of Sciences (No. Y86BRA1001), the Fundamental Research Funds for the Central Universities, the Key Research & Development Plan of Jiangsu Province (No. BE2018108), and the Postdoctoral Science Foundation of China (No. 2019M651476).

In past few years, numerous researches focusing on the issue of energy consumption have been carried out. A low complexity algorithm is designed to minimize energy consumption while scheduling tasks in homogeneous fog networks [2]. The energy minimization problem with hybrid computation offloading is investigated, wherein the terminal tasks can be offloaded to either the FNs or the cloud data server [3]. Moreover, an energy-minimized task offloading algorithm based on a fairness scheduling metric is proposed [4].

Although the energy consumption in the fog network has been considered in many aspects, few literature studies on minimizing the energy consumption by considering Energy Efficiency (EE) based on both communication and computation as a whole. To realize energy-minimization in the cloud-fog network, we have to take different combinations of helpers for TNs into consideration, which involves combinational optimization. Then, we apply a many-to-one matching for task offloading, which is guaranteed to converge and can obtain a stable solution.

In this paper, our contributions are outlined as follows. In Section II, the cloud-fog system model is introduced and the problem formulations are provided. In Section III, we first introduce Service Efficiency (SE) and redefine EE, as the indexes of preference list (PL) in matching theory. Then, we develop the stable many-to-one matching algorithm for energy-minimized task offloading (SMETO) which contains two algorithms: (i) Deferred Acceptance Algorithm Based On Energy Efficiency (EEDA) and (ii) Energy-Minimized Task Allocation (EMTA). The stability and complexity of the algorithm are analysed. At last, numerical simulations are conducted in Section IV to evaluate the performance of the proposed algorithm.

II. System model and problem formulation

A. System Overview

As depicted in **Fig. 1**, a general hybrid cloud-fog architecture comprised of multiple fog clusters and a single cloud data server is considered. Each cluster consists of several FNs.

978-1-7281-1220-6/19/$31.00 ©2019 IEEE

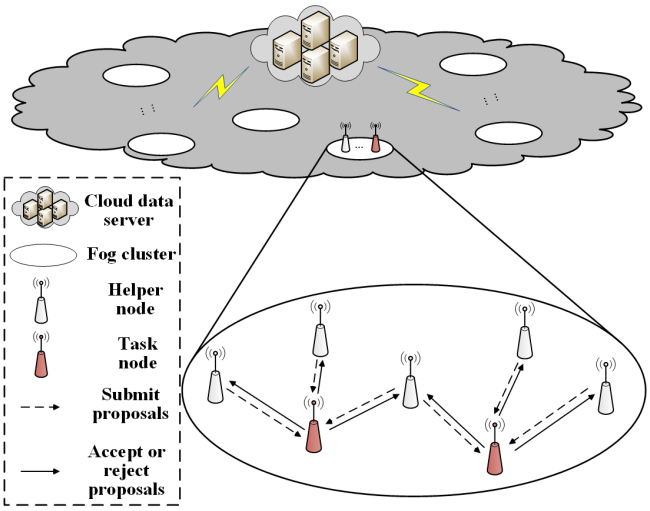

Fig. 1: Network Architecture

The FNs with computation tasks are referred to as TNs and the FNs with idle resources can serve as helper nodes (HNs). We denote the set of TNs as $\mathcal{T} = \{1, 2, \ldots, K\}$, and the set of HNs as $\mathcal{H} = \{1, 2, \ldots, M\}$. Correspondingly, we use k, m, c to refer to the k-th TN, the m-th HN and cloud data server, respectively.

At the beginning of offloading interval, TNs will broadcast their service requirement $\{L_k, \bar{D}_k\}$, where L_k is the size of data to be processed and $\bar{D}_k$ is the delay requirement to guarantee the quality of service (QoS) of TN k. We represent the feasible set of helpers for TN k as $\mathcal{C}_k$ with $\mathcal{C}_k = \mathcal{C}_k^{\mathcal{H}} \cup \mathcal{C}_k^c \cup \mathcal{C}_k^l$, where $\mathcal{C}_k^{\mathcal{H}}$, $\mathcal{C}_k^c$ and $\mathcal{C}_k^l$ are the set of HNs, the set of cloud data server and TN k itself, respectively. $\mathcal{C}_k^c = \{c\}$ and $\mathcal{C}_k^c = \emptyset$ represent whether the cloud data server serves TN k or not. $\mathcal{C}_k^l = \{l\}$ and $\mathcal{C}_k^l = \emptyset$ represent whether TN k processes the data by itself or not. Considering the limited resources in the network, each TN can schedule at most q_k helpers (HNs, cloud data server and TN itself), namely, $|\mathcal{C}_k| \leq q_k$ where $|\mathcal{C}_k|$ is the cardinality of $\mathcal{C}_k$. Besides, we assume the cloud data server can provide service for all the TNs at the same time and the computational task L_k can be divided randomly into $\boldsymbol{\alpha}_k = \{\alpha_{k,l}, \alpha_{k,c}, \alpha_{k,m} \in \mathbb{R}_+ | \alpha_{k,l} + \alpha_{k,c} + \sum_{m \in \mathcal{C}_k^{\mathcal{H}}} \alpha_{k,m} = 1\}$, where $\alpha_{k,l}$, $\alpha_{k,c}$ and $\alpha_{k,m}$ are the proportion of tasks computed locally, computed by cloud data server c and computed by the HN m, respectively.

In the following, the processing delay and energy of physical layer are analysed for communication and computation.

B. Communication Model

Since the feedback of the offloaded task is usually a control signal, we consider only the uplink transmission.

1) Communication delay: The transmission delay from TN k to helper i, $i \in \{m, c\}$, can be calculated as

$$D_{k,i}^T(\mathcal{C}_k, \boldsymbol{\alpha}_k) = \frac{\alpha_{k,i} L_k}{r_{k,i}}, i \in \mathcal{C}_k^{\mathcal{H}} \cup \mathcal{C}_k^c. \tag{1}$$

In equation (1), given an OFDMA system, the interference among FNs can be ignored and $r_{k,i}$ is the transmitting data rate from TN k to the helper i, which can be written as

$$r_{k,i} = B \log_2 \left(1 + \frac{g p_{k,i}^t}{B N_0}\right), i \in \mathcal{C}_k^{\mathcal{H}} \cup \mathcal{C}_k^c, \tag{2}$$

where $B, g, N_0, p_{k,i}^t$ are the channel bandwidth, the channel gain, the noise power spectral density and transmission power from TN k to helper i, respectively.

2) Communication energy consumption: The energy consumed when data traffic transmits from TN k to helper i, $i \in \{m, c\}$, can be obtained as follows:

$$E_{k,i}^T(\mathcal{C}_k, \boldsymbol{\alpha}_k) = \frac{\alpha_{k,i} L_k p_{k,i}^t}{r_{k,i}}, i \in \mathcal{C}_k^{\mathcal{H}} \cup \mathcal{C}_k^c. \tag{3}$$

C. Computation Model

We only consider computation delay and energy consumption of HNs while ignoring that of the cloud data server due to its unlimited resource and energy.

1) Computation delay: The computation delay of the local execution or by HN can be expressed as

$$D_j^C(\mathcal{C}_k, \boldsymbol{\alpha}_k) = \frac{\alpha_{k,j} L_k}{C_j}, j \in \mathcal{C}_k^{\mathcal{H}} \cup \mathcal{C}_k^l, \tag{4}$$

where C_j is the CPU computation capability of TN or by HN.

2) Computation energy consumption: The computation energy consumption of the local execution or by HN can be expressed as

$$E_j^C(\mathcal{C}_k, \boldsymbol{\alpha}_k) = \frac{\alpha_{k,j} L_k p_j^c}{C_j}, j \in \mathcal{C}_k^{\mathcal{H}} \cup \mathcal{C}_k^l, \tag{5}$$

where p_j^c is computation power of the TN locally or by HN.

D. Problem Formulation

We focus on the physical layer latency because the physical layer has a great effect on other layers directly and the latency of physical layer accounts for the majority of latency [5]. The total latency consists of the three components: the local execution delay $D_k(\mathcal{C}_k, \boldsymbol{\alpha}_k)$, the offloading delay to the cloud data server $D_{k,c}(\mathcal{C}_k, \boldsymbol{\alpha}_k)$ and the offloading delay to HNs $D_{k,m}(\mathcal{C}_k, \boldsymbol{\alpha}_k)$. The specific expressions are as follows:

$$D_k(\mathcal{C}_k, \boldsymbol{\alpha}_k) = D_l^C(\mathcal{C}_k, \boldsymbol{\alpha}_k), \tag{6}$$

$$D_{k,c}(\mathcal{C}_k, \boldsymbol{\alpha}_k) = D_{k,c}^T(\mathcal{C}_k, \boldsymbol{\alpha}_k), \tag{7}$$

$$D_{k,m}(\mathcal{C}_k, \boldsymbol{\alpha}_k) = D_m^C(\mathcal{C}_k, \boldsymbol{\alpha}_k) + D_{k,m}^T(\mathcal{C}_k, \boldsymbol{\alpha}_k). \tag{8}$$

The total latency can be determined by the maximum of the above three equations and can be written as

$$D(\mathcal{C}_k, \boldsymbol{\alpha}_k) = \max \left(D_k(\mathcal{C}_k, \boldsymbol{\alpha}_k), D_{k,c}(\mathcal{C}_k, \boldsymbol{\alpha}_k), \max_{m \in \mathcal{C}_k^{\mathcal{H}}} D_{k,m}(\mathcal{C}_k, \boldsymbol{\alpha}_k) \right). \tag{9}$$

On the other hand, the total energy consumption of processing the tasks generated at the TN k is expressed as

$$E_k(\mathcal{C}_k, \boldsymbol{\alpha}_k) = E_k^T(\mathcal{C}_k, \boldsymbol{\alpha}_k) + E_k^C(\mathcal{C}_k, \boldsymbol{\alpha}_k), \tag{10}$$

where $E_k^T(\mathcal{C}_k, \boldsymbol{\alpha}_k)$ indicates the energy consumed in transmission and is expressed as

$$E_k^T(\mathcal{C}_k, \boldsymbol{\alpha}_k) = \sum_{i \in \mathcal{C}_k^{\mathcal{H}} \cup \mathcal{C}_k^c} E_{k,i}^T(\mathcal{C}_k, \boldsymbol{\alpha}_k), \tag{11}$$

while $E_k^C(\mathcal{C}_k, \boldsymbol{\alpha}_k)$ is the energy consumed in computation and expressed as

$$E_k^C(\mathcal{C}_k, \boldsymbol{\alpha}_k) = \sum_{j \in \mathcal{C}_k^{\mathcal{H}} \cup \mathcal{C}_k^l} E_j^C(\mathcal{C}_k, \boldsymbol{\alpha}_k). \tag{12}$$

In the system model aforementioned, our objective is to minimize the total energy consumption in the fog network. As a result, the specific problem can be expressed as

$$\min_{\{\mathcal{C}_k\}, \{\boldsymbol{\alpha}_k\}} \quad \sum_{k \in \mathcal{T}} E_k(\mathcal{C}_k, \boldsymbol{\alpha}_k), \tag{13a}$$

$$s.t. \quad \mathcal{C}_k \subseteq \mathcal{H} \cup \{c, l\}, \forall k \in \mathcal{T}, \tag{13b}$$

$$|\mathcal{C}_k| \leq q_k, \forall k \in \mathcal{T}, \tag{13c}$$

$$\mathcal{C}_k \cap \mathcal{C}_{k'} \subseteq \{c\}, \forall k, k^{'} \in \mathcal{T} \, and \, k \neq k^{'}, \tag{13d}$$

$$\boldsymbol{\alpha}_k \in \mathbb{R}_+^{|\mathcal{C}_k|}, \forall k \in \mathcal{T}, \tag{13e}$$

$$\mathbf{1}^{\boldsymbol{T}} \boldsymbol{\alpha}_k = 1, \forall k \in \mathcal{T}, \tag{13f}$$

$$D(\mathcal{C}_k, \boldsymbol{\alpha}_k) \leq \bar{D}_k, \forall k \in \mathcal{T}, \tag{13g}$$

$$p_{k,i}^t \leq \bar{p_k}, \forall k \in \mathcal{T}, \forall i \in \mathcal{C}_k^{\mathcal{H}} \cup \mathcal{C}_k^c. \tag{13h}$$

The cloud data server and multiple HNs can be assigned to help a single TN while each HN processes the tasks offloaded from a single TN simultaneously, which is indicated in Constraint (13b) and (13d) respectively. And Constraint (13h) implies the transmission power has to be less than the maximum transmission power $\bar{p_k}$. The problem is indeed a combinatorial optimization problem, which is NP-hard [6]. In the following section, we adopt the stable SMETO to solve (13), which applies the many-to-one matching.

III. Solution Based On Matching Theory

A. Preliminary

Matching theory is an important mathematical instrument to solve combinatorial problem. The DA algorithm is an efficient algorithm that can find a stable solution in polynomial time for one-to-one or very fast for many-to-one matching [7]. To solve problem (13) and guarantee the stability of matching result, we apply DA algorithm in SMETO algorithm which consists of EEDA algorithm and EMTA algorithm.

According to the environment of network, HNs generate the PLs in descending order with SE. After receiving the proposal from HNs, the TN k calculates the EE of each helper and selects q_k helpers from its PL to minimize its energy consumption $E_k(\mathcal{C}_k, \boldsymbol{\alpha}_k)$. The definition of SE and EE are as follows:

- SE: the index of PLs in HNs, denoted as λ
- EE: the index of PLs in TNs, denoted as η

For HNs, we suppose that they would prefer to serve the TN whose channel quality is the best in the network. Hence, we can define the SE of HN m as

$$\lambda(m, k) = r_{k,m}, \tag{14}$$

where $r_{k,m}$ is expressed in detail in equation (2). It is vital to select a suitable $p_{k,m}^t$ to offload tasks. For a given TN, transmit EE (in bits per joule) is convex with the transmission power [8]. Therefore, the optimal $p_{k,m}^{t*}$ can be found to minimize the transmission energy. In addition, as the transmission power has to be less than the maximum transmission power $\bar{p_k}$, we have

$$p_{k,m}^t = \min\left(p_{k,m}^{t*}, \bar{p_k}\right), \tag{15}$$

which is applicable to the transmitting data rate $r_{k,c}$.

For TNs, their goal is to process their tasks with the least energy under the condition of service requirement. Thus we can define the EE of helper i as

$$\eta(k, i) = \frac{R_{k,i}}{P_{k,i}}, \tag{16}$$

where $R_{k,i}$ is the equivalent computation capability when tasks are offloaded from TN k to helper i and is expressed as

$$R_{k,i} = \begin{cases} C_l, & i \in \mathcal{C}_k^l, \\ r_{k,c}, & i \in \mathcal{C}_k^c, \\ 1 \Big/ \left(\frac{1}{C_m} + \frac{1}{r_{k,m}}\right), & i \in \mathcal{C}_k^{\mathcal{H}}, \end{cases} \tag{17}$$

while $P_{k,i}$ is the equivalent computation power when tasks are offloaded from TN k to helper i and is expressed as

$$P_{k,i} = \begin{cases} p_l^c, & i \in \mathcal{C}_k^l, \\ p_{k,c}^t, & i \in \mathcal{C}_k^c, \\ \left(\frac{p_m^c}{C_m} + \frac{p_{k,m}^t}{r_{k,m}}\right) \Big/ \left(\frac{1}{C_m} + \frac{1}{r_{k,m}}\right), & i \in \mathcal{C}_k^{\mathcal{H}}. \end{cases} \tag{18}$$

Thus, $\eta(k, i)$ of local execution, cloud data server and HNs can obviously be expressed as

$$\eta(k, i) = \begin{cases} \frac{C_l}{p_l^c}, & i \in \mathcal{C}_k^l, \\ \frac{r_{k,c}}{p_{k,c}^t}, & i \in \mathcal{C}_k^c, \\ \frac{C_m r_{k,m}}{p_m^c r_{k,m} + p_{k,m}^t C_m}, & i \in \mathcal{C}_k^{\mathcal{H}}. \end{cases} \tag{19}$$

The helper with higher $\eta(k, i)$ indicates it consumes less energy while processing 1-bit data. The TNs select helpers with high $\eta(k, i)$ according the PL to save energy.

B. SMETO algorithm

1) EEDA algorithm: The implementation of EEDA is depicted in **Algorithm 1**. After HNs generate PLs and propose their favorite TNs, TNs determine their PLs and sort the PLs

in descending order, then accept the first q_k helpers and reject the rest. HNs remove the TN which rejects them and select the next one in their PLs to propose. The procedure is iterative and will not terminate until the PLs is empty or there is no TNs accepting the HNs.

Algorithm 1: Deferred Acceptance Algorithm Based On Energy Efficiency (EEDA)

Require: $\mathcal{T}, \mathcal{H}, c$
Ensure: $\{\mathcal{C}_k^*\}, \{p_{k,i}^t\}$
Determine $p_{k,i}^t, \forall k \in \mathcal{T}, \forall i \in \mathcal{H} \cup \{c\}$.
Calculate the $\lambda(m,k)$ and generate the PLs $\forall k \in \mathcal{T}, \forall m \in \mathcal{H}$.
The temporary list $\mathcal{T}_k = \emptyset$, and the waiting list $\mathcal{W}_k = \emptyset, \forall k \in \mathcal{T}$.
Cloud data server makes proposals to all TNs.
while $\exists$ unmatched HN with non-empty PL **do**
  for unmatched HN m with non-empty PL **do**
    HN m proposes its favorite TN k.
    Remove TN k from the PL of HN m.
  end for
  for TN $k \in \mathcal{T}$ **do**
    TN k adds HNs and cloud data server into $\mathcal{W}_k$ after TN k receives the proposals from helpers.
    $\mathcal{T}_k = \mathcal{T}_k \cup \mathcal{W}_k$. And sort the $\mathcal{T}_k$ by η.
    According to $\mathcal{T}_k$, TN k calculates the minimum number of helpers required, denoted as q_k.
    if $q_k < |\mathcal{T}_k|$ **then**
      Accept the first q_k helpers and reject the rest.
    else
      Keep all the helpers.
    end if
    Update $\mathcal{T}_k$ and clear $\mathcal{W}_k$.
  end for
end while
for TN $k \in \mathcal{T}$ **do**
  $\mathcal{C}_k^* = \mathcal{T}_k$.
end for
Return $\{\mathcal{C}_k^*\}, \{p_{k,i}^t\}$

2) EMTA algorithm: Once obtaining the matching result, the TN k offloads tasks to the helpers in the PL orderly because higher ranking in the PL indicates higher EE. The implementation of EMTA is depicted in **Algorithm 2**. The proportion of TN k offloading to the helpers can be expressed as follows:

$$\alpha_{k,c} = \frac{r_{k,c}\bar{D_k}}{L_k}, \tag{20}$$

$$\alpha_{k,l} = \frac{C_l\bar{D_k}}{L_k}, \tag{21}$$

$$\alpha_{k,m} = \frac{C_m r_{k,m}\bar{D_k}}{(C_m + r_{k,m})L_k}, \forall m \in \mathcal{C}_k^{\mathcal{H}}. \tag{22}$$

The last helper in the PL may be allocated less tasks than expected. Thus, after completing the tasks, it can cancel the match with the TN and then turn to serve other TN.

Algorithm 2: Energy-Minimized Task Allocation (EMTA)

Require: $\{\mathcal{C}_k^*\}$,$\{p_{k,i}^t\}, \forall i \in \mathcal{C}_k^{\mathcal{H}} \cup \mathcal{C}_k^c$
Ensure: $\{\boldsymbol{\alpha}_k\}$
for task node $k \in \mathcal{T}$ **do**
  Calculate $\alpha_{k,c}$,$\alpha_{k,l}$,$\alpha_{k,m}$ according to (20), (21), (22).
end for
Return $\{\boldsymbol{\alpha}_k\}$

3) Stability and complexity analysis: In the two-side matching, it is proved that there always exists the one-side optimal stable matching with deferred acceptance algorithm [9]. The matching scheme produced by the algorithm with helpers proposing in our paper is helper-optimal stable matching. Since helpers are voluntary to participate in the matching, the decision they made will not be changed. And the choices of TNs are also definitive, which is best for the TNs. Thus, once matching successfully, both sides will be stable.

The SMETO algorithm is guaranteed to converge. The computational complexity is determined by EEDA. In the worst case where each HN goes through its PL, the **while** loop will iterate for $(M+K)K$ times. In each **while** loop, there is a sort operation, whose running time is $O((M+K)\log(M+K))$. Therefore, in the worst case, the running time of the proposed SMETO algorithm is $O(K(M+K)^2\log(M+K))$, which is polynomial time.

IV. Performance Evaluation

A. System parameters

For simulations, we consider all the FNs are uniformly scattered in a circle with radius r. All parameters are summarized in **TABLE I**. For the channel gain, we adopt a large scale fading channel model [10]:

$$\boldsymbol{PL} = 20\log(d^{km}) + 20\log(B^{kHz}) + 32.45(dB). \tag{23}$$

TABLE I: SIMULATION PARAMETERS

Parameter	Value
K (# of TNs)	$[1-10]$
M (# of HNs)	$[20-150]$
B (bandwidth)	$[0-250]$ kHz
L_k (data size of TN k)	$[200-1000]$ kbit
$\bar{D_k}$ (maximum delay of TN k)	$[0.1-1]$ s
p_i^c (computing power of HN i)	$[350-550]$ mw
C_i (computational capability of HN i)	$[100-250]$ kbit/s
$\bar{p_k}$ (maximum transmitting power)	200 mw
r (radius of simulated circle)	$[0.2-2]$ km
N_0 (noise power spectral density)	-174 dBm/Hz

B. Performance of SMETO

To demonstrate the effectiveness of our proposed algorithm, we simulate the consumed energy of both SMETO and

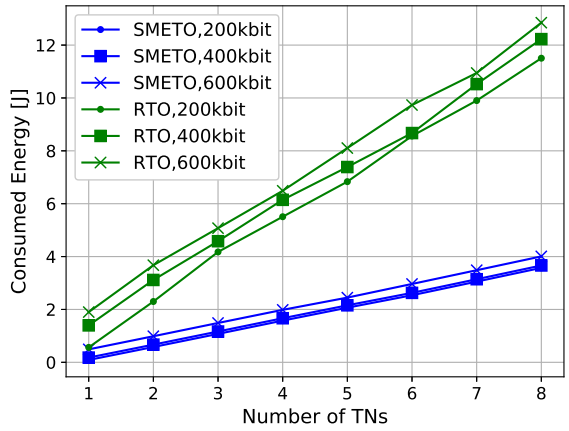

Fig. 2: SMETO versus RTO under different data sizes and different numbers of TNs ($M = 20, \bar{D}_k = 1s, B = 30kHz$)

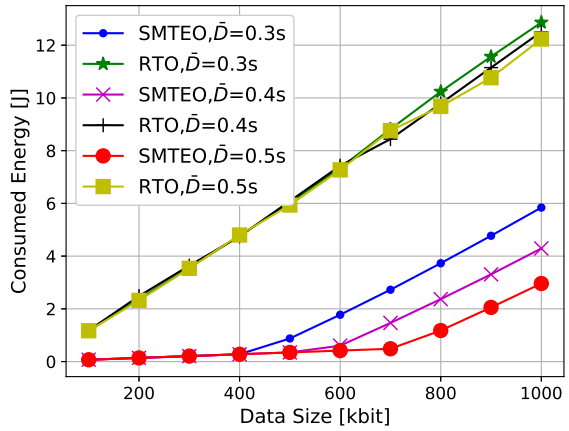

Fig. 3: Consumed energy versus data size under different $\bar{D}_k$ ($B = 250kHz, K = 10, M = 50$)

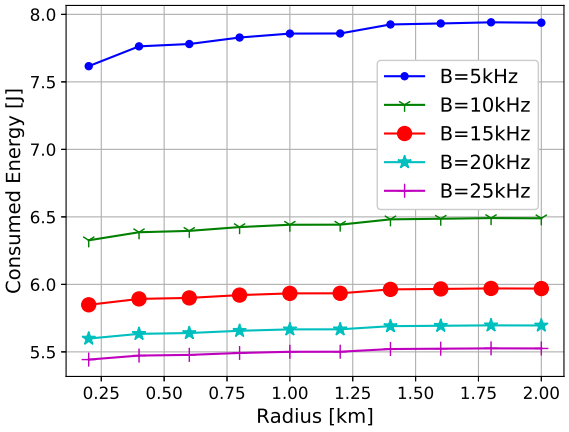

Fig. 4: Consumed energy versus radius under different bandwidths ($K = 5, M = 25$)

Random Task Offloading (RTO). In the RTO algorithm, the TN selects HNs randomly to offload its tasks. As shown in **Fig. 2**, it is obvious that as the number of TNs increases, the total energy consumed in the fog network rises linearly. Additionally, the more the number of TNs, the more energy SMETO will save than RTO. In conclusion, SMETO algorithm serves up to approximately 67% than RTO scheme.

In **Fig. 3**, the relationship between consumed energy and data size under different delay requirements is presented. As shown in the figure, all curves of SMETO have the turning points because the TN matches another helper to offload tasks. Additionally, we can observe that the energy consumption of RTO is apparently more than that of SMETO.

From **Fig. 4**, energy consumption tends to rise while the radius gets larger because more energy is consumed in transmission. In addition, we note that the wider the bandwidth is, the less energy the network consumes, which results from less consumption in transmission.

V. Conclusions

In this paper, we investigated an energy-minimized task offloading optimization problem in a typical cloud-fog network. To solve this problem, we adopted a many-to-one matching between HNs and TNs. The PLs of TNs and HNs were represented by EE and SE of TN and HN, respectively. Then we proposed the SMETO algorithm which consisted of two key components: (i) EEDA and (ii) EMTA. Finally, extensive simulations were conducted showing that the proposed SMETO algorithm outperformed RTO.

References

[1] X. Ge, J. Yang, H. Gharavi, and Y. Sun, "Energy efficiency challenges of 5G small cell networks," *IEEE Communications Magazine*, vol. 55, no. 5, pp. 184–191, May 2017.

[2] Y. Yang, K. Wang, G. Zhang, X. Chen, X. Luo, and M. Zhou, "Maximal energy efficient task scheduling for homogeneous fog networks," in *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 274–279, April 2018.

[3] X. Meng, W. Wang, and Z. Zhang, "Delay-constrained hybrid computation offloading with cloud and fog computing," *IEEE Access*, vol. 5, pp. 21 355–21 367, 2017.

[4] G. Zhang, F. Shen, Z. Liu, Y. Yang, K. Wang, and M. Zhou, "FEMTO: Fair and energy-minimized task offloading for fog-enabled iot networks," *IEEE Internet of Things Journal*, pp. 1–12, 2018.

[5] H. Ji, S. Park, J. Yeo, Y. Kim, J. Lee, and B. Shim, "Ultra-reliable and low-latency communications in 5G downlink: Physical layer aspects," *IEEE Wireless Communications*, vol. 25, no. 3, pp. 124–130, June 2018.

[6] J. Kleinberg and E. Tardos, "Algorithm design: Pearson new international edition," *Pearson Schweiz Ag*, 2005.

[7] Y. Gu, W. Saad, M. Bennis, M. Debbah, and Z. Han, "Matching theory for future wireless networks: fundamentals and applications," *IEEE Communications Magazine*, vol. 53, no. 5, pp. 52–59, May 2015.

[8] C. Xiong, G. Y. Li, S. Zhang, Y. Chen, and S. Xu, "Energy-efficient resource allocation in ofdma networks," *IEEE Transactions on Communications*, vol. 60, no. 12, pp. 3767–3778, December 2012.

[9] M. Mozaffari, W. Saad, M. Bennis, and M. Debbah, "Unmanned aerial vehicle with underlaid device-to-device communications: Performance and tradeoffs," *IEEE Transactions on Wireless Communications*, vol. 15, no. 6, pp. 3949–3963, June 2016.

[10] Y. Yu, J. Zhang, and K. B. Letaief, "Joint subcarrier and cpu time allocation for mobile edge computing," in *2016 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–6, Dec 2016.