Matching-Theory-Based Multi-User Cooperative Computing Framework

Ya Zhou, Guopeng Zhang, Kezhi Wang and Kun Yang

***Abstract*—In this paper, we propose a matching theory based multi-user cooperative computing (MUCC) scheme to minimize the overall energy consumption of a group of user equipments (UEs), where the UEs can be classified into the following roles: resource demander (RD), resource provider (RP), and standalone UE (SU). We first determine the role of each UE by leveraging the roommate matching method. Then, we propose the college admission based algorithm to divide the UEs into multiple cooperation groups, each consisting of one RP and multiple RDs. Next, we propose the rotation swap operation to further improve the performance without deteriorating the system stability. Finally, we present an effective task offloading algorithm to minimize the energy consumption of all the cooperation groups. The simulation results verify the effectiveness of the proposed scheme.**

***Index Terms*—multi-user cooperative computing, matching theory, computing task offloading.**

I. Introduction

WITH the rapid deployment of the computationally-intensive tasks, e.g., virtual reality, the requirement for UE in terms of battery life and computing resource are also increasing. Although traditional fixed infrastructure-based mobile edge computing (MEC) may help to provide computing resources to UEs, they may be inaccessible in some situations, like disasters or emergency cases where infrastructures are unavailable. MUCC [1] has recently been proposed to allow a UE to utilize available computing resources from neighboring UEs. In the framework of MUCC, the UEs can be classified as one of the following roles: 1) RD, which has a computationally-intensive task to be processed; 2) RP, which has available computing resource to provide for other RDs; and 3) SU, which may do the task itself.

A one-to-one MUCC scheme was proposed in [2] to minimize the long-term energy consumption of two UEs. In [3], a RD has the option to offload the task to a MEC server or a RP. In [4], a one-to-multiple MUCC scheme was proposed to allow an RD to partition its task into multiple parts and offload them to multiple RPs for parallel execution. In [5], an RD is allowed to select one RP from a group of potential RPs. The purpose of the above works is mainly to reduce the energy consumption or execution delay of individual users. However, they do not address the following issues: (1) how to determine the role of a UE, i.e., as an RD, an RP, or a SU; (2) how to determine the association between the RDs and the RPs; and (3) how each RP allocates its computing and communication resources to the served RDs.

This work was supported in part by the National Natural Science Foundation of China under Grant 61971421, Grant 61620106011, Grant U1705263, Grant 61871076; and the Fundamental Research Funds for the Central Universities (Grant ZYGX2019J001). *(Corresponding author: Guopeng Zhang.)*

Ya Zhou and Guopeng Zhang are with the Engineering Research Center of Mine Digitalization, Ministry of Education, and the School of Computer Science and Technology, China University of Mining and Technology, Xuzhou 221116, China (e-mail: zhou_ya@cumt.edu.cn; gpzhang@cumt.edu.cn).

Kezhi Wang is with the Department of Computer and Information Science, Northumbria University, Newcastle NE1 8ST, U.K. (e-mail: kezhi.wang@northumbria.ac.uk).

Kun Yang is with School of Electronic and Information Engineering, Nanjing University of Information Science and Technology, Nanjing, China, and also with School of Information and Communication Engineering, University of Electronic Science and Technology of China, Chengdu, China. (e-mail: kunyang@uestc.edu.cn).

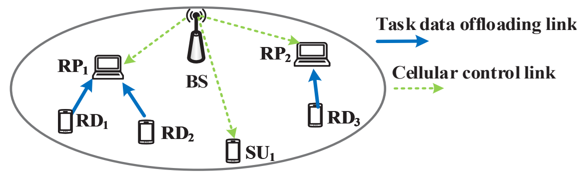


Fig. 1. The task offloading process in a cooperation group.

Against the above background, the main contribution of this paper is as follows: (1) The aim is to minimize the overall energy consumption of all UEs in a MUCC system through optimizing the user association and resource allocation. This problem is formulated as a mix integer nonlinear programming (MINLP) problem; (2) Due the high computational complexity of MINLP, we employ matching theory to design a low-complexity algorithm for finding the suboptimal solution of the problem. Specially, the energy consumption-related user preference functions are defined to solve the user role assignment and user association problems under the matching theory framework. Furthermore, a successive convex approximation (SCA) based task offloading algorithm is proposed to minimize the energy consumption of the whole system.

II. System Model

As shown in Fig. 1, we consider a MUCC system consisting of a set $\mathcal{N} \triangleq \{1, 2, \ldots, N\}$ of N UEs which can establish connections with each other via direct communications. Let $\mathcal{T} \triangleq \{t = 1, 2, ...\}$ denote the sequence of the time slots. The length of slot t ($\forall t \in \mathcal{T}$) is τ. In slot t, UE m ($\forall m \in \mathcal{N}$) has a computing task to execute, which is represented by $\phi_m = (L_m, C_m)$, where L_m (in bits) denotes the amount of task data to be processed, whereas C_m (in CPU cycles/bit) denotes the number of CPU cycles required to be executed for each data bit. To complete task ϕ_m in a slot, the CPU frequency of UE m is adjusted to $f_m = C_m L_m / \tau$, by using the dynamic voltage and frequency scaling (DVFS) technique [2]. Let γ_m denote the effective capacitance coefficient of the CPU of UE m. Then, the energy spent by UE m is as

$$E_m^{\$} = \gamma_m f_m^2 C_m L_m = \gamma_m C_m^3 L_m^3 / \tau^2, \ \ \forall m \in \mathcal{N}. \quad (1)$$

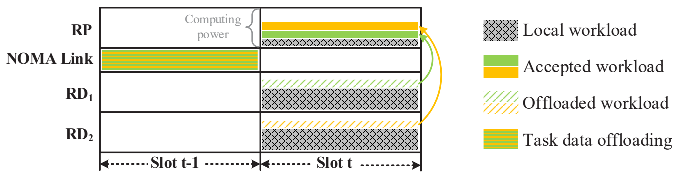


Fig. 2. The task offloading process in a cooperation group.

In any slot t, the UEs in set $\mathcal{N}$ can be divided into the following three subsets, i.e., the RD set $\mathcal{N}^{\mathbb{D}}$, the RP set $\mathcal{N}^{\mathbb{P}}$, and the SU set $\mathcal{N}^{\$}$. Then, we have

$$\mathcal{N}^x \cap \mathcal{N}^y = \emptyset, \ \ \forall x, y \in \{\mathbb{D}, \mathbb{P}, \$\} \text{ and } x \neq y, \tag{2}$$

$$\mathcal{N}^{\mathbb{D}} \cup \mathcal{N}^{\mathbb{P}} \cup \mathcal{N}^{\$} = \mathcal{N}. \tag{3}$$

Let i ($\forall i \in \mathcal{N}^{\mathbb{D}}$) denote the index of an RD, and j ($\forall j \in \mathcal{N}^{\mathbb{P}}$) the index of an RP. Any RD i can partition its task ϕ_i into two parts. One part is processed locally, while the other part, with data size $l_{i,j}$ is offloaded to RP j for cooperative computing. Then, we have

$$0 \leq l_{i,j} \leq L_i, \ \ \forall i \in \mathcal{N}^{\mathbb{D}}, \, \forall j \in \mathcal{N}^{\mathbb{P}}. \tag{4}$$

We define the indicator function $\mathbb{1}_{0\leq l_{i,j} \leq L_i} \in \{0,1\}$ to represent the association between the RDs and the RPs. If RD i offloads part of its task ϕ_i to RP j, $\mathbb{1}_{0< l_{i,j} \leq L_i} = 1$, otherwise, $\mathbb{1}_{l_{i,j}=0} = 0$. In addition, RD i can choose at most one RP j to offload task ϕ_i, that is

$$\sum_{j\in\mathcal{N}^{\mathbb{P}}} \mathbb{1}_{l_{i,j}} \leq 1, \ \forall i \in \mathcal{N}^{\mathbb{D}}, \tag{5}$$

In order to avoid allocating too many RDs to RP j in a time slot, we preset a parameter a_j ($a_j \geq 1$) representing the maximum number of RDs that can be served by RP j in a slot, that is

$$\sum_{i\in\mathcal{N}^{\mathbb{D}}} \mathbb{1}_{l_{i,j}} \leq a_j, \ \forall j \in \mathcal{N}^{\mathbb{P}}. \tag{6}$$

The set of RDs that are associated with RP j can thus be denoted by $\mathcal{M}_j \triangleq \left\{i \in \mathcal{N}^{\mathbb{D}} \mid \mathbb{1}_{l_{i,j}} = 1\right\}$.

Since the UEs have limited battery energy, any UE j can be used as a RP only when the available energy E_j^{Ax} is greater than the threshold $E_j^{\min}$. So we get the following constraint

$$E_j^{\text{Ax}} \geq E_j^{\min}, \ \forall j \in \mathcal{N}^{\mathbb{P}}. \tag{7}$$

Next, we introduce the task offloading process in a cooperation group composed of RP j and the associated RDs in set $\mathcal{M}_j$. As illustrated in Fig. 2, the task offloading will span the following two consecutive slots.

Slot $t-1$ (*Task data offloading*): In slot $t-1$, all the RDs in set $\mathcal{M}_j$ simultaneously transfer their task data to RP j by using non-orthogonal multiple access (NOMA) [3], while the system allocates orthogonal channels (of the same size of w MHz) for different RPs to receive the task data. Let $p_{i,j}$ denote the transmit power of RD i associated with RP j, and $g_{i,j}$ denote the channel gain between RD i and RP j. When successive interference cancellation (SIC) [3] is used, the achievable data rate of RD i is given by

$$r_{i,j} = w \log_2 \left(1 + \frac{p_{i,j} g_{i,j}}{\sum_{n\in\mathcal{M}_j\setminus\{i\}} p_{n,j} g_{n,j} + \sigma^2}\right), \tag{8}$$

where σ^2 is the noise power at the receiver of RP j. For correct receiving the task data of RD i of $l_{i,j}$ bits, the lower bound of the transmit power of RD i is obtained as [3]

$$p_{i,j} = \frac{\sigma^2}{g_{i,j}} \left(2^{\frac{l_{i,j}}{\tau w}} - 1\right) 2^{\frac{1}{\tau w}\sum_{n\in\mathcal{M}_j\setminus\{i\}} l_{n,j}} \leq P_i^{\max}, \tag{9}$$

where $P_i^{\max}$ is the maximum transmit power of RD i.

By using eq. (9), the energy consumption of RD i in slot $t-1$ for task data offloading is obtained as

$$E_{i,j}^{\text{Tx}} = p_{i,j}\tau, \ \ \forall i \in \mathcal{M}_j, \, \forall j \in \mathcal{N}^{\mathbb{P}}. \tag{10}$$

Slot t (*Cooperative computing*): After collecting the task data of all the RDs in $\mathcal{M}_j$, RP j can increase the CPU frequency as $f_j = \left(L_j + \sum_{i\in\mathcal{M}_j} \mathbb{1}_{l_{i,j}} l_{i,j}\right)/\tau$ for completing the accepted tasks as well as its own task in slot t. The energy spent by RP j for processing the tasks is given by

$$E_j^{\mathbb{P}} = \gamma_j C_j^3 \left(L_j + \sum_{i\in\mathcal{M}_j} \mathbb{1}_{l_{i,j}} l_{i,j}\right)^3 /\tau^2, \ \ \forall j \in \mathcal{N}^{\mathbb{P}}. \tag{11}$$

Since the size of the result is small, the energy spent by RP j for feeding back the result is ignored.

In the meanwhile, RD i ($\forall i \in \mathcal{M}_j$) can save its energy by decreasing the CPU frequency to $f_i = \left(L_i - \sum_{j\in\mathcal{N}_{\mathbb{P}}} \mathbb{1}_{l_{i,j}} l_{i,j}\right)/\tau$ for processing the remaining part of task ϕ_i. Therefore, the energy spent by RD i for local execution in slot t is given by

$$E_i^{\text{Cx}} = \gamma_i C_i^3 \left(L_i - \sum_{j\in\mathcal{N}_{\mathbb{P}}} \mathbb{1}_{l_{i,j}} l_{i,j}\right)^3 /\tau^2, \ \ \forall i \in \mathcal{N}^{\mathbb{D}}. \tag{12}$$

Then, the total energy spent by RD i to complete task ϕ_i is given by

$$E_i^{\mathbb{D}} = E_i^{\text{Cx}} + \sum_{j\in\mathcal{N}^{\mathbb{P}}} \mathbb{1}_{l_{i,j}} E_{i,j}^{\text{Tx}}, \ \ \forall i \in \mathcal{N}^{\mathbb{D}}. \tag{13}$$

III. Problem Presentation and Solution

Our objective is to minimize the overall energy consumption of all the UEs in the system, which can be given by

$$\min_{l_{i,j},\, \mathbb{1}_{l_{i,j}}} \ \sum_{i\in\mathcal{N}^{\mathbb{D}}} E_i^{\mathbb{D}} + \sum_{j\in\mathcal{N}^{\mathbb{P}}} E_j^{\mathbb{P}} + \sum_{n\in\mathcal{N}^{\$}} E_n^{\$} \tag{14}$$

$$\text{s.t.} \quad (2),\ (3),\ (4),\ (5),\ (6),\ (7),\ (9).$$

Problem (14) is a MINLP, which is very difficult to solve in general. Next, we present an effective algorithm to find the sub-optimal solution of this problem based on matching theory. Note that we first do not consider constraint (7), thus allowing any UE to be used as a RP no matter how much energy it remains. This constraint will be dealt with in Sec. III-D by proposing a user role control operation. Furthermore, it should be noted that the cooperative UEs in slot $(t-1)$ can perform the computing tasks accepted in slot $(t-2)$, and the communication link in slot t can be used to transfer the task data required in slot $(t+1)$. Hence, problem (14) minimizes the energy consumption of the cooperative UEs in each slot.

A. User role assignment

From the view of one-to-one roommate matching [6], an RP and the associated RD can be seen as a pair of roommate, and the UEs who do not get matched as the SUs. The user benefit is defined as the measure for two agents if they can be matched. Assume UE m ($\forall m \in \mathcal{N}$) is matched to UE k ($\forall k \in \mathcal{N}$ and $m \neq k$), where UE m and UE k respectively act as the RD and the RP in the cooperation, then, they will get the same benefit as

$$U_{m,k} = U_{k,m} = \max_{l_{i,j}} \left(\left(E_m^{\$} + E_k^{\$} \right) - \left(E_m^{\mathbb{D}} + E_k^{\mathbb{P}} \right) \right), \quad (15)$$

where $\left(E_m^{\$} + E_k^{\$} \right)$ denotes the total energy consumption of UEs m and k when they work independently, whereas $\left(E_m^{\mathbb{D}} + E_k^{\mathbb{P}} \right)$ denotes the total energy consumption when they cooperate. Hence, the proposed benefit function (15) gives the maximum energy that UEs m and k can save through cooperation. Function (15) is a special case of problem (18) which can be solved by the proposed SCA algorithm.

By solving function (15), any UE m can rank the other UEs in set $\mathcal{N}$ according to the benefits that they can get from matching with other UEs. The preference list for UE m to choose roommates is set to $\mathcal{PL}_m = \left\{ U_{m,k}^{(1)}, \cdots, U_{m,N-1}^{(N-1)} \right\}$, where $U_{m,k}^{(1)}$ and $U_{m,N-1}^{(N-1)}$ denote the most preferred and the least preferred UEs for UE m, respectively. Then, the role of each UE can be determined by using the Irving algorithm [6], which is presented in Appendix A of the supplementary material [1]. The RP and RD in each successful matching pair are put into sets $\mathcal{N}^{\mathbb{P}}$ and $\mathcal{N}^{\mathbb{D}}$, respectively, but the UEs which can not successfully match a roommate are put into the SU set $\mathcal{N}^{\$}$, rather than simply rejected as in the classical *Irving algorithm*. As a result, the UE set $\mathcal{N}$ is partitioned into three subsets $\mathcal{N}^{\mathbb{D}}$, $\mathcal{N}^{\mathbb{P}}$ and $\mathcal{N}^{\$}$.

If two UEs i and j ($i \neq j$) both have the incentive to leave their current partners and form a new pair with each other, these two UEs form a blocking pair (i, j). One can see from Appendix B of the supplementary material[1] that the matching result of the proposed user role assignment algorithm is stable as there exists no blocking pair in the matching result.

B. User association

After obtaining the RD and RP sets, we consider that one RP can serve multiple RDs in a slot. This user association problem can be seen as a multi-to-one two-side matching, which can be transformed into the college admission problem (CAP) [7] with the aim to achieve a stable multi-to-one matching. In the user association problem, the two disjoint UE sets, i.e., the RP set $\mathcal{N}^{\mathbb{P}}$ and the RD set $\mathcal{N}^{\mathbb{D}}$ correspond to the college set and the student set, respectively. The benefit that RD i can get after associating with RP j is shown as the energy saving for RD i with the help of RP j, then, one has

$$U_{i,j}^{\mathbb{D}} = \max_{l_{i,j}} \left(E_i^{\$} - E_i^{\mathbb{D}} \right). \quad (16)$$

The benefit or the energy saving which RP j can get is

$$U_{j,i}^{\mathbb{P}} = \max_{l_{i,j}} \left(\left(E_i^{\$} + E_j^{\$} \right) - \left(E_i^{\mathbb{D}} + E_j^{\mathbb{P}} \right) \right). \quad (17)$$

Following (16), the preference list of RD i is defined as $\mathcal{PL}_i^{\mathbb{D}}$, which is a ranking of RD i over all the RPs in set $\mathcal{N}^{\mathbb{P}}$ in descending order according to $U_{i,j}^{\mathbb{D}}$ $\left(\forall j \in \mathcal{N}^{\mathbb{P}} \right)$. Similarly, following (17), the preference list of RP j is defined as $\mathcal{PL}_j^{\mathbb{P}}$, which is a ranking of RP j over all the RDs in set $\mathcal{N}^{\mathbb{D}}$ in descending order according to $U_{j,i}^{\mathbb{P}}$ $\left(\forall i \in \mathcal{N}^{\mathbb{D}} \right)$.

Up to now, we have mapped the user association problem into the CAP. It can then be solved by using the Gale-Shapley (GS) algorithm, which is presented in Appendix C of the supplementary material[1]. Although the obtained matching result is stable, some RDs may not match their preferred RPs [7], [8]. Hence we propose the rotation swap operation (RSO) which allows RDs to associate with other RPs with better performance through exchanging their currently matched RPs without losing the stability. Before presenting the RSO, some definitions and notations are given below.

Definition 1. *Cabal: a cabal $\mathcal{K} = \{k_1, ..., k_x, ..., k_K\}$ is a subset of $\mathcal{N}^{\mathbb{D}}$, such that $\Omega(k_{x-1}) \succ_{k_x} \Omega(k_x)$ according to $\mathcal{PL}_{k_x}^{\mathbb{D}}$, $\forall k_x \in \mathcal{K}$ $(x - 1 = K$ when $x = 1)$.*

Definition 2. *Accomplice: the accomplice set $\mathcal{H}(\mathcal{K})$ of cabal $\mathcal{K}$ is a subset of $\mathcal{N}^{\mathbb{D}}$, such that $h \in \mathcal{H}(\mathcal{K})$ if 1) $h \notin \mathcal{K}$, for any $k_x \in \mathcal{K}$ if $\Omega(k_x) \succ_h \Omega(h)$ and $h \succ_{\Omega(k_x)} k_x$, or 2) $h \in \mathcal{K}$ and $h = k_l$ ($\forall k_l \in \mathcal{K}$), for any $k_x \in \mathcal{K}$ and $x \neq l$ if $\Omega(k_x) \succ_{k_l} \Omega(k_{l-1})$ and $k_l \succ_{\Omega(k_x)} k_{x+1}$.*

Definitions 1 and **2** show that the RDs in set $\mathcal{H}(\mathcal{K})$ may have prevented the RDs in set $\mathcal{K}$ from matching their more preferred RPs. Next, we propose the falsify operation, which enables the RDs in set $\mathcal{H}(\mathcal{K})$ to help the RDs in set $\mathcal{K}$ match more preferred RPs. The detail is given below.

Let $s \succ_i n$ denote that RD i prefers RP s over RP n in $\mathcal{PL}_i^{\mathbb{D}}$. Let $\Omega(i)$ denote the partner of RD i obtained by using the GS algorithm. Assume $\Omega_{\mathrm{S}}(i)$ denotes the partner of RD i obtained by further performing the RSO. If $\Omega_{\mathrm{S}}(i) \succeq_i \Omega(i)$, we say that matching Ω_{S} is "at least as good as" matching Ω, which is denoted by $\Omega_{\mathrm{S}} \geq \Omega$. We rewrite the preference list of RD i as $\mathcal{PL}_i^{\mathbb{D}} = \left(\mathcal{PL}_L^{\mathbb{D}}(i), \Omega(i), \mathcal{PL}_R^{\mathbb{D}}(i) \right)$, where $\mathcal{PL}_L^{\mathbb{D}}(i)$ and $\mathcal{PL}_R^{\mathbb{D}}(i)$ denote the RPs that are ranked higher and lower than $\Omega(i)$, respectively. Then, the falsify operation is to move RP θ ($\forall \theta \in \mathcal{PL}_L^{\mathbb{D}}(i)$) from $\mathcal{PL}_L^{\mathbb{D}}(i)$ to $\mathcal{PL}_R^{\mathbb{D}}(i)$. Let $\pi_r\left(\mathcal{PL}_i^{\mathbb{D}} \right)$ denote the random permutation of $\mathcal{PL}_i^{\mathbb{D}}$. Then, we can have the following Lemma.

Lemma 1. *Let $\mathcal{J} \subseteq \mathcal{N}^{\mathbb{D}}$. If all the RDs in set $\mathcal{J}$ ($\forall i \in \mathcal{J}$) submit their falsified lists in the form $\left(\pi_r\left(\mathcal{PL}_L^{\mathbb{D}}(i) - \theta \right), \Omega(i), \pi_r\left(\mathcal{PL}_R^{\mathbb{D}}(i) + \theta \right) \right)$, then $\Omega_S \geq \Omega$.*

Proof. See Appendix D of the supplementary material[1]. □

Lemma 1 shows that if the RDs which have matched RPs change the location of their rejected RPs in the preference list, the matching result will not change. The detail of the RSO in presented in **Algorithm 1**, wherein lines 4-10, the RDs in set $\mathcal{H}$ help the RDs in set $\mathcal{K}$ match their more preferred RPs through the falsify operation. In each round of RSO, most of

[1] Y. Zhou, G. Zhang, K. Wang, and K. Yang, "Matching-Theory-Based Multi-User Cooperative Computing Framework," *arXiv e-prints*, p. arXiv:2106.01551, Oct. 2021.

the existing algorithms, e.g. [7], [8], allow only two users to swap resources with each other, while **Algorithm 1** allows more than two users to swap resources at the same time, thus greatly improving the execution efficiency of the algorithm.

Algorithm 1 Rotation Swap Operation (RSO)

Let $\Omega = \Omega_0$, where Ω_0 is the matching result obtained by performing the Irving algorithm and the GS algorithm sequentially.
Find the largest cabal $\mathcal{K}$ from Ω by using **Definition 1**.
Find the accomplice $\mathcal{H}(\mathcal{K})$ of cabal $\mathcal{K}$ by using **Definition 2**.
for all RD $i \in \mathcal{K}$ **do**
  if $i \in \mathcal{H}(\mathcal{K}) - \mathcal{K}$ **then**
    RD i falsifies its preference list $\mathcal{PL}_i^{\mathbb{D}}$ as $\left(\pi_r(\mathcal{PL}_L^{\mathbb{D}}(i)-\theta),\ \Omega(i),\ \pi_r(\mathcal{PL}_R^{\mathbb{D}}(i)+\theta)\right)$, where $\theta = \{c \mid \Omega(m) \in \Omega(\mathcal{K}),\ m \succ_c m+1\}$.
  else
    RD i ($i = i_l, \forall i_l \in \mathcal{K}$) falsifies its preference list $\mathcal{PL}_i^{\mathbb{D}}$ as $\left(\pi_r(\mathcal{PL}_L^{\mathbb{D}}(i)-\theta),\ \Omega(i-1),\ \pi_r(\mathcal{PL}_R^{\mathbb{D}}(i)+\theta)\right)$, where $\theta = \{c \mid \Omega(m) \in \Omega(\mathcal{K}),\ w \succ_l \Omega(m-1),\ l \succ_w m+1\}$.
  end if
end for
return The modified preference lists.

C. Resource allocation

Now, the UEs in the system have been divided into multiple *cooperation groups*, each consisting of one RP and multiple RDs. Therefore, problem (14) can be rewritten as

$$\min_{l_{i,j},\, p_{i,j}} E_j^{\mathbb{P}} + \sum_{i \in \mathcal{M}_j} E_i^{\mathbb{D}} \tag{18}$$
$$\text{s.t.} \qquad (4),\ (9).$$

We note that problem (18) is non-convex due to the non-convexity of constraint (9). To address this issue, we define $N_0 = \sigma^2/g_{i,j}$ and $\alpha = 2^{1/\tau w}$, and convert constraint (9) into the following form

$$p_{i,j} = N_0 \left(\alpha^{\sum_{i\in\mathcal{M}_j} l_{i,j}} - \alpha^{\sum_{n\in\mathcal{M}_j\setminus\{i\}} l_{n,j}}\right) \le P_i^{\text{Max}}. \tag{19}$$

We obtain the partial derivatives of the $p_{i,j}$ with respect to $l_{i,j}$ and $l_{n,j}$ as

$$\triangledown_{l_{i,j}} p_{i,j} = N_0 \ln\alpha \left(\alpha^{\sum_{i\in\mathcal{M}_j} l_{i,j}}\right). \tag{20}$$

and

$$\triangledown_{l_{n,j}} p_{i,j} = N_0 \ln\alpha \left(\alpha^{\sum_{i\in\mathcal{M}_j} l_{i,j}} - \alpha^{\sum_{n\in\mathcal{M}_j\setminus\{i\}} l_{n,j}}\right). \tag{21}$$

respectively. Then, we get the following Lemma.

Lemma 2. *$\triangledown_{l_{i,j}} p_{i,j}$ and $\triangledown_{l_{n,j}} p_{i,j}$ are Lipschitz continuous on $l_{i,j}$ and $l_{n,j}$ with the constants L_C and L_F, respectively, where L_C and L_F are the Lipschitz constants.*

Proof. See Appendix E of the supplementary material[1]. □

If the first derivative of a non-convex function is Lipschitz continuous on the variables with the corresponding Lipschitz constants and each variable is nonempty, closed, and convex, then, the non-convex function can be converted to a convex one by SCA [3]. **Lemma 2** indicates that $p_{i,j}$ meets this rule and thus can be converted to a convex one by using SCA.

Let $\mathcal{D}_j \triangleq \left(l_{1,j}, ..., l_{|\mathcal{M}_j|,j}\right)^T$ denote the offloading strategy profile of the RDs in set $\mathcal{M}_j$. Let $\delta = 1, 2, ...$ denote the iterative numbers. Then, $l_{i,j}[\delta]$ represents the amount of task data offloaded from RD i to RP j in the δ^{th} iteration. By using eqs. (20) and (21), we derive the gradient of $p_{i,j}$ on $\mathcal{D}_j[\delta]$ as

$$\triangledown_{\mathcal{D}_j[\delta]} p_{i,j} = \left(\frac{\partial p_{i,j}}{\partial l_{1,j}[\delta]}, \frac{\partial p_{i,j}}{\partial l_{2,j}[\delta]}, ..., \frac{\partial p_{i,j}}{\partial l_{|\mathcal{M}_j|,j}[\delta]}\right)^T. \tag{22}$$

According to [9], a strongly convex function to approximate $p_{i,j}$ can then be constructed as

$$\widetilde{p}_{i,j} \triangleq p_{i,j}[\delta] + (\mathcal{D}_j - \mathcal{D}_j[\delta])^T \triangledown_{\mathcal{D}_j[\delta]} p_{i,j} + \frac{\lambda}{2} ||\mathcal{D}_j - \mathcal{D}_j[\delta]||_2 + \frac{1}{2} (\mathcal{D}_j - \mathcal{D}_j[\delta])^T (\mathcal{D}_j - \mathcal{D}_j[\delta]) \triangledown^2_{\mathcal{D}_j[\delta]} p_{i,j}. \tag{23}$$

where $\lambda \ge 0$. Now, we can approximate constraint (9) as

$$\widetilde{p}_{i,j} \le P_i^{\text{Max}}, \ \ \forall i \in \mathcal{M}_j. \tag{24}$$

By replacing constraint (9) with constraint (24), problem (18) is converted to a convex one and can be solved by using convex optimization tools, e.g., CVX [3]. The detail of the SCA algorithm is presented in Appendix F of the supplementary material[1].

D. Overall algorithm

Algorithm 2 Overall algorithm to solve problem (14)

1: Perform the *user role control* to limit the UEs with available energy less than the threshold, i.e., $E_m^{\text{Ax}} \le E_m^{\text{min}}$ to become RPs.
2: Perform the *Irving algorithm* to partition the UE set $\mathcal{N}$ into three subsets $\mathcal{N}^{\mathbb{P}}$, $\mathcal{N}^{\mathbb{D}}$, $\mathcal{N}^{\mathbb{S}}$.
3: Perform the *GS algorithm* to establish the association between the UEs in set $\mathcal{N}^{\mathbb{P}}$ and the UEs in set $\mathcal{N}^{\mathbb{D}}$.
4: **repeat**
5:   Perform *Algorithm 1*, i.e., the RSO to find *cabal* and falsify the *preference lists* of the UEs in the *cabal*.
6:   Perform the *GS algorithm* by using the modified *preference lists* to update the UE association, that is
7:   **if** there exist no $j \in \mathcal{N}^{\mathbb{P}}$ such that $\Omega(j) \succ_j \Omega_{\text{S}}(j)$ **then**
8:     update the UE association.
9:   **end if**
10: **until** the matching result has no cabal.
11: Perform the *SCA* algorithm for each *cooperation group* to obtain the resource allocation.

We present the overall algorithm for implementing the MUCC in **Algorithm 2**. The algorithm first performs the user role control operation (line 1), which limits the UEs with the remaining energy less than the threshold to become RPs, thus satisfying constraint (7). The complexity of the SCA algorithm (line 11) is related to the number of iterations δ as well as the complexity of the solver for the convex problems [3]. Since the adopted CVX toolbox is based on the *standard interior point method*, whose complexity is $\mathbf{O}\big(N^3\big)$ [10], the complexity of the SCA algorithm is thus $\mathbf{O}\big(\delta N^3\big)$. As analyzed in Appendix G of the supplementary material[1], the computational complexity of **Algorithm 2** is $\mathbf{O}\big(\delta N^3\big)$.

The proposed MUCC scheme is enabled by D2D communications. The effective range of the D2D communications is limited to one-hop, and a cellular base station (BS) is required to establish and manage the D2D links. Therefore, **Algorithm 2** can be performed at a BS in a *centralized* manner, as the BS can obtain the channel state information (CSI), and the

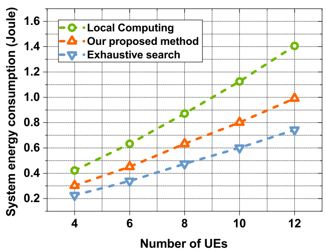


Fig. 3. The overall energy versus the number of UEs.

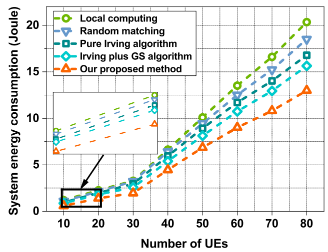
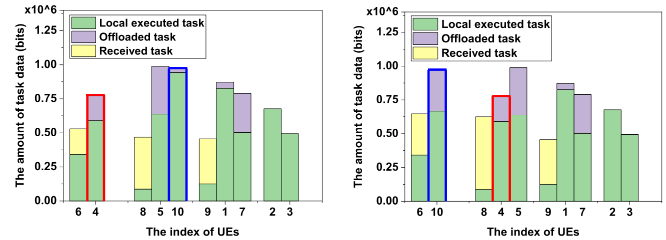


Fig. 4. The overall energy versus the number of UEs.



(a) Before performing the RSO.

(b) After performing the RSO.

Fig. 5. Task offloading results.

information of available energy and computing resource of all UEs through dedicated feedback channels. However, inaccurate CSI estimation will prevent the system from achieving the optimal performance. Please refer to Appendix G of the supplementary material[1] for a detailed analysis.

IV. Simulation Results

We consider a 100×100 m^2 rectangular area, where multiple UEs are randomly distributed. This is a general WLAN case that enables distributed UEs to perform D2D communications and further perform cooperative computing. Other parameters used in the simulation are given below. The number of CPU cycles required to execute one task bit is $C_m = 500$, and the effective capacitance coefficient is $\gamma_m = 10^{-28}$ [2]. The noise power is $\sigma^2 = 10^{-9}$. The maximum transmit power of the UE is set to 0.1 W. The duration of a time slot is $\tau = 0.2$ s. The bandwidth is $w = 1$ MHz. The channel power gain is set to $g_{i,j} = \zeta_{i,j}/d^{\beta}$ [7], where d is the distance between two UEs (in meters), $\beta = 3$ is the power-scaling path-loss factor, and $\zeta_{i,j}$ follows an exponential distribution with a unit mean, which captures the fading and shadowing effects. The maximum connection constraint of the UE is $a_m = 2$.

To verify the effectiveness of the proposed MUCC scheme, we compare our proposed algorithm with the exhaustive search (ES) algorithm. Although the ES algorithm can find the optimal solution of problem (14), the computational complexity is too high to be used in practice. The number of UEs in the system is set to less than 12. The simulation results in shown in Fig. 3. One can see from Fig. 3 that the performance loss of our proposed algorithm is small compared to the optimal solution obtained by using the ES algorithm, but our algorithm far outperforms the local computing method. It indicates that the proposed algorithm achieves a better compromise between computational complexity and optimal performance.

Next, we compare our proposed algorithm with the one-to-one random-matching algorithm, the pure Irving algorithm, and the Irving plus GS algorithm in a network with a larger number of UEs. The amount of task data for each UE is randomly distributed in $[0, 1]$ Mbits. Fig. 4 shows the overall energy versus the number of the UEs. One can see that our proposed algorithm achieves better energy efficiency over the other algorithms in all cases. The reason behind is that the proposed algorithm realizes the global user matching and resource allocation according to the available computing resources to the RPs, the required computing resources for the RDs, and the channel conditions between them, thus greatly reducing the overall energy consumption of the system.

Finally, we show how the proposed RSO influences the system performance when there are 10 UEs. Figs. 5(a) and 5(b) show the amount of offloaded data before and after performing the RSO, respectively. One can see that after performing the RSO, the amount of tasks offloaded by both UEs 4 and 10 are increased, thus reducing their energy consumption and further reducing the total energy consumption of the system. The reason behind is that the RDs in set $\mathcal{H}$ help the RDs in set $\mathcal{K}$ match their more preferred RPs through the falsify operation (lines 4-10 of **Algorithm 1**). Thus, the RSO can further decrease the overall energy consumption of the system through benefiting the RDs.

V. Conclusion and Future Works

This paper presents a MUCC scheme based on matching theory that answers the questions about who and how to cooperate with. In the future work, we will extend the single-hop MUCC scheme to the multi-hop MUCC scheme, where the task data of a RD could be forwarded by multiple relay UEs to reach the RP. In such a case, the energy consumption of all the relay UEs along a route should be considered when balancing the energy consumption for task execution and that for data transmission. In addition, meeting the delay requirements for user computing tasks is another huge challenge.

References

[1] X. Chen and J. Zhang, "When d2d meets cloud: Hybrid mobile task offloadings in fog computing," in *2017 ICC*, May 2017, pp. 1–6.

[2] Q. Lin, F. Wang, and J. Xu, "Optimal task offloading scheduling for energy efficient d2d cooperative computing," *IEEE Commun. Lett.*, vol. 23, no. 10, pp. 1816–1820, Oct 2019.

[3] D. Wu, F. Wang, X. Cao, and J. Xu, "Wireless powered user cooperative computation in mobile edge computing systems," in *2018 GC Wkshps*, Dec 2018, pp. 1–7.

[4] X. Cao, F. Wang, J. Xu, R. Zhang, and S. Cui, "Joint computation and communication cooperation for energy-efficient mobile edge computing," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4188–4200, Jun. 2019.

[5] Z. Sheng, C. Mahapatra, V. C. M. Leung, M. Chen, and P. K. Sahu, "Energy efficient cooperative computing in mobile wireless sensor networks," *IEEE Trans. on Cloud Comput.*, vol. 6, no. 1, pp. 114–126, Jan 2018.

[6] R. W. Irving, "An efficient algorithm for the "stable roommates" problem," *J Algorithms*, vol. 6, no. 4, pp. 577–595, 1985.

[7] Y. Gu, Y. Zhang, M. Pan, and Z. Han, "Matching and cheating in device to device communications underlying cellular networks," *IEEE J. Sel. Areas Commun.*, vol. 33, no. 10, pp. 2156–2166, Oct 2015.

[8] B. Wang, Y. Sun, S. Li, and Q. Cao, "Hierarchical matching with peer effect for low-latency and high-reliable caching in social iot," *IEEE Internet Things J.*, vol. 6, no. 1, pp. 1193–1209, Feb 2019.

[9] F. Facchinei, S. Sagratella, and G. Scutari, "Flexible parallel algorithms for big data optimization," in *2014 ICASSP*, May 2014, pp. 7208–7212.

[10] J. Xu and L. Qiu, "Energy efficiency optimization for mimo broadcast channels," *IEEE Trans. Wireless Commun.*, vol. 12, no. 2, pp. 690–701, February 2013.