Delay-sensitive Task Offloading with D2D Service-sharing in Mobile Edge Computing Networks

Hongru Zeng, Xian Li, Suzhi Bi, and Xiaohui Lin

***Abstract*—This letter considers a mobile edge computing (MEC) system where some mobile users (MUs) have not pre-installed the service program required to process their task data. Depending on the availability of the service program, and thus the ability to process data locally, users are tagged as eligible users (EUs) and ineligible users (IUs), respectively. Other than offloading all the task data to the edge server under a stringent spectrum, we consider a device-to-device (D2D) enabled service sharing method that allows IUs to obtain the service program from its adjacent EUs, such that some tasks can be processed locally at the user devices. To fully utilize the stringent wireless spectrum, we propose a spectrum-aggregation scheme that an EU who shares its service program to an IU can occupy the bandwidth originally allocated to both users. We aim to minimize the execution latency of all the MUs by jointly optimizing the service sharing, computation offloading, and bandwidth allocation. We formulate the problem as a mixed integer non-linear programming (MINLP), where the major difficulties are the combinatorial service sharing and task offloading decisions at UEs and the strong coupling between the integer decisions and system bandwidth allocation. To deal with this problem, we propose an efficient algorithm named MOP that first obtains the combinatorial decisions by formulating a service matching game, and accordingly derives the closed-form solution of the optimal bandwidth allocation. Simulation results show that MOP achieves less than 2.8% optimality gap compared to exhaustive search method, and offers substantial performance gain over the considered benchmark algorithms.**

***Index Terms*—Mobile edge computing (MEC), service placement, matching theory, resource allocation.**

I. Introduction

Via pushing computation resources towards network edges, MEC can effectively support computation-intensive and latency-critical applications, such as autonomous driving and virtual reality [1]. To complete a computation task in an MEC system, it requires both the task input data and the service program that processes the input. However, it is common that a mobile user (MU) has not pre-installed the required service program (e.g., MU newly arrived at an MEC network), thus needs to obtain the program from the service provider, e.g., the edge server (ES), to enable local task data processing. Extensive studies have been conducted to jointly optimize the service placement and task offloading in MEC system considering that the MUs directly download the service program from the ES [2]–[6]. However, task computing performance with direct service downloading is susceptible to the high path loss and deep shadowing between the ES and MU.

Enabling direct communication between two wireless devices in proximity, device-to-device (D2D) communication is a key solution to tackle the high path loss in MEC systems [7]–[10]. For example, [8] studied the joint computation and communication resource allocation problem in a three-node MEC, where a helper node either assists the MU in task processing or delivers the task data to the ES. [9] extended the study in [8] to a multi-user MEC system, and proposed a cooperative task processing policy to maximize the system computation throughput. Considering a wireless powered MEC system where an MU is assisted by multiple helper nodes, [10] jointly optimized the transmit energy beamforming and task offloading to maximize the system computation rate. However, all these works [8]–[10] assume that service program is available at all MUs and focus on optimizing task execution. When service program is absent at some MUs, besides downloading from the ES, users can also obtain the service program via D2D transmissions. In this case, it requires joint optimization of service sharing and task executions to achieve optimal computation performance. Solving for the optimal solution is very challenging in large-size networks due to the combinatorial D2D association decisions and their strong coupling with the system resource allocation.

In this paper, we study the joint service sharing and computation offloading problem in a D2D-enabled MEC network. In particular, we consider that the service program is initially available only at the ES and a subset of MUs. Depending on the availability of the service program, and thus the ability to process data locally, we refer to the two types of users as eligible users (EUs) and ineligible users (IUs), respectively. To efficiently utilize the computation power of MUs, we propose service data sharing between the EUs and IUs to enable parallel local data processing at user devices. The major contributions are: 1) To maximize the spectral usage, we propose a spectrum-aggregation scheme that an EU can occupy the bandwidth of another IU as long as it shares its service program. 2) We formulate a mixed integer non-linear programming (MINLP) problem to minimize the execution latency of all MUs. The problem involves jointly optimizing the binary decisions on service placement and computation offloading, and bandwidth allocation among MUs. 3) To handle the intractability of the problem, we propose a two-step algorithm named MOP, which first determines the binary service placement and task offloading decisions by formulating a service matching game, and then obtains the closed-form solution of optimal bandwidth allocation. 4) We prove the proposed MOP algorithm produces a stable solution and demonstrate with simulations that MOP achieves a near-optimal performance with low computational complexity.

This work was supported in part by the National Key Research and Development Program under Project 2019YFB1803305; in part by the National Natural Science Foundation of China under Project 61871271; in part by the Key Project of Department of Education of Guangdong Province under Grant 2020ZDZX3050; in part by the Shenzhen Science and Technology Program under Project JCYJ20210324093011030 and Project JCYJ20190808120415286. The associate editor coordinating the review of this article and approving it for publication was Kaoru Ota. (Corresponding author: Suzhi Bi.)

All the authors are with the College of Electronics and Information Engineering, Shenzhen University, Shenzhen 518060, China (email: 1910433060@email.szu.edu.cn; {xianli, bsz, xhlin}@szu.edu.cn). Suzhi Bi is also with Peng Cheng Laboratory, Shenzhen 518066, China.

II. System Model and Problem Formulation

A. D2D Service Sharing Model

We consider an MEC network consisting of an ES and N MUs. All the MUs have tasks of the same type that can be processed by a certain service program. Without loss of generality, we assume that the service program is initially available at the ES and the first M MUs (i.e., EUs), while unavailable at the rest $(N-M)$ MUs (i.e., IUs). Each MU i is allocated with a dedicated communication bandwidth B_i, $i=1,\cdots,N$, such that all MUs can transmit simultaneously without causing mutual interference. We consider a binary offloading policy that the task of an MU can be either computed locally or offloaded to the ES for edge processing [11]. To perform local computing, the IUs have to fetch the service program from either the ES or the EUs. Because an IU that computes locally does not transmit any task data, we propose a spectrum-aggregation scheme to fully utilize the spectrum. That is, suppose that an IU i fetches the service from a nearby EU j, EU j is allowed to temporarily occupy the bandwidth of both users B_i+B_j, for both service and task data transmissions. Notice that task computation and data communication can be performed simultaneously [6].

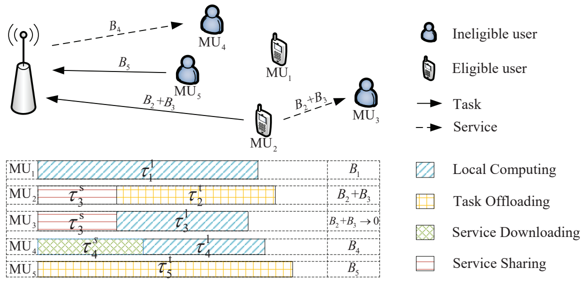
B_4
B_5
B_2+B_3
B_2+B_3
MU_4
MU_5
MU_1
MU_2
MU_3
Ineligible user
Eligible user
Task
Service
MU_1 τ_1^{l} B_1
MU_2 τ_3^{s} τ_2^{t} B_2+B_3
MU_3 τ_3^{s} τ_3^{l} $B_2+B_3\to 0$
MU_4 τ_4^{s} τ_4^{l} B_4
MU_5 τ_5^{t} B_5
Local Computing
Task Offloading
Service Downloading
Service Sharing

Fig. 1: An example system operation model.

An example operation of the considered system is shown in Fig. 1, where τ_i^{t}, τ_i^{s} and τ_i^{l}, $\forall i=1,\cdots,N$, denote the time cost of MU i on task offloading, service transmission and local computing, respectively. For an EU, it can perform local data processing directly (e.g., the MU_1). In contrast, an IU either offloads its task data to the ES for edge computing (e.g., the MU_5) or performs local computing after obtaining the service program (e.g., the MU_3 and MU_4). In particular, MU_4 downloads the service data from the ES, while MU_3 obtains the service via D2D link from MU_2. Accordingly, the occupied bandwidth of MU_2 increases to B_2+B_3, with which MU_2 sends the service data to the MU_3, and can also offload its task data for edge processing. Notice that after receiving the service program, MU_3 and MU_4 become EUs in the next round of task execution. The bandwidth will also be re-allocated with potential new users entering the system.

B. Communication Model

We denote the set of EUs, IUs and all MUs as $\mathcal{N}_{\mathrm{E}}$, $\mathcal{N}_{\mathrm{I}}$ and $\mathcal{N}=\mathcal{N}_{\mathrm{E}}\cup\mathcal{N}_{\mathrm{I}}$, respectively. We denote the service downloading decision of IU i by a binary variable $S_{i\mathrm{e}}$. In particular, $S_{i\mathrm{e}}=1$ if the IU i downloads the service data from the ES and $S_{i\mathrm{e}}=0$ otherwise. Let P_{e} be the fixed transmit power at the ES and h_i be the channel gain between the ith MU and the ES. Then, the achievable rate of IU i for downloading service data from ES is

$$\bar{r}_i^{\mathrm{s}}=S_{i\mathrm{e}}B_i\log_2[1+P_{\mathrm{e}}h_i/(B_i\delta_i^2)],\forall i\in\mathcal{N}_{\mathrm{I}},\tag{1}$$

where δ_i^2 denotes the power spectrum density of additive white Gaussian noise (AWGN) at IU i.

On the other hand, we denote S_{ij} as the binary indicator, where $S_{ij}=1$ if IU i receives the service data from EU j and $S_{ij}=0$ otherwise. Let P_i be the fixed transmit power of MU i. Then, the achievable rate of IU i for fetching service data from EU j is

$$\tilde{r}_{ij}^{\mathrm{s}}=S_{ij}\tilde{B}_j\log_2[1+P_jg_{ij}/(\tilde{B}_j\delta_i^2)],\forall i\in\mathcal{N}_{\mathrm{I}},j\in\mathcal{N}_{\mathrm{E}},\tag{2}$$

where $\tilde{B}_j=B_i+B_j$ and g_{ij} represents the channel gain between IU i and EU j.

We assume for simplicity that the channel is reciprocal. Then, for an IU i, the task offloading rate is

$$r_i^{\mathrm{t}}=B_i\log_2[1+P_ih_i/(B_i\delta_e^2)],\forall i\in\mathcal{N}_{\mathrm{I}},\tag{3}$$

where δ_e^2 denotes the power spectrum density of AWGN at the ES. With the spectrum-aggregation policy, the task offloading rate of an EU j is

$$r_j^{\mathrm{t}}=\bar{B}_j\log_2[1+P_jh_j/(\bar{B}_j\delta_e^2)],\forall j\in\mathcal{N}_{\mathrm{E}},\tag{4}$$

where $\bar{B}_j=\sum_{i\in\mathcal{N}_{\mathrm{I}}}S_{ij}B_i+B_j$ is the available bandwidth at EU j. Here, we assume that each EU can associate with at most one IU, i.e., $\sum_{i\in\mathcal{N}_{\mathrm{I}}}S_{ij}\le 1,\forall j\in\mathcal{N}_{\mathrm{E}}$.

C. Computation Model

We denote the task data size of MU i as L_i in bits. We use a binary indicator C_i $(i\in\mathcal{N})$ to denote the task execution mode of MU i. When $C_i=1$, MU i offloads the task to the ES for edge computing. Here we neglect the edge CPU processing time due to the substantially more powerful computing power of the ES [11]. Then, the edge processing time of MU i equals task data transmission time

$$\tau_i^{\mathrm{t}}=L_i/r_i^{\mathrm{t}},\forall i\in\mathcal{N}.\tag{5}$$

When $C_i=0$, MU i performs local computing. Let f_i be the fixed CPU frequency at MU i. The local computing time is

$$\tau_i^{\mathrm{l}}=\phi L_i/f_i,\forall i\in\mathcal{N},\tag{6}$$

where ϕ is the required CPU cycles to process one bit of data.

For an EU j, it can perform local computing directly without service fetching. The total task computation time of EU j is

$$T_j^{\mathrm{eu}}=(1-C_j)\tau_j^{\mathrm{l}}+C_j\left(\textstyle\sum_{i\in\mathcal{N}_{\mathrm{I}}}S_{ij}\tau_i^{\mathrm{s}}+\tau_j^{\mathrm{t}}\right),j\in\mathcal{N}_{\mathrm{E}},\tag{7}$$

where $\tau_i^{\mathrm{s}}=D_{\mathrm{s}}/(S_{i\mathrm{e}}\bar{r}_i^{\mathrm{s}}+\sum_{j\in\mathcal{N}_{\mathrm{E}}}S_{ij}\tilde{r}_{ij}^{\mathrm{s}})$ is the time cost on service sharing and D_{s} is the service data size.

On the other hand, an IU can either offload the task data to the ES for edge processing or fetch the service program for local computing from the ES or an EU. Notice that it can commence local task processing only after obtaining the service program. Then, the total execution latency of IU i is

$$T_i^{\mathrm{iu}}=(1-C_i)(\tau_i^{\mathrm{l}}+\tau_i^{\mathrm{s}})+C_i\tau_i^{\mathrm{t}},i\in\mathcal{N}_{\mathrm{I}}.\tag{8}$$

D. Problem Formulation

We use a binary indicator b_i to denote the type of MU, i.e., $b_i=1$ if the ith MU is an EU or $b_i=0$ otherwise. Then, the total task computation time of the ith MU is $T_i=b_iT_i^{\mathrm{eu}}+(1-b_i)T_i^{\mathrm{iu}}$. In this paper, we aim to minimize the weighted computation time of all the MUs, i.e.,

$$\begin{aligned}
\min_{\boldsymbol{\tau},\boldsymbol{C},\boldsymbol{S},\boldsymbol{\mathcal{B}}}\quad & \textstyle\sum_{i\in\mathcal{N}}\theta_iT_i && (9a)\\
\mathrm{s.t.}\quad & C_i\in\{0,1\},\forall i\in\mathcal{N}, && (9b)\\
& S_{ij},S_{i\mathrm{e}}\in\{0,1\},\forall i\in\mathcal{N}_{\mathrm{I}},j\in\mathcal{N}_{\mathrm{E}}, && (9c)\\
& \textstyle\sum_{j\in\mathcal{N}_{\mathrm{E}}}S_{ij}+S_{i\mathrm{e}}+\mathcal{C}_i=1,\forall i\in\mathcal{N}_{\mathrm{I}}, && (9d)\\
& \textstyle\sum_{i\in\mathcal{N}_{\mathrm{I}}}S_{ij}\le 1,\forall j\in\mathcal{N}_{\mathrm{E}}, && (9e)\\
& \textstyle\sum_{i\in\mathcal{N}}B_i\le B, && (9f)\\
& \{\boldsymbol{\tau},\boldsymbol{C},\boldsymbol{S},\boldsymbol{\mathcal{B}}\}\in\mathcal{T}, && (9g)
\end{aligned}$$

where θ_i is the weighting factor of MU i. For instance, by setting $\theta_i = 1/N$, we minimize the average computation time. $\boldsymbol{C} = \{C_i, i \in \mathcal{N}\}$, $\boldsymbol{\mathcal{B}} = \{B_i, i \in \mathcal{N}\}$, $\boldsymbol{\tau} = \{\tau_n^{\mathrm{t}}, \tau_n^{\mathrm{l}}, \tau_i^{\mathrm{s}}, i \in \mathcal{N}_{\mathrm{I}}, n \in \mathcal{N}\}$, $\boldsymbol{S} = \{S_{ij}, S_{ie}, i \in \mathcal{N}_{\mathrm{I}}, j \in \mathcal{N}_{\mathrm{E}}\}$. (9d) and (9e) describe the constraint on service placement and task offloading. (9f) restricts the bandwidth allocation among MUs. (9g) is the constraint on execution delay, where $\mathcal{T}$ is detailed in (10) at the top of next page.

As shown in (9), the variables of different MUs are strongly coupled due to the dependency between service placement and task computing. Involving both integer and continuous variables, (9) is generally a hard MINLP. One potential method to solve (9) is enumerating all the $(M+2)^{N-M} \cdot 2^M$ possible combinations of $\boldsymbol{C}$ and $\boldsymbol{S}$, and solve for each combination the optimal $\boldsymbol{\tau}$ and $\boldsymbol{\mathcal{B}}$. However, this will incur prohibitively high computational complexity when either N or M is large. To deal with this problem, we introduce an efficient two-stage **M**atching-**Op**timization method (MOP) in Section III.

III. Delay-Sensitive Service Sharing and Resource Allocation

MOP solves (9) in two steps: it first uses a matching-based method to obtain the binary decisions on service placement and task offloading, and then computes the optimal bandwidth allocation in closed-form via convex optimization. The proposed MOP algorithm is performed at the ES, which has perfect system information of CSI and service and task distribution. Compared to task offloading and service transmission, the time and energy cost on system information acquisition are very small and thus neglected in our work.

A. Matching-based User Association

We first optimize the service placement and task offloading given bandwidth allocation. Through extensive offline experiments, we find that a simple equal bandwidth allocation method produces excellent performance, i.e., $B_i = B/N, \forall i \in \mathcal{N}$. Then, we rewrite (9) as

$$\min_{\boldsymbol{\tau}, \boldsymbol{C}, \boldsymbol{S}} \sum_{i \in \mathcal{N}} \theta_i T_i, \quad \mathrm{s.t.} \quad (9\mathrm{b}) - (9\mathrm{e}) \text{ and } (9\mathrm{g}). \tag{11}$$

Solving (11) requires constructing service associations between IUs and service holders, i.e, the ES and EUs. In the following, we leverage matching theory [12] to efficiently solve (11) by formulating a service matching game between the IUs and EUs.

Definition 1. *For each* IU $i \in \mathcal{N}_{\mathrm{I}}$ *and* EU $j \in \mathcal{N}_{\mathrm{E}}$*, the service matching* Ω_{S} *constructs mappings from the set* $\mathcal{N}_{\mathrm{I}}$ *to the set of all subsets of* $\mathcal{N}_{\mathrm{E}} \cup \{\mathrm{ES}, \mathrm{IU}\ i\}$ *with*
1) $|\Omega_{\mathrm{S}}(i)| = 1, \forall i \in \mathcal{N}_{\mathrm{I}}$*;* $|\Omega_{\mathrm{S}}(j)| = 1, \forall j \in \mathcal{N}_{\mathrm{E}}$*;*
2) $|\Omega_{\mathrm{S}}(\mathrm{ES})| \leq N$*;*
3) $\Omega_{\mathrm{S}}(i) = j \Leftrightarrow \Omega_{\mathrm{S}}(j) = i$*.*

The first two conditions denote that each EU can be associated with only one IU, while the ES can be associated with multiple IUs. The third condition captures the symmetry of service matching, i.e, associating IU i to an EU j is equivalent to associating EU j with IU i. Notice that when $\Omega_{\mathrm{S}}(i) = i$, the IU i performs task offloading to the ES without the need of acquiring the service data, i.e., $C_i = 1$ in (8).

For an IU $i \in \mathcal{N}_{\mathrm{I}}$ and two devices $k, k' \in \mathcal{N}_{\mathrm{E}} \cup \{\mathrm{ES}, \mathrm{IU}\, i\}$, we say that IU i prefers k over k' if $T_{i,k}^{\mathrm{iu}} < T_{i,k'}^{\mathrm{iu}}$, and denote the preference as

$$k' \prec_i k, \text{if } T_{i,k}^{\mathrm{iu}} < T_{i,k'}^{\mathrm{iu}}, k, k' \in \mathcal{N}_{\mathrm{E}} \cup \{\mathrm{ES}, \mathrm{IU}\, i\}. \tag{12}$$

Algorithm 1: The Matching-based User Association

```
Input: $\Omega_{\mathrm{S}}(i) = \Omega_{\mathrm{S}}(j) = \varnothing$, $\tilde{\mathcal{N}}_{\mathrm{I},j} = \varnothing$, $\forall i \in \mathcal{N}_{\mathrm{I}}$,
       $j \in \mathcal{N}_{\mathrm{E}}$; $\mathbb{P}_i = \{k_{i,1}, \ldots, k_{i,m_i}, \ldots, k_{i,M+2}\}$, $\forall i \in \mathcal{N}_{\mathrm{I}}$.
1  Initializing $m_i = 1$, $\forall i \in \mathcal{N}_{\mathrm{I}}$.
2  while $\Omega_{\mathrm{S}}(i) = \varnothing$ for some $i$ do
3      for all $i \in \mathcal{N}_{\mathrm{I}}$ do
4          if $\Omega_{\mathrm{S}}(i) = \varnothing$ then
5              IU $i$ proposes to $k_{i,m_i}$.
6              if $k_{i,m_i}$ = ES or IU $i$ then
7                  $\Omega_{\mathrm{S}}(i) = k_{i,m_i}$.
8              else
9                  if $k_{i,m_i}$ = EU $j$ then
10                     Add $i$ into $\tilde{\mathcal{N}}_{\mathrm{I},j}$.
11             Set $m_i = m_i + 1$.
12     for all $j \in \mathcal{N}_{\mathrm{E}}$ do
13         Compute $i^* = \arg\max_{i \in \tilde{\mathcal{N}}_{\mathrm{I},j}} \{\Psi_j(i)\}$.
14         Set $\Omega_{\mathrm{S}}(j) = i^*$ and $\tilde{\mathcal{N}}_{\mathrm{I},j} = \{i^*\}$.
15 Each IU $i \in \mathcal{N}_{\mathrm{I}}$ determines $C_i$ and $S_{ij}$ using (14); Each EU
       $j \in \mathcal{N}_{\mathrm{E}}$ determines $C_j = 0$ or $1$ that minimizes $T_j^{\mathrm{eu}}$ in (7).
Output: service and task association
```

Here, $T_{i,k}^{\mathrm{iu}}$ is the total task computation time of IU i with $\Omega_{\mathrm{S}}(i) = k$, i.e., setting $S_{i,k} = 1$ in (8). By enumerating all the $M+2$ devices $k \in \mathcal{N}_{\mathrm{E}} \cup \{\mathrm{ES}, \mathrm{IU}\ i\}$, we construct the preference list of IU i as $\mathbb{P}_i = \{k_{i,1}, \ldots, k_{i,m_i}, \ldots, k_{i,M+2}\}$, where $k_{i,m_i+1} \prec_i k_{i,m_i}$ for $m_i = 1, \cdots, M+1$.

Suppose that an EU j receives service association request from IU i, it calculates the reduced computation time of IU i by accepting the request, which is given by $\Psi_j(i) = T_{i,i}^{\mathrm{iu}} - T_{i,j}^{\mathrm{iu}}$. For an EU j receiving multiple requests from a set $\tilde{\mathcal{N}}_{\mathrm{I},j}$ of IUs, we define the preference of EU j as

$$q' \prec_j q, \text{ if } \Psi_j(q') < \Psi_j(q), \ q, q' \in \tilde{\mathcal{N}}_{\mathrm{I},j}. \tag{13}$$

Based on preference lists of all IUs, we determine the service matching solution in an iterative manner as below: First, all IUs make requests on service association according to their preference list in parallel. Specifically, IU i requests association from the service provider k_{i,m_i} (i.e., the m_ith element in $\mathbb{P}_i$ and we set an initial $m_i = 1$). Second, the service providers $\{\mathcal{N}_{\mathrm{E}} \cup \mathrm{ES}\}$ decide whether to accept the requests. The ES accepts all the requests from IUs, while an EU j accepts the IU i with maximum performance gain up to the current iteration. The rejected IUs will propose to the next most favorable service provider k_{i,m_i+1} in the next loop. The iteration repeats until all IUs are associated.

Once the service association $\Omega_{\mathrm{S}}(i)$ is determined for all $i \in \mathcal{N}_{\mathrm{I}}$, the task offloading decisions of the IUs are accordingly determined as

$$\begin{cases} C_i = 0, S_{ij} = 1, & \text{if } \Omega_{\mathrm{S}}(i) = j, j \in \mathcal{N}_{\mathrm{E}}, & (14\mathrm{a}) \\ C_i = 1, & \text{if } \Omega_{\mathrm{S}}(i) = i, & (14\mathrm{b}) \\ C_i = 0, S_{ie} = 1, & \text{if } \Omega_{\mathrm{S}}(i) = \mathrm{ES}. & (14\mathrm{c}) \end{cases}$$

After determining the service association S_{ij}'s, each EU performs either edge computing or local computing by simply comparing the two options. That is, each EU $j \in \mathcal{N}_{\mathrm{E}}$ selects C_j=0 or 1 that minimizes the task completion time T_j^{eu} in (7).

We summarize matching-based user association in Algorithm 1. Initially, we set $\Omega_{\mathrm{S}}(i) = \tilde{\mathcal{N}}_{\mathrm{I},j} = \varnothing$ for all the devices $i \in \mathcal{N}_{\mathrm{I}}$ and $j \in \mathcal{N}_{\mathrm{E}}$, and construct the preference list $\mathbb{P}_i, \forall i$, according to (12). Then, in line 2-14, we update Ω_{S} according to the association decisions until each IU has

$$\begin{cases} \boldsymbol{\mathcal{T}} = \{\boldsymbol{\tau}, \boldsymbol{C}, \boldsymbol{S}, \boldsymbol{\mathcal{B}} \mid (S_{ij}+S_{\mathrm{ie}})[D_{\mathrm{s}}/(S_{\mathrm{ie}}\bar{r}_i^{\mathrm{s}} + \sum_{j\in\mathcal{N}_{\mathrm{E}}} S_{ij}\tilde{r}_{ij}^{\mathrm{s}})] \le \tau_i^{\mathrm{s}}, C_j(L_j/r_j^{\mathrm{t}}) \le \tau_j^{\mathrm{t}}, C_i(L_i/r_i^{\mathrm{t}}) \le \tau_i^{\mathrm{t}}, (1-C_j)(\phi L_j/f_j) \le \tau_j^{\mathrm{l}}, \\ \quad (1-C_i)(\phi L_i/f_i) \le \tau_i^{\mathrm{l}}, (1-C_i)(\tau_i^{\mathrm{s}}+\tau_i^{\mathrm{l}}) + C_i\tau_i^{\mathrm{t}} \le T_i^{\mathrm{iu}}, (1-C_j)\tau_j^{\mathrm{l}} + C_j(\sum_{i\in N_i} S_{ij}\tau_i^{\mathrm{s}} + \tau_j^{\mathrm{t}}) \le T_j^{\mathrm{eu}}, i\in\mathcal{N}_{\mathrm{I}}, j\in\mathcal{N}_{\mathrm{E}}\}. \end{cases} \tag{10}$$

a matching association. In line 15, we determine the task offloading decisions of all EUs. In the following, we show that the user association process achieves a stable matching defined as follows.

Definition 2. *Given a matching with $\Omega_{\mathrm{S}}(i) = j$ and $\Omega_{\mathrm{S}}(i') = j'$, (i, j') is said to be blocking the matching Ω_{S} and form a blocking pair if: 1)$j \prec_i j'$, 2)$i' \prec_{j'} i$. A matching Ω_{S} is stable if there is no blocking pair.*

In other words, a blocking pair (i, j') exists if both IU i and EU j' are better off by forming a new matching $\Omega_{\mathrm{S}}(i) = j'$. We prove the stability of MOP by considering the following two cases: 1) when an EU j receives request from only one IU (e.g., IU i) throughout the iterations, we have $\Omega_{\mathrm{S}}(i) = j$ and there is obviously no blocking pair for Ω_{S}. 2) When an EU j receives request from multiple IUs, and finally matches with IU i, i.e., $\Omega_{\mathrm{S}}(i) = j$. We suppose that IU i prefers another EU j' to its final matching EU j. Then, IU i must have already proposed to EU j' previously and got rejected. Therefore, EU j' should prefer its current matching IU to IU i. According to Definition 2, there is no blocking pair following the proposed matching procedure and MOP achieves a two-sided stable matching between the IUs and EUs.

The complexity of Algorithm 1 is analyzed as follows. First, the complexity of constructing the preference lists of all the IUs is $\mathcal{O}((N-M)M)$. Second, for each of the $(N-M)$ IUs, it will reach a matching after at most M rejections by the EUs. Therefore, the complexity of the service association is $\mathcal{O}((N-M)M)$. Finally, the complexity of determining the offloading decisions of all the users is $\mathcal{O}(N)$. Overall, the total complexity of the proposed algorithm is $\mathcal{O}((N-M)M)$, which is much more efficient compared to the exponential complexity of enumeration-based method.

B. Optimal Bandwidth Allocation

With the matching results, we introduce auxiliary variables ξ_j's and reduce the original problem (9) into the following bandwidth allocation problem:

$$\min_{\boldsymbol{\tau},\boldsymbol{\mathcal{B}},\boldsymbol{\xi}} \ \sum\nolimits_{i\in\mathcal{N}} \theta_i T_i \tag{15a}$$

$$\text{s. t.} \ \ \xi_j \le \sum\nolimits_{i\in\mathcal{N}_{\mathrm{I}}} S_{ij}B_i + B_j, j\in\mathcal{N}_{\mathrm{E}}, \tag{15b}$$

$$\{\boldsymbol{\tau}, \boldsymbol{\mathcal{B}}\} \in \mathcal{T}. \tag{15c}$$

(15) is a convex optimization problem and thus can be solved by off-the-shelf convex optimization tools. To further reduce the complexity, we derive the closed-form optimal solution using Lagrange duality method. For convenience, we denote $z_1 = P_i h_i/\delta_e^2$, $z_2 = P_{\mathrm{e}} h_i/\delta_i^2$, $z_3 = P_j g_{ij}/\delta_i^2$, $z_4 = P_j h_j/\delta_e^2$. We present the optimal solution in the following Proposition 1.

Proposition 1. *The optimal bandwidth allocation has the following structure:*

$$\xi_j^* = \begin{cases} \hat{B}_j^*, & \text{if } C_j = 1,\ \exists S_{ij} = 1, \forall i\in\mathcal{N}_{\mathrm{I}}, \quad (16a) \\ f(\lambda_5/\lambda_4, z_4), & \text{if } C_j = 1,\ S_{ij} = 0, \forall i\in\mathcal{N}_{\mathrm{I}}, \quad (16b) \\ f(\lambda_5/\lambda_3, z_4), & \text{if } C_j = 0,\ \exists S_{ij} = 1, \forall i\in\mathcal{N}_{\mathrm{I}}, \quad (16c) \\ 0, & \text{if } C_j = 0,\ S_{ij} = 0, \forall i\in\mathcal{N}_{\mathrm{I}}, \quad (16d) \end{cases}$$

$$B_i^* = \begin{cases} \check{B}_i^*, & \text{if } C_i = 0,\ \exists S_{ij} = 1, \forall j\in\mathcal{N}_{\mathrm{E}}, \quad (17a) \\ f(\lambda_1/\lambda_3, z_2), & \text{if } C_i = 0,\ S_{\mathrm{ie}} = 1, \quad (17b) \\ f(\lambda_1/\lambda_2, z_1), & \text{if } C_i = 1. \quad (17c) \end{cases}$$

Here, $\hat{B}_j^*$ is the unique solution of $\mathcal{L}_1'(\xi_j) = 0$ and can be efficiently obtained via bisection search, where $\mathcal{L}_1'(\xi_j)$ is given in (20). *$\check{B}_i^* = B$ if $\lambda_1 - \lambda_5 < 0$ and $\check{B}_i^* = 0$ otherwise. $f(\lambda, z) = -z/\left((\mathcal{W}(-1/\exp(1+\lambda\ln 2)))^{-1} - 1\right)$ where $\mathcal{W}(\cdot)$ is the Lambert $\mathcal{W}$ function. $\lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5$ denotes the non-negative Lagrangian multipliers.*

Proof. Please refer to Appendix A. □

With the semi-closed-form solution in (16) and (17) given the dual variables $\boldsymbol{\lambda}$, the optimal $\boldsymbol{\lambda}$ can be iteratively obtained via the ellipsoid method, which is omitted here due to the page limit [11]. Since there are $(2N+1)$ dual variables, the complexity of the subgradient method is $\mathcal{O}((2N+1)^2)$ [13]. As a result, the overall complexity of MOP method is $\mathcal{O}((N-M)M + (2N+1)^2)$. The result in Proposition 1 reveals that an MU with higher transmit power and better channel quality has a larger bandwidth for task offloading. Besides, when IU i fetches service from ES, the allocated bandwidth increases with the transmit power and channel gain.

IV. Simulation Results

In this section, we conduct numerical simulations to evaluate the proposed MOP method. We consider $N = 8$ MUs, where four IUs and four EUs are randomly deployed in a $150 \times 150\ \mathrm{m}^2$ area. The ES locates at the center of the area. The channel gain between the ES and the ith MU follows a path-loss model $h_i = G_A \left(\frac{3\times 10^8}{4\pi f_c d_i}\right)^{\sigma}$, where d_i denotes the distance between MU i and ES, $G_A = 4.11$ captures the total antenna gain, $f_c = 2.4$ GHz represents the carrier frequency, and $\sigma = 2.7$ denotes the path-loss exponent, respectively. We assume that the task data size follows uniform distribution $L_i \in U[50, 500]$ Mbits. All MUs have identical weighting factors, i.e., $\theta_i = 1, \forall i \in \mathcal{N}$. Unless otherwise stated, we set the system bandwidth $B = 10$ MHz and the transmit power of ES as $P_{\mathrm{e}} = 1$ W. For all MUs, we set equal transmit power $P_i = 0.2$ W, local CPU frequency $f_i = 2$ GHz, and the required CPU cycles per bit $\phi = 100$ cycles/bit, $\forall i \in \mathcal{N}$.

We consider four benchmarks for comparison:

- Matching in proximity (PROX): the preference list of an IU is ordered by its distance to the EUs and ES.
- Random matching (RAND): the IUs and EUs form service matching randomly.
- Edge computing only (ECO): all MUs offload their tasks to the ES (i.e., $C_i = 1, \forall i \in \mathcal{N}$).
- Optimal matching (OPTI): exhaustively enumerating all service association and task offloading combinations.

In Fig. 2(a), we first demonstrate the weighted sum of computation time (WSCT) under varying service data size D_{s}. As shown in the figure, the WSCT of ECO is unrelated to service data size D_{s}. As D_{s} increases, crossovers can be observed between PROX and ECO as well as RAND and ECO. This is because that IUs tend to perform edge processing when D_{s} is large. For the proposed MOP method, it shows a significant superiority over the benchmark methods for all considered D_{s}, and can achieve similar WSCT performance

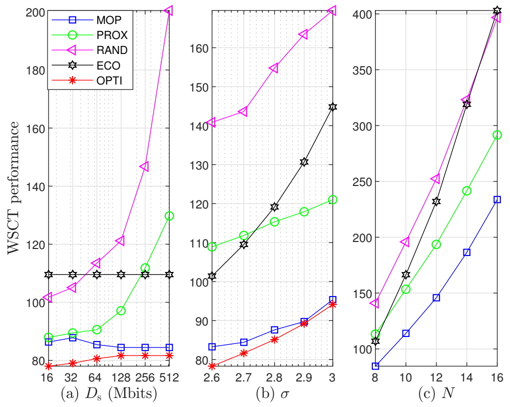

Fig. 2: The WSCT performance versus (a) service data size $D_{\rm s}$, (b) path-loss exponent σ and (c) number of MUs N.

as the OPTI method especially at large $D_{\rm s}$ (e.g., $D_{\rm s} \geq 128$ Mbits).

Then, we show in Fig. 2(b) the WSCT performance as a function of the path-loss exponent σ with $D_{\rm s} = 256$ Mbits. For all considered methods, the value of WSCT increases with σ, since a larger σ yields a more severe signal attenuation during service transferring and task offloading. For the ECO method, it gets a lower WSCT at small σ than PROX, while performs worse than PROX when σ gets large. This is because that the MUs prefer local computing to edge processing at large σ. Nevertheless, the proposed MOP method significantly outperforms the benchmarks and achieves near-optimal performance (on average 2.8% optimality gap) for all considered σ.

We also show in Fig. 2(c) the WSCT performance under a varying number of users from $M = 8$ to 16 (equal number of EUs and IUs). Due to the prohibitive computation complexity of OPTI under large M, we omit the result of OPTI here. The result in Fig. 2(c) shows that the WSCT increases with M for all considered methods. Besides, the proposed MOP method achieves a much lower cost than the other three benchmark methods for all considered M.

V. Conclusion

This letter studied the joint service placement and task offloading problem in a D2D-aided MEC network. We proposed a spectrum aggregation method and formulated an MINLP problem to minimize the computation delay of all MUs. To deal with the intractability of the problem in large-size network, we proposed an efficient matching-based method named MOP. The simulation results show that the proposed method achieves near-optimal delay performance.

Appendix A
Proof of Proposition 1

We prove Proposition 1 by considering different combinations of C and S. To start with, we write the partial Lagrangian function of (15) as

$$\begin{aligned}\mathcal{L}_1(\boldsymbol{\mathcal{B}},\boldsymbol{\tau},\boldsymbol{\lambda}) &= \textstyle\sum_i\sum_j(\theta_i T_i+\theta_j T_j)+\lambda_1\left(\sum_i B_i+\sum_j B_j-B\right)\\ &+\lambda_2 C_i\left[\frac{L_i}{\tau_i^{\rm t}}-B_i\log_2\left(1+\frac{P_i h_i}{B_i\delta_e^2}\right)\right]\\ &+\lambda_3(S_{i{\rm e}}+S_{ij})\left(\frac{Ds}{\tau_i^{\rm s}}-S_{i{\rm e}}\bar{r}_i^{\rm s}-\textstyle\sum_{j\in\mathcal{N}_{\rm E}}S_{ij}\tilde{r}_{ij}^{\rm s}\right) \quad (18)\\ &+\lambda_4 C_j\left[\frac{L_j}{\tau_j^{\rm t}}-\xi_j\log_2\left(1+\frac{P_j h_j}{\xi_j\delta_e^2}\right)\right]\\ &+\lambda_5\left(\xi_j-\textstyle\sum_{i\in\mathcal{N}_{\rm I}}S_{ij}B_i-B_j\right).\end{aligned}$$

In the case of $C_j = 1$ and $S_{ij} = 1$, $i \in \mathcal{N}_{\rm I}, j \in \mathcal{N}_{\rm E}$, we take the derivative of $\mathcal{L}_1$ with respect to B_i, ξ_j, and obtain that

$$\frac{\partial\mathcal{L}_1}{\partial B_i} = \lambda_1 - \lambda_5, \quad (19)$$

$$\begin{aligned}\mathcal{L}_1'(\xi_j) = \frac{\partial\mathcal{L}_1}{\partial\xi_j} &= \lambda_5 - \frac{\lambda_3}{\ln 2}\left[\ln\left(1+z_3/\xi_j\right)-\frac{z_3/\xi_j}{(1+z_3/\xi_j)}\right]\\ &-\frac{\lambda_4}{\ln 2}\left[\ln\left(1+z_4/\xi_j\right)-\frac{z_4/\xi_j}{(1+z_4/\xi_j)}\right].\end{aligned} \quad (20)$$

Lemma 1. *$\mathcal{L}_1'(\xi_j)$ is a monotonically increasing function of ξ_j and $\mathcal{L}_1'(\xi_j) = 0$ has a unique solution $\hat{B}_j^* \in [0, B]$.*

Proof. By taking the derivative of $\mathcal{L}_1'$ of ξ_j, we can find $\frac{\partial\mathcal{L}_1'}{\partial\xi_j} > 0$ with $\xi_j > 0$, that is, $\mathcal{L}_1'(\xi_j)$ is a monotonically increasing function of ξ_j. When $\xi_j \to 0$, $\mathcal{L}_1'(\xi_j) \to -\infty$. When $\xi_j \to B$, if $\mathcal{L}_1'(B) < 0$, then $\forall\xi_j \in [0, B], \mathcal{L}_1'(\xi_j) < 0$ and $\mathcal{L}_1$ get the minimum with $\hat{B}_j^* = B$; if $\mathcal{L}_1'(B) > 0$, then there is a unique solution $\hat{B}_j^* \in [0, B]$ for $\mathcal{L}_1'(\xi_j) = 0$. □

By solving $\frac{\partial\mathcal{L}_1}{\partial B_i} = 0$ and $\mathcal{L}_1'(\xi_j) = 0$, we obtain the results in (16a) and (17a). The results of other cases in (16) and (17) can be obtained similarly by plugging the corresponding matching pairs (C_j, S_{ij}), and are omitted here for brevity.

References

[1] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Commun. Surv. Tutor.*, vol. 19, no. 4, pp. 2322–2358, Aug. 2017.
[2] Z. Lin, S. Bi, and Y.-J. A. Zhang, "Optimizing AI service placement and resource allocation in mobile edge intelligence systems," *IEEE Trans. Wireless Commun.*, pp. 1–1, May 2021. [Online]. Available: 10.1109/TWC.2021.3081991
[3] J. Xu, L. Chen, and P. Zhou, "Joint service caching and task offloading for mobile edge computing in dense networks," in *Proc.IEEE INFOCOM*, Honolulu, HI, USA, Apr. 2018, pp. 207–215.
[4] S. Bi, L. Huang, and Y.-J. A. Zhang, "Joint optimization of service caching placement and computation offloading in mobile edge computing systems," *IEEE Trans. Wireless Commun.*, vol. 19, no. 7, pp. 4947–4963, Jul. 2020.
[5] K. Poularakis, J. Llorca, A. M. Tulino, I. Taylor, and L. Tassiulas, "Joint service placement and request routing in multi-cell mobile edge computing networks," in *Proc.IEEE INFOCOM*, Paris, France, Apr. 2019, pp. 10–18.
[6] X. Li, S. Bi, and H. Wang, "Optimizing resource allocation for joint AI model training and task inference in edge intelligence systems," *IEEE Wireless Commun. Lett.*, vol. 10, no. 3, pp. 532–536, Mar. 2021.
[7] Y. Wu, J. Chen, L. P. Qian, J. Huang, and X. S. Shen, "Energy-aware cooperative traffic offloading via device-to-device cooperations: An analytical approach," *IEEE Trans. Mobile Comput.*, vol. 16, no. 1, pp. 97–114, Jan. 2017.
[8] 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.
[9] Y. He, J. Ren, G. Yu, and Y. Cai, "D2D communications meet mobile edge computing for enhanced computation capacity in cellular networks," *IEEE Trans. Wireless Commun.*, vol. 18, no. 3, pp. 1750–1763, Mar. 2019.
[10] D. Wu, F. Wang, X. Cao, and J. Xu, "Wireless powered user cooperative computation in mobile edge computing systems," in *Proc. IEEE Globecom Workshops*, Abu Dhabi, UAE, Dec. 2018, pp. 1–7.
[11] S. Bi and Y. J. Zhang, "Computation rate maximization for wireless powered mobile-edge computing with binary computation offloading," *IEEE Trans. Wireless Commun.*, vol. 17, no. 6, pp. 4177–4190, Jun. 2018.
[12] 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.
[13] S. Boyd and L. Vandenberghe, *Convex Optimization.* Cambridge University Press, 2004.