Task Offloading for End-Edge-Cloud Orchestrated Computing in Mobile Networks

Chuan Sun[1], Hui Li[1], Xiuhua Li[1*], Junhao Wen[1], Qingyu Xiong[1], Xiaofei Wang[2], and Victor C. M. Leung[3, 4]
[1]School of Big Data & Software Engineering, Chongqing University, Chongqing, China
[2]TKLAN, College of Intelligence & Computing, Tianjin University, Tianjin, China
[3] College of Computer Science & Software Engineering, Shenzhen University, Shenzhen, China
[4]Department of Electrical & Computer Engineering, The University of British Columbia, Vancouver, Canada
*Corresponding author: Xiuhua Li (lixiuhua@cqu.edu.cn)
Email:{c.sun, h.li, lixiuhua, jhwen, xiong03}@cqu.edu.cn, xiaofeiwang@tju.edu.cn, vleung@ece.ubc.ca

***Abstract*—Recently, mobile edge computing has received widespread attention, which provides computing infrastructure via pushing cloud computing, network control, and storage to the network edges. To improve the resource utilization and Quality of Service, we investigate the issue of task offloading for End-Edge-Cloud orchestrated computing in mobile networks. Particularly, we jointly optimize the server selection and resource allocation to minimize the weighted sum of the average cost. A cost minimization problem is formulated under joint the constraints of cache resource and communication/computation resource of edge servers. The resultant problem is a Mixed-Integer Non-linear Programming, which is NP-hard. To tackle this problem, we decompose it into simpler subproblems for server selection and resource allocation, respectively. We propose a low-complexity hierarchical heuristic approach to achieve server selection, and a Cauchy-Schwards Inequality based closed-form approach to efficiently determine resource allocation. Finally, simulation results demonstrate the superior performance of the proposed scheme on reducing the weighted sum of the average cost in the network.**

I. Introduction

In the recent years, the proliferation of mobile devices (MDs) and tremendous growth of mobile networking technologies have gradually driven many novel emerging services such as virtual reality (VR) and augmented reality (AR), resulting in continuing growth of nearly 11-fold in the global mobile data traffic over the next 5 years [1]. These services are latency-critical and computation-intensive tasks. The powerful cloud server (CS) is deployed far away from users, resulting in heavy load, core network congestion and high transmission latency in conventional cloud computing [2]. Thus, cloud computing is limited to process latency-critical tasks. Mobile edge computing (MEC) is a novel computing paradigm via pushing computation/storage resource from the CS to the network edges to improve quality of service (QoS) especially on reducing latency. In the context of mobile network, edge servers (ESs) can be deployed at or near base stations (BSs). MEC circumvents the above drawbacks introduced by offloading tasks from the MDs to the remote CS infrastructures [3].

Although various latency-critical and computation-intensive tasks can be processed at the ESs in proximity to the MDs for reducing congestion and service latency, it is not easily realized even by MEC [4]. The main bottleneck lies in that the task offloading is limited by computing, communication and caching capacities of the ESs as compared with cloud computing, and raises the optimization problem of server selection and resource allocation throughout an MEC system [5], i.e., which server to process tasks and how to jointly allocate communication resource, computation resource and cache resource of the ESs, respectively. Thus, task offloading is a crucial issue with MEC.

Several pioneering works have been presented and achieved good results in task offloading under their assuming system and problem settings [6], [7]. There are some important objectives for the collaborative server selection and resource allocation to improve the QoS, e.g., minimizing latency and minimizing energy consumption. Ren *et al.* [8] investigated the collaboration between cloud computing and edge computing to improve their computing efficiency by minimizing the weighted-sum latency of all MDs. But the studies in [7], [8] did not consider the computation capability of the MDs. Huang *et al.* [9] considered a wireless powered MEC network to minimize time consumption, which just jointly utilized a single ES and multiple MDs and neglected the computation resource of the CS. In [10] and [11], the authors investigated to minimize the latency under the energy constraint. However, the studies in [6], [10] did not consider the minimization issue of energy consumption. Sardellitti *et al.* [12] proposed a joint communication and computation resource allocation algorithm to minimize the MDs' energy consumption in the multi-cell MEC system. To jointly minimize latency and energy consumption, the work in [13] considered both of the two objectives as a multi-objective optimization problem, which also neglected the computation resource of the MDs. The above works considered the constraints of communication and computation capacities, but the constraint of the storage capacity of the ESs is neglected. The issue of minimizing latency and energy consumption in the MEC system with more practical considerations is still unexplored well.

In this paper, we are motivated to exploit task offloading for a collaborative End-Edge-Cloud orchestrated computing system. In the considered system, we focus on the joint server selection and resource allocation when processing tasks. Under the constraints of computing, communication and caching

978-1-7281-3106-1/20/$31.00 ©2020 IEEE

Authorized licensed use limited to: CHONGQING UNIVERSITY. Downloaded on March 20,2024 at 07:23:58 UTC from IEEE Xplore. Restrictions apply.

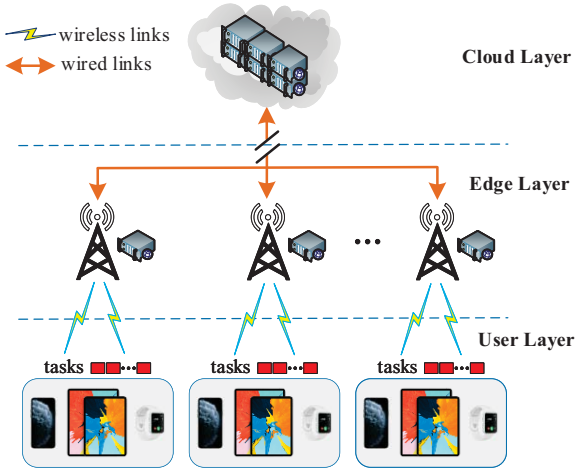

Fig. 1. Topology of hierarchical task offloading in an MEC network.

resources of ESs, the optimization problem aims at minimizing the weighted sum of the average cost consisting of completing tasks latency and MDs' energy consumption. However, the formulated problem is a Mixed-Integer Non-linear Programming (MINLP) problem, which is NP-hard. To tackle this problem, we decompose it into simpler subproblems for searching server selection and determining resource allocation, respectively. We propose a low-complexity hierarchical heuristic approach to achieve server selection, and a Cauchy-Schwards Inequality based closed-form approach to efficiently determine resource allocation. Simulation results demonstrate the effectiveness of the proposed scheme in the considered network.

The remainder of the paper is organized as follows. Section II describes the system model and problem formulation. We investigate a novel collaborative End-Edge-Cloud task offloading approach in Section III. Numerical results are presented in Section IV, followed by concluding remarks in Section V.

II. System Model and Problem Formulation

A. System Model

We consider an MEC network over fiber-wireless connection as illustrated in Fig. 1. The MEC is modeled as a hierarchical computing topology, consisting of cloud layer, edge layer, and user layer. This MEC is an End-Edge-Cloud orchestrated computing system with one centralized CS and several single-antenna BSs. The powerful CS is connected to the BSs through wired optical cables but deployed far away. Each BS is deployed with an ES that has limited capacities of computing and caching. Each ES can serve multiple MDs simultaneously via wireless cellular links. The ESs can be regarded as an intermediary between the MDs and the CS. Compared with conventional cloud computing, some of the tasks in the MDs could be offloaded to the closer ESs rather than the remote CS, thereby reducing traffic congestion in wired optical cables and service latency.

In this paper, we consider the case of the MEC with one CS, a single BS, and M MDs (denoted the set as $\mathcal{M} = \{1, 2, ..., M\}$). The i-th MD is assumed to generate only one new computation task T_i in a considered time period. Denote the set of all computation tasks as $\mathcal{T} = \{T_1, T_2, ..., T_M\}$. We assume that T_i, $\forall i \in \mathcal{M}$ is homogeneous. We consider that the ES makes the offloading strategies of $\mathcal{T}$ based on the collected task information $\{I_i\}$ and the network environment condition in the considered time period. We assume that the measurement of the computation load of T_i is based on the number of CPU cycles. Particularly, the information I_i of T_i includes the data size D_i, the task priority W_i and the total number of CPU cycles C_i to complete T_i. Especially, different tasks have different priority values. For example, the ES is serving two MDs. One user wants to buy a commodity through a mobile phone which generates a mobile payment task. Simultaneously, another user is watching Youtube via iPad, which generates a video requesting task. Under such circumstances, the payment task is more important than the video requesting. As the information size is very small and the MDs are close to the ES, we assume that the latency and energy consumption of information exchanges can be ignored.

As the considered MEC is an End-Edge-Cloud orchestrated computing system, $\mathcal{T}$ can be processed locally at the MDs, or offloaded to the ES, or transmitted to the CS. We assume that each task is indivisible. In other words, the entire task must be executed at one of the three layers. We model the server selection of each task T_i as three sub-decisions, denoted as $\boldsymbol{\pi}_i = (\pi_i^L, \pi_i^E, \pi_i^C)$, where $\pi_i^L, \pi_i^E, \pi_i^C \in \{0, 1\}$ are the indicators for whether T_i is processed in the MD, the ES, or the CS, respectively. Three offloading sub-decisions can jointly determine where T_i is processed, and affect the corresponding latency and energy consumption. The server selection $\boldsymbol{\pi}_i$ is constrained by

$$\pi_i^L + \pi_i^E + \pi_i^C = 1, \quad \forall i \in \mathcal{M}. \tag{1}$$

Note that only one of π_i^L, π_i^E and π_i^C can be 1 for T_i. In the following, we will discuss the task transmission model and task computation model in detail.

1) Task Transmission Model: In the considered MEC, data (e.g. computation tasks, environment information, and offloading strategies) is transmitted between the MDs and the ES over wireless links, the ES and the CS transmit data though wired optical cables. Then we model the time consumption and energy consumption of task transmissions. Specifically, we assume that the channel state can be estimated. We denote the wireless channel gain from the i-th MD to the ES as h_i, and the transmit power of the i-th MD as P_i. The transmission rate between the i-th MD and the ES is calculated as

$$r_i = b_i^E \log_2(1 + \frac{P_i |h_i|^2}{\sigma_N^2}), \tag{2}$$

where σ_N^2 is the noise power, b_i^E is the allocated radio bandwidth resource for task T_i from i-th MD to the ES. However, the entire radio bandwidth resource B^E of the MDs is limited, i.e., $\sum_{i \in \mathcal{M}} b_i^E \leq B^E$.

Then the transmission latency of uploading task T_i to the

Authorized licensed use limited to: CHONGQING UNIVERSITY. Downloaded on March 20,2024 at 07:23:58 UTC from IEEE Xplore. Restrictions apply.

ES via wireless links is denoted as

$$t_i^{(LE)} = \frac{D_i}{r_i}. \tag{3}$$

In addition to transmission time consumption, uploading tasks from the MDs to the ES will generate energy consumption (i.e. battery energy loss at the MDs) which is as

$$e_i^{(LE)} = P_i t_i^{(LE)}. \tag{4}$$

In terms of the communication between the ES and the CS, they are connected through wired optical cables with high bandwidth. However, the powerful CS is usually deployed far away from the MDs, and massive traffic is usually transmitted through multiple intermediate nodes. Under such a circumstance, we assume that the transmission latency of offloading task to the CS primarily affects the QoS. Thus, we ignore the energy consumption of the task transmission from the ES to the CS. The transmission time $t_i^{(EC)}$ of T_i from the ES to the CS can be expressed as

$$t_i^{(EC)} = \frac{D_i}{R^C}, \tag{5}$$

where R^C denotes the average transmission rate of each task from the ES to the CS [8].

2) Task Computation Model: According to server selection, T_i can be executed at the i-th MD, the ES, or the CS. When T_i is executed locally at the i-th MD, we denote its average computing speed as f_i^L. The computation time consumption t_i^L for completing T_i locally is represented as

$$t_i^L = \frac{C_i}{f_i^L}. \tag{6}$$

The local energy consumption e_i^L to complete T_i at the i-th MD is calculated as

$$e_i^L = \kappa (f_i^L)^2 C_i, \tag{7}$$

where $\kappa = 10^{-26}$ is the energy consumption coefficient associated with the chip architecture [14].

When T_i is executed at the ES, we denote the computation time consumption at the ES as t_i^E, which is expressed as

$$t_i^E = \frac{C_i}{f_i^E}, \tag{8}$$

where f_i^E is the allocated computing rate for T_i at the ES, which is constrained by the total computation capability of the ES as $\sum_{i\in\mathcal{M}} f_i^E \le F^E$.

If T_i is offloaded to the CS, completing T_i also generates time delay. The corresponding the computation time consumption t_i^C at the CS can be calculated as

$$t_i^C = \frac{C_i}{F^C}, \tag{9}$$

where F^C is the average computing speed for each task at the CS.

B. Problem Formulation

In the considered MEC, we focus on how to improve the QoS. The latency and energy consumption of processing tasks are two important factors. Thus, our goal is to minimize the system service latency t_{sys} and minimize the system energy consumption e_{sys} by jointly optimizing the server selection $\boldsymbol{\pi}_i$ and the resource allocation $\boldsymbol{\gamma}_i$=(f_i^E, b_i^E), expressed as

$$t_{sys} = \min_{\{\boldsymbol{\pi}_i, \boldsymbol{\gamma}_i\}} \sum_{i\in\mathcal{M}} t_i, \tag{10}$$

$$e_{sys} = \min_{\{\boldsymbol{\pi}_i, \boldsymbol{\gamma}_i\}} \sum_{i\in\mathcal{M}} e_i, \tag{11}$$

where $t_i = t_i^L \pi_i^L + (t_i^E + t_i^{(LE)})\pi_i^E + (t_i^C + t_i^{(LE)} + t_i^{(EC)})\pi_i^C$ is the total latency of completing T_i, $e_i = e_i^L \pi_i^L + e_i^{(LE)}(\pi_i^E + \pi_i^C)$ is the total energy consumption of the i-th MD. However, in such a multi-objective optimization problem, two optimization goals are usually coupled, it is impossible for each goal to achieve optimality simultaneously.

As mentioned earlier, different tasks have different priorities. The higher priority of T_i indicates that the delay of completing T_i should be lower for improving the QoS. Meanwhile, there is a non-ignorable constraint that the battery of the MDs is not powerful. When the i-th MD has low battery, we mainly consider energy consumption. Motivated by this idea, we intend to use linear scalarization to reformulate this multi-objective optimization. The goal is converted to minimize the weighted sum of the average cost (WSC), consisting of completing tasks latency and MDs' energy consumption. The corresponding optimization problem can be formulated as

$$\mathcal{P}: \min_{\{\boldsymbol{\pi}_i, \boldsymbol{\gamma}_i\}} \sum_{i\in\mathcal{M}} \left[\frac{\beta_i t_i}{\bar{t}} + \frac{(1-\beta_i)e_i}{\bar{e}}\right] \tag{12a}$$

$$s.t. \quad \pi_i^L + \pi_i^E + \pi_i^C = 1, \tag{12b}$$

$$\forall \pi_i^L \in \{0,1\}, \forall \pi_i^E \in \{0,1\}, \forall \pi_i^C \in \{0,1\}, \tag{12c}$$

$$\sum_{i\in\mathcal{M}} D_i \pi_i^E \le D^E, \sum_{i\in\mathcal{M}} b_i^E \le B^E, \sum_{i\in\mathcal{M}} f_i^E \le F^E, \tag{12d}$$

$$\forall b_i^E \in [0, B^E], \forall f_i^E \in [0, F^E], \tag{12e}$$

where $\bar{t} = \max_{i\in\mathcal{M}}(\frac{C_i}{f_i^L})$ and $\bar{e} = \max_{i\in\mathcal{M}}(\kappa (f_i^L)^2 C_i)$ are for normalization, D^E is the maximum caching capability of the ES, $\{\pi_i^L, \pi_i^E, \pi_i^C\}$ are 0-1 discrete variables, $\{f_i^E, b_i^E\}$ are continuous variables, the weight β_i of the objectives is the parameter of the scalarization. The β_i is to balance service latency and energy consumption. We model β_i as a function related to task priority $W_i \in [0,1]$ and percentage of the MDs' remaining battery $E_i^{RB} \in [0,1]$ as

$$\beta_i = \theta W_i + (1-\theta) E_i^{RB}, \quad \beta_i \in [0,1], \tag{13}$$

where $\theta \in [0,1]$ is a parameter depending on the QoS.

Authorized licensed use limited to: CHONGQING UNIVERSITY. Downloaded on March 20,2024 at 07:23:58 UTC from IEEE Xplore. Restrictions apply.

The optimization objective of the problem in $(12a)$ can be rewritten as

$$\mathcal{P}: \min_{\{\boldsymbol{\pi}_i,\boldsymbol{\gamma}_i\}} \sum_{i\in\mathcal{M}} [A_{1,i}\pi_i^L + (\frac{A_{2,i}}{f_i^E} + \frac{A_{3,i}}{b_i^E})\pi_i^E + (A_{4,i} + \frac{A_{3,i}}{b_i^E})\pi_i^C] \tag{14a}$$

$$\textit{s.t.}\ \text{the same with (12b), (12c), (12d), and (12e),} \tag{14b}$$

where $A_{1,i} = (\frac{\beta_i}{\bar{t}f_i^L} + \frac{(1-\beta_i)\kappa(f_i^L)^2}{\bar{e}})C_i$, $A_{2,i} = \frac{\beta_i C_i}{\bar{t}}$, $A_{3,i} = (\frac{\beta_i}{\bar{t}} + \frac{(1-\beta_i)P_i}{\bar{e}})\frac{D_i}{\log_2(1+\frac{P_i|h_i|^2}{\sigma_N^2})}$, and $A_{4,i} = \frac{\beta_i}{\bar{t}}(\frac{C_i}{F^C} + \frac{D_i}{R^C})$ are positive constants.

The joint End-Edge-Cloud server selection and resource allocation optimization problem in (14) is an MINLP problem, which is NP-hard. Considering that the number of MDs is large, it is of high complexity to get the optimal solution by using exact methods.

III. Design of Task Offloading Framework

The considered system is the formed end-edge-cloud orchestrated computing topology, and task offloading consists of server selection and resource allocation. Therefore, to address the above complex optimization problem in (14), we decompose it into simpler subproblems as below.

1) Server selection subproblem $\mathcal{P}_a$: To find an optimal or a suboptimal server selection $\boldsymbol{\pi}_i$, i$\in\mathcal{M}$, the optimization problem $\mathcal{P}$ is a 0-1 knapsack problem as

$$\mathcal{P}_a: \min_{\boldsymbol{\pi}_i} \sum_{i=1}^{M} (A_{1,i}, (\frac{A_{2,i}}{f_i^E} + \frac{A_{3,i}}{b_i^E}), (A_{4,i} + \frac{A_{3,i}}{b_i^E}))\boldsymbol{\pi}_i^\top \tag{15a}$$

$$\textit{s.t.}\ \text{the same with (12b) and (12c).} \tag{15b}$$

Denote $\boldsymbol{\Pi}$=$(\boldsymbol{\Pi}^L,\boldsymbol{\Pi}^E,\boldsymbol{\Pi}^C)$=$(\boldsymbol{\pi}_1,\boldsymbol{\pi}_2,...,\boldsymbol{\pi}_M)^\top$ as entire server selection, where $\boldsymbol{\Pi}^L$=$(\pi_1^L,\pi_2^L,...,\pi_M^L)^\top$, $\boldsymbol{\Pi}^E$=$(\pi_1^E,\pi_2^E,...,\pi_M^E)^\top$, $\boldsymbol{\Pi}^C$=$(\pi_1^C,\pi_2^C,...,\pi_M^C)^\top$. $\boldsymbol{\Pi}$ has 3^M possible selections. Thus, the search algorithms will take a lot of time to converge due to the large search space. It becomes crucial to find a feasible solution for determining where all tasks are offloaded.

2) Resource allocation subproblem $\mathcal{P}_b$: As the server selection $\boldsymbol{\Pi}$ is determined, the optimization problem $\mathcal{P}$ becomes a nonlinear optimization problem as

$$\mathcal{P}_b: \min_{\boldsymbol{\gamma}_i} \sum_{i=1}^{M} [(A_{1,i}\pi_i^L + A_{4,i}\pi_i^C) + \frac{A_{2,i}\pi_i^E}{f_i^E} + \frac{A_{3,i}(\pi_i^E+\pi_i^C)}{b_i^E}] \tag{16a}$$

$$\textit{s.t.}\ \text{the same with (12d) and (12e),} \tag{16b}$$

where all except f_i^E and b_i^E are constants. Denote $\boldsymbol{\Gamma} = (\boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2, ..., \boldsymbol{\gamma}_M)^\top$.

Based on the hierarchical task offloading topology, we consider using a hierarchical idea to solve the problem $\mathcal{P}_a$. Some tasks with the small data size and the small number of required CPU cycles are processed locally, which will take less time and energy consumption than offloading tasks to the ES or the CS. Since the computing, communication and caching capacities of the ES are limited, when the number of tasks offloaded from the MDs to the ES is too large, it will increase the delay and energy consumption of processing tasks. Thus, it needs to search optimal server selection between the MDs, the ES, and the CS via iterative approach. Each iteration will determine a new $\boldsymbol{\Pi}$, then we need to calculate the minimum value (i.e. finding the best $\boldsymbol{\Gamma}$) of the objective function (16) under the constraints (12d) and (12e).

From the perspective of practical engineering implementations, we propose an iterative approach to optimize the server selection and resource allocation in an alternating way. First, we search server selection $\boldsymbol{\Pi}$ by layer by layer. Then a closed-form solution $\boldsymbol{\Gamma}$ is determined by the Cauchy-Schwards Inequality. Finally, iterate server selection and resource allocation until satisfying the stopping condition. The main procedures of the proposed approach are shown as follows.

Procedure 1: After initialization, we first exploit the local server selection $\boldsymbol{\Pi}^L = \mathbf{1}$ that all tasks are processed at the MDs. Then, K tasks are chosen to be offloaded from the MDs to the ES. To determine the offloading sub-decision between the user layer and the edge layer, the difference of the weighted sum of the average cost between the MDs and the ES is calculated as

$$D_{WSC} = \beta_i(t_i^L - t_i^E - t_i^{(LE)}) + (1-\beta_i)(e_i^L - e_i^{(LE)}) \tag{17a}$$

$$= \beta_i C_i(\frac{1}{f_i^L} - \frac{1}{f_i^E} - \frac{\varepsilon_i}{r_i}) + (1-\beta_i)C_i[\kappa(f_i^L)^2 - \frac{P_i\varepsilon_i}{r_i}]. \tag{17b}$$

When the quality value of $P_i|h_i|^2$ is larger, tasks with the bigger data size will take more computation resource at the MDs. Therefore, we offload the task with bigger $(C_iP_i|h_i|^2)$ from the MDs to the ES, and the edge offloading action π_i^E=1. All tasks at the ES should satisfy the constraint (12d) and (12e). If the total data size of all tasks at the ES is larger than the caching capacity of the ES, we offload some tasks with smaller D_i from the ES to the remote CS. The $\boldsymbol{\Pi}^E$ can be determined and $\boldsymbol{\Pi}^C = \mathbf{1} - \boldsymbol{\Pi}^L - \boldsymbol{\Pi}^E$. Through the above heuristic offloading strategy, the server selection $\boldsymbol{\Pi}$ can be determined as constants.

Procedure 2: To solve the convex nonlinear optimal problem $\mathcal{P}_b$, we propose a Cauchy-Schwards Inequality based closed-form approach to efficiently determine resource allocation $\boldsymbol{\Gamma}$. Applying Cauchy-Schwards Inequality and the constraint (12e), we have $F^E\sum_{i=1}^M(\frac{A_{2,i}\pi_i^E}{f_i^E}) \geq \sum_{i=1}^M f_i^E \sum_{i=1}^M(\frac{A_{2,i}\pi_i^E}{f_i^E}) \geq (\sum_{i=1}^M\sqrt{A_{2,i}\pi_i^E})^2$ and $B^E\sum_{i=1}^M(\frac{A_{3,i}(\pi_i^E+\pi_i^C)}{b_i^E}) \geq \sum_{i=1}^M b_i^E\sum_{i=1}^M(\frac{A_{3,i}(\pi_i^E+\pi_i^C)}{b_i^E}) \geq (\sum_{i=1}^M\sqrt{A_{3,i}(\pi_i^E+\pi_i^C)})^2$. To make two inequalities become equal, we can get the closed expression as

$$f_i^{E*} = \frac{F^E\sqrt{A_{2,i}\pi_i^E}}{\sum_{i=1}^M\sqrt{A_{2,i}\pi_i^E}}, \tag{18}$$

$$b_i^{E*} = \frac{B^E\sqrt{A_{3,i}(\pi_i^E+\pi_i^C)}}{\sum_{i=1}^M\sqrt{A_{3,i}(\pi_i^E+\pi_i^C)}}. \tag{19}$$

Authorized licensed use limited to: CHONGQING UNIVERSITY. Downloaded on March 20,2024 at 07:23:58 UTC from IEEE Xplore. Restrictions apply.

The details of the proposed hierarchical heuristic approach for solving the whole problem are shown in Algorithm 1. The computation complexity of Algorithm 1 is $O(NM \log M)$.

Algorithm 1 Hierarchical heuristic method for task offloading.

Input: $\{I_1, I_2, ..., I_M\}$, maximum number of iterations N, number of tasks offloading K.
Output: $\{\mathbf{\Pi}, \mathbf{\Gamma}\}$.

Initialize $\mathbf{\Pi} = 0, \mathbf{\Gamma} = 0$.
$\mathbf{\Pi}^L = \mathbf{1}$.
for j = 1 to N **do**
  —Procedure 1. [Determine $\mathbf{\Pi}$.]
  Sort local tasks by $(C_i P_i |h_i|^2)$ in a descending order, choose the first K tasks to offload.
  if Task is offloaded to the ES **then**
    $\pi^L = 0$,
  else
    $\pi^E = 1$.
  end if
  if $\sum_{i=1}^{M} D_i \pi_i^E > D^E$ **then**
    Sort $D_s,\ s \in S = \{s | \pi_s^E = 1\}$ in an ascending order.
    repeat
      $\pi_s^E = 0$.
      $s \leftarrow s + 1$.
    until $\sum_{s \in S} D_s \pi_s^E \leq D^E$.
  end if
  $\mathbf{\Pi}^C = \mathbf{1} - \mathbf{\Pi}^L - \mathbf{\Pi}^E$.
  Update $\mathbf{\Pi}_j \leftarrow \mathbf{\Pi}$.
  —Procedure 2. [Determine $\mathbf{\Gamma}$.]
  Solve $\mathcal{P}_b$ via the Cauchy-Schwards Inequality based closed-form approach under $\mathbf{\Pi}_j$.
  $\gamma_i \leftarrow (f_i^{E*}, b_i^{E*}),\ \forall i \in \mathcal{M}$.
  Store $\{\mathbf{\Pi}, \mathbf{\Gamma}\}_{best}$ and cost WSC_j.
  if $WSC_j < WSC_{min}$ **then**
    Update $\{\mathbf{\Pi}, \mathbf{\Gamma}\}_{best} \leftarrow \{\mathbf{\Pi}, \mathbf{\Gamma}\}_j$.
    Update $WSC_{best} \leftarrow WSC_j$.
  end if
end for
return $\{\mathbf{\Pi}, \mathbf{\Gamma}\}_{best}$.

IV. Simulation Results

Numerical experiments are provided to evaluate the performance of the proposed approach. For simulation purpose, all parameters are selected according to real-word scenario. Specifically, we consider that the MEC network covers a circular with a radius area of 1.5 km, 100 MDs are randomly and uniformly distributed in this region. We consider large-scale fading and small-scale fading in the wireless channel. Particularly, the pass loss (dB) is 36.8+36.7log(d), where d is the distance in meter; the log-normal shadowing parameter is 7 dB and antenna gain is 5 dBi; the small-scale fading is Raylergh fading with unit variance [15]. The wireless bandwidth, transmit power of each MD and noise power is set as 20 Mhz, [1.0, 1.5] W, and 10^{-13} W, respectively. The average transmission rate of each task from the ES to the CS, computing capability of the ES and average computing resource of the CS for each task is set as 3 Mbps, 3×10^9 Cycles/sec, and 1×10^9 Cycles/sec, respectively [8]. The remaining battery E_i^{RB} of the i-th MD is set to range in (0, 1]. The MDs' CPU computing capacities are random in [300, 600] Cycle/sec [16]. The data size D_i is randomly set from [2, 5] Mbits and the total number of CPU cycles $C_i = D_i \varepsilon_i$, where ε_i is the computation intensity set to be 300 Cycles/bit. The task priority W_i is uniformly distributed in the range of (0, 1]. Further, the cache constraint of the ES is set to a percentage α of the total data size, and set the number of tasks offloading K as 1.

We compare the performance of the proposed approach with the following alternatives: 1) Random approach: the tasks are randomly offloaded at local layer, edge layer, or cloud layer. $\mathbf{\Pi}$ is randomly set under the constraint (12b) and (12c). $\mathbf{\Gamma}$ is also randomly allocated under the constraint (12d) and (12e); and 2) Average approach: where each task is processed by average idea, which means that server selection $\mathbf{\Pi}$ and resource allocation $\mathbf{\Gamma}$ are set on average.

First, we illustrate the convergence behavior of the hierarchical heuristic approach based on iteration as shown in Fig. 2. We study three different resource allocation approaches under θ=0.5. For comparison, we set α to 0.1 and 0.2 respectively, so that six independent simulations are considered. It can be seen that the proposed approach has very low the weighted sum of the average cost, especially α=0.1. When the number of iteration N=24, the proposed approach with α=0.1 begins to converge, and the result with α=0.2 has the lower convergence value. In Fig. 3, we study different server selection approaches under M range of 25 to 35. Similarly, six independent simulations are considered. We observe that the result of our proposed approach with α=0.1 is best. At different numbers of the MDs, the weighted sum of the average cost of our approach is lower.

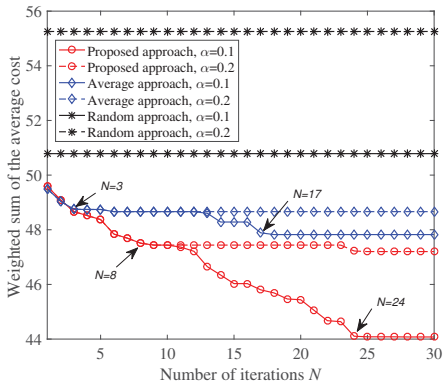

Fig. 2. Convergence behavior of the hierarchical heuristic approach.

In Fig. 4, we further investigate the impact of various parameters on the performance of three different approaches in terms of the weighted sum of the average cost. We set by default the number of iteration N=30. The cache con-

Authorized licensed use limited to: CHONGQING UNIVERSITY. Downloaded on March 20,2024 at 07:23:58 UTC from IEEE Xplore. Restrictions apply.

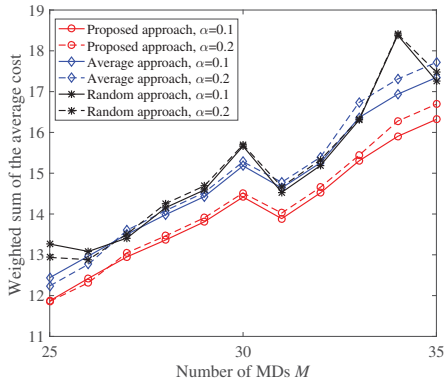

Fig. 3. Weighted sum of the average cost versus the number of MDs.

straint ratio α is the range of 0 to 1, and the balance ratio $\theta \in [0, 1]$ related to task priority and the MDs' remaining battery percentage is studied. We observe that the proposed approach and the average approach with α=0.05 can achieve the minimum. Because fewer tasks are offloaded to the ES, which will be allocated more communication/computation resources. Different weight values θ have less impact on the optimization problem under the proposed approach and the average approach, but the result of the random approach fluctuates greatly at different ratios. Because $\mathbf{\Gamma}$ randomly allocated effects that the result is fluctuating. All of the above results demonstrate the superior performance of the proposed approach on reducing the weighted sum of the average cost.

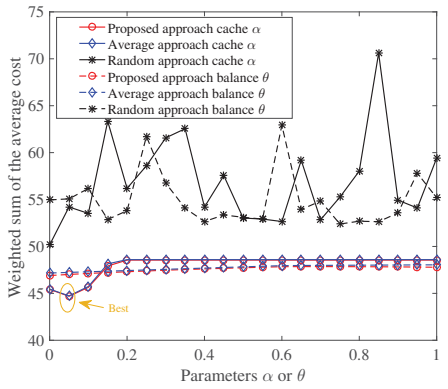

Fig. 4. Weighted sum of the average cost versus different α and θ.

V. Conclusion

In this paper, we have investigated the issue of joint server selection and resource allocation for task offloading in the collaborative End-Edge-Cloud orchestrated computing system. A weighted sum of the average cost minimization problem under the constraints of communication, computation, and cache resources has been formulated as an MINLP. To tackle the complex optimization problem, we have decomposed it into two subproblems. The first one is associated with server selection, and we have proposed a low-complexity hierarchical heuristic approach to solve it. The other one is associated with resource allocation, and its optimal solution can be solved by the proposed Cauchy-Schwards Inequality based closed-form approach. Simulation results have revealed the superior performance of the proposed scheme on reducing the weighted sum of the average cost in the considered network.

Acknowledgment

This work is supported in part by National Key R & D Program of China (Grants No. 2018YFF0214700 and 2018YFF0214706), National NSFC (Grants No. 61902044 and 61672117), Chongqing Research Program of Basic Research and Frontier Technology (Grant No. cstc2019jcyj-msxmX0589), Key Research Program of Chongqing (Grant No.CSTC2017jcyjBX0025), and Major Science & Technology Program of Guangxi (Grant No.GKAA17129002).

References

[1] A. Tongaonkar, "A look at the mobile app identification landscape," *IEEE Internet Computing*, vol. 20, no. 99, pp. 1–1, Aug. 2018.
[2] B. Cao, L. Zhang, Y. Li, D. Feng, and W. Cao, "Intelligent offloading in multi-access edge computing: A state-of-the-art review and framework," *IEEE Commun. Mag.*, vol. 57, no. 3, pp. 56–62, Mar. 2019.
[3] X. Wang, Y. Han, C. Wang, Q. Zhao, X. Chen, and M. Chen, "In-edge ai: Intelligentizing mobile edge computing, caching and communication by federated learning," *IEEE Network*, pp. 1–10, Jul. 2019.
[4] Y. Han, X. Wang, V. C. M. Leung, D. Niyato, X. Yan, and X. Chen, "Convergence of edge computing and deep learning: A comprehensive survey," *arXiv preprint arXiv:1907.08349*, Jul. 2019.
[5] P. Mach and Z. Becvar, "Mobile edge computing: A survey on architecture and computation offloading," *IEEE Commun. Surveys & Tutorials*, vol. 19, no. 3, pp. 1628–1656, Mar. 2017.
[6] M. Chen and Y. Hao, "Task offloading for mobile edge computing in software defined ultra-dense network," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 3, pp. 587–597, Mar. 2018.
[7] X. Li, X. Wang, and V. C. M. Leung, "Weighted network traffic offloading in cache-enabled heterogeneous networks," in *in Proc. IEEE ICC*. IEEE, May. 2016, pp. 1–6.
[8] J. Ren, G. Yu, Y. He, and G. Y. Li, "Collaborative cloud and edge computing for latency minimization," *IEEE Trans. Vehi. Tech.*, vol. 68, no. 5, pp. 5031–5044, Mar. 2019.
[9] L. Huang, S. Bi, and Y. J. Zhang, "Deep reinforcement learning for online computation offloading in wireless powered mobile-edge computing networks," *IEEE Trans. Mobile Comput.*, pp. 1–1, Jul. 2019.
[10] J. Ren, G. Yu, Y. Cai, and Y. He, "Latency optimization for resource allocation in mobile-edge computation offloading," *IEEE Trans. Wireless Commun.*, vol. 17, no. 8, pp. 5506–5519, Jun. 2018.
[11] A. Yousefpour, G. Ishigaki, R. Gour, and J. P. Jue, "On reducing iot service delay via fog offloading," *IEEE Internet of Things J.*, vol. 5, no. 2, pp. 998–1010, Jan. 2018.
[12] S. Sardellitti, G. Scutari, and S. Barbarossa, "Joint optimization of radio and computational resources for multicell mobile-edge computing," *IEEE Trans. Signal & Inf. Process. over Networks*, vol. 1, no. 2, pp. 89–103, Jun. 2015.
[13] N. Eshraghi and B. Liang, "Joint offloading decision and resource allocation with uncertain task computing requirement," in *in Proc. IEEE INFOCOM*, Jun. 2019, pp. 1414–1422.
[14] Y. Liu, M. J. Lee, and Y. Zheng, "Adaptive multi-resource allocation for cloudlet-based mobile cloud computing system," *IEEE Trans. Mobile Computing*, vol. 15, no. 10, pp. 2398–2410, Nov. 2015.
[15] X. Wu, X. Li, Q. Li, V. C. M. Leung, and P. Ching, "Latency driven fronthaul bandwidth allocation and cooperative beamforming for cache-enabled cloud-based small cell networks," in *in Proc. IEEE ICASSP*, Apr. 2019, pp. 4594–4598.
[16] J. Chen, S. Chen, Q. Wang, B. Cao, G. Feng, and J. Hu, "iraf: a deep reinforcement learning approach for collaborative mobile edge computing iot networks," *IEEE Internet of Things J.*, pp. 7011–7024, Apr. 2019.

Authorized licensed use limited to: CHONGQING UNIVERSITY. Downloaded on March 20,2024 at 07:23:58 UTC from IEEE Xplore. Restrictions apply.