Energy-Efficient Task Offloading Under E2E Latency Constraints

Mohsen Tajallifar, Sina Ebrahimi, Mohammad Reza Javan, Nader Mokari, and Luca Chiaraviglio,

Abstract

In this paper, we propose a novel resource management scheme that jointly allocates the transmit power and computational resources in a centralized radio access network architecture. The network comprises a set of computing nodes to which the requested tasks of different users are offloaded. The optimization problem minimizes the energy consumption of task offloading while takes the end-to-end-latency, i.e., the transmission, execution, and propagation latencies of each task, into account. We aim to allocate the transmit power and computational resources such that the maximum acceptable latency of each task is satisfied. Since the optimization problem is non-convex, we divide it into two sub-problems, one for transmit power allocation and another for task placement and computational resource allocation. Transmit power is allocated via the convex-concave procedure. In addition, a heuristic algorithm is proposed to jointly manage computational resources and task placement. We also propose a feasibility analysis that finds a feasible subset of tasks. Furthermore, a disjoint method that separately allocates the transmit power and the computational resources is proposed as the baseline of comparison. A lower bound on the optimal solution of the optimization problem is also derived based on exhaustive search over task placement decisions and utilizing Karush–Kuhn–Tucker conditions. Simulation results show that the joint method outperforms the disjoint method in terms of acceptance ratio. Simulations also show that the optimality gap of the joint method is less than 5%.

Index Terms

Mobile edge computing, task offloading, resource allocation, end-to-end latency, task placement.

I. Introduction

A. Background

In order to fulfill the requirements of 5G mobile networks, key enabling technologies such as network function virtualization (NFV) and multi-access/mobile edge computing (MEC) are

M. Tajallifar, S. Ebrahimi, and N. Mokari are with the Department of Electrical and Computer Engineering, Tarbiat Modares University, Tehran, 14115-111 Iran e-mail: nader.mokari@modares.ac.ir. M. Javan is with Shahrood University of Technology. L. Chiaraviglio is with University of Rome Tor Vergata.

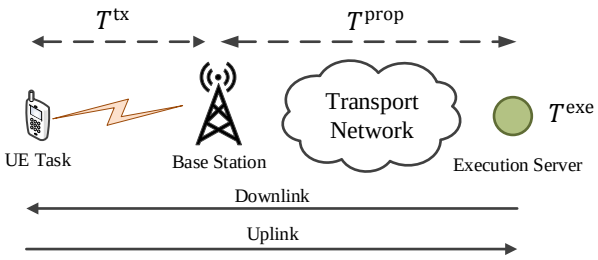


Fig. 1: A typical task offloading example.

introduced. With NFV, the network functions (NFs) that traditionally used dedicated hardware are implemented in applications running on top of commodity servers [1]. On the other hand, MEC aims to support low-latency mobile services by bringing the remote servers closer to the mobile users [2], [3]. Moreover, MEC enables the offloading of the computational burden of users' tasks to reduce the impact of the limited battery power of user equipment (UE). Note that when executing servers are NFV-enabled, they are able to process various types of tasks. As a result, there is no restriction on offloading a task to a predetermined server.

A typical task offloading example is shown in Fig. 1. In task offloading, the non-processed data of a task is sent from UE to an executing server that offloads the computational burden of the task execution on the executing server. As Fig. 1 shows, the user transmits the non-processed data of the task over the wireless link to its serving base station, which results in transmit latency T^{tx}. Then, the received data is transmitted to an executing server. Executing servers are placed at the base station and distant nodes in the transport network. The data transmission through the transport network adds the propagation latency T^{prop} to the offloading process. Finally, the received data is processed at the executing server with execution latency T^{exe} and then is sent back to the user over the downlink. Therefore, the end-to-end (E2E) latency of task offloading is equal to the summation of T^{tx}, T^{prop}, and T^{exe} in both uplink and downlink.

B. Related Works

We classify the related works on task offloading into four categories and discuss their applicability in practical scenarios.

1) Task offloading to multiple executing servers: In task offloading, a UE decides to whether offload a task to a single executing server or to select an executing server out of multiple servers. Offloading a task to one server in a set of executing servers in a multi-tier heterogeneous network

is considered in [4], [5]. Moreover, the authors in [6]–[8] propose to offload a user's task to one of executing servers at base stations in a multi-cell network. Note that in the aforementioned works, the executing servers are located at the edge of the radio access network and the computational resources in the non-radio part of the network are not considered. In contrast, it is possible to offload a task to any server in the network in [9], i.e., servers in radio access and non-radio parts of the network. However, radio resources are not allocated in [9]. Note that, ignoring computational resources in the non-radio part of the network or ignoring radio resource allocation results in an inefficient task offloading.

2) Task placement and computational resource allocation: Task offloading is comprised of two steps: i) task placement to select an executing server, and ii) computational resource allocation that allocates the resources of the executing server to each task. In this context, various works only focus on task placement with given computational resources for each task [4]–[6], [9]–[12], while others include resource allocation as well [7], [13]–[21]. Note that the servers in non-radio part of the network are not involved in these works. As a result, computationally intensive tasks with moderate sensitivity to latency may occupy the capacity of executing servers in radio part of the network while high capacity servers in non-radio part of the network are underutilized.

3) Joint Radio and Computational Resource Allocation: Extensive research is made on joint radio and computational resource allocation [7], [11], [13]–[27]. In these works, radio resources including transmit power and/or bandwidth as well as computational resources are allocated to each task. Energy-efficient resource allocation is performed in [11], [13], [17], [19], [21], [24], [26], [27], and a weighted combination of consumed energy and latency is optimized in [7], [14]–[16], [20], [22], [25]. Moreover, the impact of radio link quality without radio resource allocation is taken into account by [5], [6], [8], [10], [28], [29]. In these works, the latency of data transmission over radio links is taken into account, which impacts the optimal task placement. Note that although joint optimization of radio and computational resources increases the degrees of freedom in task offloading, the available computational resources in the radio access network are very limited, which limit the acceptance ratio of the network.

4) Feasibility Analysis: When task offloading is subjected to a maximum acceptable latency, sufficient resources are required in various parts of the network. In case of insufficient resources, a feasibility analysis is needed to determine a feasible subset of requested tasks. One approach to face infeasibility is making some simplifying assumptions, e.g., assuming sufficient available resources for task offloading [9], [22] or offloading a task when it is beneficial, i.e., when

offloading results in less energy consumption or latency [10]. In practice, however, the resources are limited and tasks are subjected to execution deadlines. As a result, a feasibility analysis is inevitable. The feasibility analysis is performed by introducing a binary optimization variable, which is one when the task is accepted or zero when the task is rejected [4], [7], [12], [13], [15], [19]–[21], [27]. Note that finding optimal binary variables results in combinatorial optimization problems that are challenging and of high complexity.

C. Motivation

The performance of a task offloading method is mainly measured by its latency and energy consumption. In practice, E2E latency comes from radio links, transport network links, and execution at the servers; and the energy consumption is impacted by consumed transmit power and computational resources.

Optimizing the performance of task offloading necessitates a joint optimization of all available resources in the network. However, existing works optimize a subset of resources and focus only on one part of the whole network. Moreover, the impact of E2E latency is not considered in the literature. As a result, existing methods may not perform well in practice.

In this paper, we propose a task offloading method that optimizes the energy consumption in terms of transmit power and computational resources under E2E latency constraints. Throughout the paper, the task offloading is referred to the process of transmit power allocation over radio links, task placement, i.e., selecting an executing server and its path, and computational resource allocation. The proposed method jointly allocates required transmit power to tasks, places each task in a proper NFV-enabled node, and allocates sufficient computational resources to the tasks. With this joint method, high latency of radio links caused by weak radio channels is compensated by a proper task placement and computational resource allocation. In contrast, high execution latency caused by limited computational resources is compensated by consuming more transmit power in radio links. As a result, more tasks are served, compared to a disjoint method wherein transmit power allocation is independent of task placement and computational resource allocation.

NFV enables a general-purpose server to execute various tasks without needing a specialized server for each task. Therefore, various tasks are dynamically offloaded to general-purpose executing servers in a network of NFV-enabled nodes instead of offloading each task to a respective specialized server. As a result, a task placement method is needed to determine an executing server and its route for each task. In spite of conventional routing methods that choose

a route to a predetermined server, our task placement method jointly determines an executing server, the associated route to the executing server, and the required computational resources in the executing server.

We assume a deadline for offloading each task, i.e., sending the task from UE to the executing server and sending it back to UE performed under a maximum acceptable latency constraint. As a result, the sum of latencies in radio link, transport network links, and execution at the executing server is less than the maximum acceptable latency. The feasibility of this E2E offloading method depends on the available resources and location of executing servers in the network. For example, when the available transmit power is low, the radio link latency is large, which may violate E2E latency. In contrast, when the available computational resources at the executing server are low, the execution latency is large, which may also violate the E2E latency constraint. Therefore, our task offloading method includes a feasibility analysis that finds a set of feasible tasks.

The infeasibility of task offloading depends on the value of maximum acceptable latencies, i.e., lower values of maximum acceptable latencies result in a larger number of infeasible tasks and higher values result in a smaller number of infeasible tasks. Inspired by this fact and in contrast to the existing works, we add a non-negative variable to each maximum acceptable latency. Non-negative variables are zero for feasible tasks and are positive for infeasible tasks. Therefore, the set of feasible tasks is obtained by solving an optimization problem that minimizes the sum of non-negative variables, i.e., maximizes the number of feasible tasks.

Joint task offloading results in a non-convex problem due to coupling optimization variables. Moreover, the task placement is performed by obtaining binary variables, which makes the optimization problem further complicated. To deal with the optimization problem, we decouple transmit power allocation from task placement and computational resource allocation. Transmit power allocation is performed via the well-known convex-concave procedure (CCP) and a heuristic algorithm is proposed for task placement and computational resource allocation. CCP and the heuristic algorithm are alternatively applied until convergence. Note that both CCP and the heuristic algorithm preserve the monotonicity of convergence.

We also develop two baseline methods to evaluate the efficiency of our joint task offloading method. The first is a disjoint method in which transmit power allocation is performed independent of task placement. In doing so, the maximum acceptable E2E latency of each task is divided into a radio latency constraint and a non-radio latency constraint. We allocate transmit power under the radio latency constraint. Then, the task placement and computational resource

allocation are performed under the non-radio latency constraint.

The second baseline method achieves a lower bound on the optimal solution of the joint task offloading optimization problem. The lower bound is achieved by relaxing some constraints in the optimization problem, which comes from leveraging practical assumptions such as orthogonality of wireless channels in large-scale antenna array systems. The optimal solution is then found by an exhaustive search over all feasible task placement candidates, finding the optimal computational resource allocation for each placement candidate, and choosing the placement candidate that results in the lowest objective value.

D. Contributions

In this paper, we develop an energy-efficient task offloading method that offloads the computational burden of a task from a UE to one of executing servers in a network of NFV-enabled nodes. In doing so, a task is offloaded by sending non-processed data of the task from the UE to a radio remote head (RRH) over a radio link, sending the data from the RRH toward the executing server through a transport network, and sending the processed data back from the executing server to UE. We assume that each task is offloaded under a respective deadline, i.e., the E2E latency of task offloading is less than the maximum acceptable latency of the task.

The main contributions and achievements of this paper are as follows:

- We develop a joint task offloading method in a practical scenario, i.e., the proposed method allocates the transmit power, finds an executing server and the route to it, and allocates the computational resources in an energy-efficient manner. Moreover, the proposed method takes the E2E latency of task offloading into account. By the proposed method, the impact of weak radio links is compensated by placing the tasks in servers closer to UEs and consuming more computational resources. In contrast, limited computational resources are compensated by allocating more transmit power, resulting in an efficient and adaptive task offloading method.
- We propose a novel method for task placement and computational resource allocation. While the conventional routing methods find a route to a predetermined node, our proposed method jointly finds the executing server, its associated route, and the required computational resources in an energy-efficient manner.
- We find a lower bound on the objective function of the optimization problem in the feasibility analysis, i.e., an upper bound on the acceptance ratio of the proposed method. The lower

bound is obtained by relaxing some of constraints in the optimization problem, performing an exhaustive search over all feasible task placement candidates, and finding the optimal computational resource allocation by utilizing Karush-Kuhn-Tucker conditions.

- Simulation results show that the proposed joint method outperforms its disjoint counterpart in terms of acceptance ratio. Moreover, the lower bound on the optimal solution is almost tight because the joint method attains the lower bound in practical scenarios. Specifically, the optimality gap of the joint method is less than 5%.

E. Organization

The rest of the paper is organized as follows. Section II introduces the system model. Section III describes the optimization problem formulation. In Section IV, we propose joint task offloading while disjoint task offloading and lower bound on optimal task offloading are proposed in Sections V and VI, respectively. Simulation results are presented in Section VII and the paper is concluded in Section VIII.

F. Notation

The notation used in this paper are given as follows. The vectors are denoted by bold lowercase symbols. Operators $\|\cdot\|$ and $|\cdot|$ are vector norm and absolute value of a scalar, respectively. $(\mathbf{a})^{\mathrm{T}}$ is transpose of $\mathbf{a}$ and $[a]^{+} = \max(a, 0)$. $\mathcal{A}\backslash\{a\}$ discards the element a from the set $\mathcal{A}$. Finally, $\mathbf{a} \sim \mathcal{CN}(\mathbf{0}, \mathbf{\Sigma})$ is a complex Gaussian vector with zero mean and covariance matrix $\mathbf{\Sigma}$.

II. System Model

The structure of the radio access network, channel model, and signaling scheme as well as NFV-enabled network, computational resources, and capacity of network links are described in this section.

A. Radio Access Network (RAN)

We consider a centralized RAN architecture with a baseband unit (BBU) pool, which serves a set of U RRHs, each equipped with M antennas. The set of all users is denoted by $\mathcal{K}$. Each user is equipped with a single antenna and the total number of users is $K = |\mathcal{K}|$. The considered model is shown in Fig. 2. It is assumed that each RRH is connected to the BBU pool through a fronthaul link.

We assume that each user requests a single task. Task k is represented by a triplet $< L_k, D_k, T_k >$,

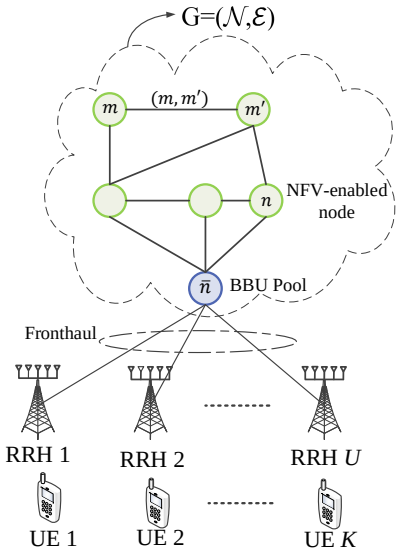


Fig. 2: System model.

where L_k is the load of task k (i.e., the required CPU cycles), D_k is the data size of task k (in terms of bits), and T_k is the maximum acceptable latency of task k.

Each UE transmits the non-processed data of its task to its serving RRH through a wireless link. We assume that each UE is served by a single RRH. The set of users served by RRH u is $\mathcal{K}_u = \{k \in \mathcal{K} | J_u^k = 1\}$ where J_u^k is an indicator which equals 1 if UE k is connected to RRH u (0 otherwise). In this paper, we assume that the UE-RRH assignment is given and fixed. Focusing on the wireless link, we assume a narrow-band block fading channel model [21]. The channel vector between UE k and RRH u is denoted by $\mathbf{h}_{u,k}$, where $\mathbf{h}_{u,k} = \sqrt{Q_{u,k}}\tilde{\mathbf{h}}_{u,k}$ in which $Q_{u,k}$ represents the path loss between RRH u and UE k and small-scale fading is modeled as $\tilde{\mathbf{h}}_{u,k} \sim \mathcal{CN}(\mathbf{0}, \mathbf{I}_M)$. Similar to [16], [17], we assume that the channel state information (CSI) is constant over the offloading time. As we show through simulations, this assumption is non-restrictive in practical scenarios in sub-6 GHz bands. UE k transmits a symbol $x_k \sim \mathcal{CN}(\mathbf{0}, 1)$ with transmit power ρ_k toward its serving RRH. The transmit power of UE k is constrained to a maximum value, i.e., $\rho_k \leq P_k^{\max} \quad \forall k$. The received signal vector at RRH u is:

$$\mathbf{y}_u = \sum_{k \in \mathcal{K}} \mathbf{h}_{u,k} \sqrt{\rho_k} x_k, \quad \forall u. \tag{1}$$

We assume the maximum ratio combining (MRC) at RRHs because of its simplicity. Nevertheless, MRC is asymptotically optimal in massive MIMO systems [30]. Therefore, the combined signal is:

$$\mathbf{z}_u = \mathbf{F}_u^{\mathrm{H}} \mathbf{y}_u, \quad \forall u, \tag{2}$$

where $\mathbf{F}_u = [\mathbf{f}_k], \forall k \in \mathcal{K}_u$ and $\mathbf{f}_k = \frac{\mathbf{h}_{u,k}}{\|\mathbf{h}_{u,k}\|}$. The estimated signal of UE k is:

$$z_k = \mathbf{f}_k^{\mathrm{H}} \mathbf{h}_{u,k} \sqrt{\rho_k} x_k + \sum_{j \in \mathcal{K} \setminus \{k\}} \mathbf{f}_k^{\mathrm{H}} \mathbf{h}_{u,j} \sqrt{\rho_k} x_k + \mathbf{f}_k^{\mathrm{H}} \mathbf{n}_u, \quad \forall k \in \mathcal{K}_u,$$

where $\mathbf{n}_u \sim \mathcal{CN}(\mathbf{0}, \sigma_{\mathrm{n}}^2 \mathbf{I}_M)$ is the received noise vector at RRH u. Thus, the signal to interference plus noise ratio (SINR) of UE k is:

$$\mathrm{SINR}_k = \frac{\|\mathbf{h}_{u,k}\|^2 \rho_k}{\sum_{j \in \mathcal{K} \setminus \{k\}} \frac{|\mathbf{h}_{u,k}^{\mathrm{H}} \mathbf{h}_{u,j}|^2}{\|\mathbf{h}_{u,k}\|^2} \rho_j + \sigma_{\mathrm{n}}^2}, \quad \forall k \in \mathcal{K}_u. \tag{3}$$

Hence, the achievable data rate by UE k is $R_k = W \log_2(1 + \mathrm{SINR}_k)$[1] bits per second (bps), where W is the radio access network bandwidth. The radio transmission latency of task k in the uplink is $T_k^{\mathrm{tx}} = \frac{D_k}{R_k}$[2]. The sum of data rates of UEs served by RRH u is less than the capacity of its fronthaul link, i.e., $\sum_{k \in \mathcal{K}_u} R_k \leq B_{\mathrm{f},u}, \forall u$. In this paper, similar to [10], [29], and [11], we assume that the processed data size of task k is small. Moreover, since the power budget of RRHs is generally large, the radio transmission latency in the downlink is assumed negligible.

B. NFV-enabled Network

The NFV-enabled network includes a graph $G = (\mathcal{N}, \mathcal{E})$, where $\mathcal{N}$ and $\mathcal{E}$ are the set of nodes and edges (or links), respectively. A typical node in $\mathcal{N}$ is denoted by n while the BBU pool is indicated by $\bar{n}$ (which is also a node in $\mathcal{N}$). The link between two nodes m and m' is denoted by (m, m'). Each NFV-enabled node is comprised of an executing server and a routing device. The processing capacity (i.e., the maximum CPU cycles per second that are carried out) of the executing server in NFV-enabled node n is indicated by Υ_n. Moreover, the capacity of link (m, m') is indicated by $B_{(m,m')}$ in terms of bps.

In this paper, we assume the full offloading scheme, i.e., the task of each user is completely executed in an executing server in the NFV-enabled network. Therefore, there is a need for placing each task to a proper executing server. A task placement decision consists of selecting an NFV-enabled node n and its associated path from $\bar{n}$ to n . We denote the b^{th} path between nodes $\bar{n}$ and n as p_n^b where $b \in \mathcal{B}_n = \{1 \cdots B_n\}$ and B_n is the total number of paths between nodes $\bar{n}$ and n. Note that a path between $\bar{n}$ and n may comprise some intermediate nodes,

[1]For wide-band channel model, the data rate of UE k is the sum rate over all sub-carriers allocated to UE k.

[2]No buffering is assumed in the transport network routing. Therefore, transmission time of tasks' data over the transport network links is not taken into account.

which only forward the tasks' data via their routing devices and do not deliver the data to their executing servers. We define decision variable $\xi^k_{p^b_n}$, which equals 1 when task k is offloaded to node n and sent over path p^b_n (0 otherwise). Each task is offloaded to one and only one node and path when we have:

$$\sum_{n\in\mathcal{N}}\sum_{b\in\mathcal{B}_n}\xi^k_{p^b_n}=1,\quad \forall k. \tag{4}$$

Indicator $I^{(m,m')}_{p^b_n}$ determines whether a link contributes to a path. The indicator is equal to 1 when link (m,m') contributes to path p^b_n (0 otherwise). Moreover, the set of all links that contribute to path p^b_n is $\mathcal{E}_{p^b_n}=\left\{(m,m')\in\mathcal{E}|I^{(m,m')}_{p^b_n}=1\right\}$. The amount of computational resources allocated to task k is denoted by υ_k (in terms of CPU cycles per second). Note that the execution of each tasks is performed at only one node. To ensure that the allocated computational resources do not violate the processing capacity of that node, we should have:

$$\sum_{k\in\mathcal{K}}\sum_{b\in\mathcal{B}_n}\upsilon_k\xi^k_{p^b_n}\le\Upsilon_n,\quad \forall n. \tag{5}$$

Since the data of task k is sent over the network with rate R_k, the aggregated rates of all tasks that pass a link should not exceed its capacity, which is guaranteed by the following constraint:

$$\sum_{k\in\mathcal{K}}\sum_{n\in\mathcal{N}}\sum_{b\in\mathcal{B}_n}I^{(m,m')}_{p^b_n}\xi^k_{p^b_n}R_k\le B_{(m,m')},\quad \forall(m,m')\in\mathcal{E}. \tag{6}$$

The execution latency of task k is $T^{\text{exe}}_k=\frac{L_k}{\upsilon_k}$. The processed data of task k is sent toward the BBU pool (i.e., node $\bar{n}$). In this paper, we assume the path of uplink and downlink are the same. Therefore, the overall propagation latency of task k over the path p^b_n is twice the propagation latency of path p^b_n. Thus, the propagation latency of task k is $T^{\text{prop}}_k=2\sum_{n\in\mathcal{N}}\sum_{b\in\mathcal{B}_n}\sum_{(m,m')\in\mathcal{E}_{p^b_n}}\xi^k_{p^b_n}\delta_{(m,m')}$, where $\delta_{(m,m')}$ is the propagation latency of link (m,m'). Table I summarizes the notation used in the paper.

III. Problem Formulation

In this section, we formulate the optimization problem of joint task offloading. Each task is offloaded under its E2E latency constraint and in an energy-efficient manner. The objective function is $\mathcal{E}(\boldsymbol{\xi},\boldsymbol{\upsilon},\boldsymbol{\rho})=\sum_{k\in\mathcal{K}}\rho_k+\eta\sum_{n\in\mathcal{N}}\sum_{k\in\mathcal{K}}\sum_{b\in\mathcal{B}_n}\Lambda_n\xi^k_{p^b_n}{\upsilon_k}^3$, where $\boldsymbol{\xi}=[\xi^1_{p^1_1},\cdots,\xi^K_{p^{B_N}_N}]^{\mathrm{T}}$, $\boldsymbol{\upsilon}=[\upsilon_1,\cdots,\upsilon_K]^{\mathrm{T}}$, and $\boldsymbol{\rho}=[\rho_1,\cdots,\rho_K]^{\mathrm{T}}$ are the vectors of all $\xi^k_{p^b_n},\upsilon_k$, and ρ_k, respectively; Λ_n denotes the computational energy efficiency coefficient of node n [18], and η is a weight. Note

TABLE I: Main Notation.

Notation	Definition	Notation	Definition
U, M, K	Number of RRHs, antennas and users	W	Radio access network bandwidth
$\mathcal{K}, \mathcal{N}, \mathcal{E}$	Set of all users, nodes and links	$\mathcal{K}_u$	Set of users served by RRH u
$P_k^{\max}$	Power budget of UE k	$\mathbf{h}_{u,k}$	Channel vector between user k and RRH u
L_k, D_k, T_k	Load, data size and maximum acceptable latency of task k	$\xi_{p_n^b}^k$	Decision variable for assignment of node n and its associated path p_n^b to task k
Υ_n	Processing capacity of node n	$B_{\mathrm{f},u}$	Capacity of fronthaul link for RRH u
$B_{(m,m')}$, $\delta_{(m,m')}$	Capacity and propagation latency of link (m, m')	Λ_n	Computational energy efficiency coefficient of the node n
p_n^b	b^{th} path between nodes $\bar{n}$ and n	υ_k	Computational resources allocated to task k
$\mathcal{B}_n$	Set of paths between nodes $\bar{n}$ and n	ρ_k	Allocated transmit power to UE k
$\mathcal{E}_{p_n^b}$	Set of all links that contribute in path p_n^b	α_k	Non-negative variable of task k
$I_{p_n^b}^{(m,m')}$	Indicator determining whether link (m, m') contributes in path p_n^b	R_k	Data rate of task k
J_u^k	Indicator determining whether UE k is assigned to RRH u	T_k^{exe}	Execution latency of task k
T_k^{tx}	Radio transmission latency of task k	T_k^{prop}	Propagation latency of task k

that the first term in $\mathcal{E}$ is the transmit power consumption and the second term is the power consumption of executing servers. Therefore, the joint task offloading optimization problem is:

$$
\begin{aligned}
\min_{\boldsymbol{\xi},\boldsymbol{\upsilon},\boldsymbol{\rho}} \quad & \mathcal{E}(\boldsymbol{\xi},\boldsymbol{\upsilon},\boldsymbol{\rho}) \\
\text{s.t.} \quad & \text{C1:} \quad T_k^{\mathrm{exe}} + T_k^{\mathrm{prop}} + T_k^{\mathrm{tx}} \leq T_k, \quad \forall k, \\
& \text{C2:} \quad \textstyle\sum_{k\in\mathcal{K}} \sum_{b\in\mathcal{B}_n} \upsilon_k \xi_{p_n^b}^k \leq \Upsilon_n, \quad \forall n, \\
& \text{C3:} \quad \textstyle\sum_{k\in\mathcal{K}} \sum_{n\in\mathcal{N}} \sum_{b\in\mathcal{B}_n} I_{p_n^b}^{(m,m')} \xi_{p_n^b}^k R_k \leq B_{(m,m')}, \quad \forall (m,m') \in \mathcal{E}, \\
& \text{C4:} \quad \textstyle\sum_{k\in\mathcal{K}_u} R_k \leq B_{\mathrm{f},u}, \quad \forall u, \\
& \text{C5:} \quad \rho_k \leq P_k^{\max}, \quad \forall k, \\
& \text{C6:} \quad \textstyle\sum_{n\in\mathcal{N}} \sum_{b\in\mathcal{B}_n} \xi_{p_n^b}^k = 1, \quad \forall k,
\end{aligned}
\tag{7}
$$

under variables: $\boldsymbol{\xi} \in \{0,1\}, \boldsymbol{\upsilon} \geq 0, \boldsymbol{\rho} \geq 0$. Constraint C1 guarantees that the maximum acceptable latency of task offloading is respected. Constraints C2 and C3 make sure that all tasks are offloaded without violation in processing capacity of nodes and capacity of links, respectively. Constraint C4 ensures the capacity of fronthaul links. Constraint C5 guarantees the power budget of UEs while constraint C6 makes sure that each task is offloaded to only one node and path.

IV. Joint Task Offloading (JTO)

In this section, we solve optimization problem (7). This problem is non-convex due to integer variable $\boldsymbol{\xi}$ and coupling variables in C1-C4. Therefore, we solve (7) by decoupling transmit power

allocation from task placement and computational resource allocation. In doing so, transmit power is allocated given task placement and allocated computational resources. Then, we perform task placement and computational resource allocation having allocated transmit powers. The proposed approach needs a feasible initialization. However, it is likely for constraint C1 to make (7) infeasible. Thus, we need to propose a feasibility analysis to find a feasible subset of tasks.

A. Feasibility Analysis

The feasible set of (7) is extended by adding a non-negative variable α_k to the maximum acceptable latency of task k. Thus, the feasibility problem is constructed by replacing the objective function of (7) with the sum of non-negative variables, i.e., $\sum_{k=1}^{K} \alpha_k$ [31]. The constraints which cause infeasibility are found by solving the feasibility problem and determining the constraints with positive values of non-negative variables. The feasibility problem is:

$$\begin{aligned} \min_{\boldsymbol{\xi},\boldsymbol{\upsilon},\boldsymbol{\rho},\boldsymbol{\alpha}} \quad & \textstyle\sum_{k\in\mathcal{K}} \alpha_k \\ \text{s.t.} \quad & \text{C1-a:} \quad T_k^{\text{exe}} + T_k^{\text{prop}} + T_k^{\text{tx}} \leq T_k + \alpha_k, \quad \forall k \in \mathcal{K} \\ & \text{C2-C6,} \end{aligned} \tag{8}$$

under variables: $\boldsymbol{\xi} \in \{0,1\}, \boldsymbol{\upsilon} \geq 0, \boldsymbol{\rho} \geq 0, \boldsymbol{\alpha} \geq 0$. Note that non-negative variables are added only to C1 because when C1 is eliminated, the optimization problem (7) is always feasible. Thus, we seek for the tasks whose maximum acceptable latencies are violated and eliminate them one by one until a subset of feasible tasks remains. The solution to (8) not only provides the infeasible constraints but also determines the level of infeasibility, i.e., constraints with larger values of non-negative variables need more resources to become feasible. Therefore, we first eliminate the tasks with larger values of non-negative variables.

Without loss of equivalence, we add the summation of inequalities in C1-a as a new constraint C7. Therefore, optimization problem (8) is restated as:

$$\begin{aligned} \min_{\boldsymbol{\xi},\boldsymbol{\upsilon},\boldsymbol{\rho},\boldsymbol{\alpha}} \quad & \textstyle\sum_{k\in\mathcal{K}} \alpha_k \\ \text{s.t.} \quad & \text{C1-a:} \quad T_k^{\text{exe}} + T_k^{\text{prop}} + T_k^{\text{tx}} \leq T_k + \alpha_k, \quad \forall k \\ & \text{C2-C6,} \\ & \text{C7:} \textstyle\sum_{k\in\mathcal{K}} \left(T_k^{\text{exe}} + T_k^{\text{prop}} + T_k^{\text{tx}} - T_k\right) \leq \sum_{k\in\mathcal{K}} \alpha_k. \end{aligned} \tag{9}$$

This optimization problem is equivalent with:

$$\begin{aligned} \min_{\boldsymbol{\xi},\boldsymbol{\upsilon},\boldsymbol{\rho},\boldsymbol{\alpha}} \quad & \textstyle\sum_{k\in\mathcal{K}} \left(T_k^{\text{exe}} + T_k^{\text{prop}} + T_k^{\text{tx}}\right) \\ \text{s.t.} \quad & \text{C1-a, and C2-C6,} \end{aligned} \tag{10}$$

in which the term $\sum_{k\in\mathcal{K}} T_k$ is removed from the objective because it is constant. We solve (10) by decoupling transmit power allocation from task placement and coumputational resource allocation. In other words, we solve (10) under variables $\boldsymbol{\upsilon}, \boldsymbol{\xi}, \boldsymbol{\alpha}$, having $\boldsymbol{\rho}$ fixed and vice versa. To perform task placement and computational resource allocation, we need an initial $\boldsymbol{\rho} = \boldsymbol{\rho}^0$ that satisfies C3 and C4, which are satisfied with a small value of R_k, i.e., small values of ρ_k. Next, we solve the following optimization problem:

$$\begin{aligned} \min_{\boldsymbol{\alpha},\boldsymbol{\upsilon},\boldsymbol{\xi}} \quad & \textstyle\sum_{k\in\mathcal{K}} \left(T_k^{\text{exe}} + T_k^{\text{prop}}\right) \\ \text{s.t.} \quad & \text{C1-a, C2,C3, and C6} \end{aligned} \tag{11}$$

by a heuristic method. As in Algorithm 1, we find variables $\boldsymbol{\xi}$ and $\boldsymbol{\upsilon}$ that minimize the objective of (11). Then, we set the non-negative variables for a feasible C1. In doing so, for task k, we calculate the amount of unused computational resources at all nodes, formally expressed as $\tilde{\Upsilon}_n^k = \Upsilon_n - \sum_{j\in\mathcal{K}\setminus\{k\}} \sum_{b\in\mathcal{B}_n} \upsilon_j \xi_{p_n^b}^j$. Morevoer, the available capacity of link (m, m') is $\tilde{B}_{(m,m')}^k = B_{(m,m')} - \sum_{j\in\mathcal{K}\setminus\{k\}} \sum_{n\in\mathcal{N}} \sum_{b\in\mathcal{B}_n} I_{p_n^b}^{(m,m')} \xi_{p_n^b}^j R_j$. A task is placed in node n only when there is a feasible path between $\bar{n}$ and n, i.e., a path with sufficient capacity in all of its links. The set of all such nodes is $\mathcal{N}_k$. For each $n \in \mathcal{N}_k$, we calculate $T_k^{\text{exe}} + T_k^{\text{prop}}$ when $\upsilon_k = \tilde{\Upsilon}_n^k$. Next, we find the node and feasible path that give the smallest $T_k^{\text{exe}} + T_k^{\text{prop}}$, denoted by $n^\star$ and $b^\star$, respectively. Note that from C1, the sufficient computational resources allocated to task k is $\upsilon_{\text{temp}} = \frac{L_k}{T_k - T_k^{\text{tx}} - T_k^{\text{prop}}}$. When $\tilde{\Upsilon}_{n^\star}^k \geq \upsilon_{\text{temp}}$, C1 is satisfied by setting $\upsilon_k = \upsilon_{\text{temp}}$ and $\alpha_k = 0$. Otherwise, we set $\upsilon_k = \tilde{\Upsilon}_{n^\star}^k$ and $\alpha_k = T_k^{\text{tx}} + T_k^{\text{exe}} + T_k^{\text{prop}} - T_k$. Next, the available computational resources of nodes and available capacity of links are updated and this process is repeated for all of tasks. Note that we begin with tasks that require lower resources, i.e., tasks with lower values of T_k.

After solving (11), we allocate the transmit power by solving:

$$\begin{aligned} \min_{\boldsymbol{\rho}} \quad & \textstyle\sum_{k=1}^{K} T_k^{\text{tx}} \\ \text{s.t.} \quad & \text{C1-a, and C3-C5.} \end{aligned} \tag{12}$$

Note that in the heuristic method, we have $T_k^{\text{tx}} + T_k^{\text{exe}} + T_k^{\text{prop}} = T_k + \alpha_k$. As a result, any feasible solution to (12) does not increase T_k^{tx} because (12) is infeasible for larger values of T_k^{tx}. Hence, replacing (12) with its feasibility problem counterpart does not impact the decreasing monotonicity of the objective function in (10). The feasibility problem of (12) is:

$$\begin{aligned} \text{find} \quad & \boldsymbol{\rho} \\ \text{s.t.} \quad & \text{C1-a, and C3-C5.} \end{aligned} \tag{13}$$

Algorithm 1: Heuristic Algorithm for Solving (11).

Input: $\boldsymbol{\rho}$

sort $\boldsymbol{\alpha}$**:** $T_{[1]} \leq T_{[2]} \leq \cdots T_{[|\mathcal{K}|]}$

for $k = [1] : [|\mathcal{K}|]$ **do**

% Find a feasible node according to capacity of paths terminated at that node

$\mathcal{N}_k = \{n \in \mathcal{N} | \exists b : R_k \leq \tilde{B}^k_{(m,m')} \forall (m,m') \in \mathcal{E}_{p^b_n}\}$

$\tilde{\Upsilon}^k_n = \Upsilon_n - \sum_{j \in \mathcal{K} \setminus \{k\}} \sum_{b \in \mathcal{B}_n} \upsilon_j \xi^j_{p^b_n}, \quad \forall n$

%Find the best node and its associated path

$(n^\star, b^\star) = \underset{n \in \mathcal{N}^k, b \in \mathcal{B}_n}{\arg\min}\ T^{\text{exe}}(\tilde{\Upsilon}^k_n) + T^{\text{prop}}(p^b_n)$

set $\quad \xi^k_{p^{b^\star}_{n^\star}} = 1$ and $\xi^k_{p^b_n} = 0, \forall (n,b) \neq (n^\star, b^\star)$

% Update computational resource allocation and non-negative variables

$\upsilon_{\text{temp}} = \frac{L_k}{T_k - T^{\text{tx}}_k - T^{\text{prop}}_k}$

if $\tilde{\Upsilon}^k_{n^\star} \geq \upsilon_{\text{temp}}$ **then**

set $\quad \upsilon^\star_k = \upsilon_{\text{temp}}$ and $\alpha^\star_k = 0$

else

$\upsilon^\star_k = \tilde{\Upsilon}^k_{n^\star}$ and $\alpha^\star_k = T^{\text{tx}}_k + T^{\text{exe}}_k + T^{\text{prop}}_k(p^{b^\star}_{n^\star}) - T_k$

Output: $\boldsymbol{\alpha}^\star, \boldsymbol{\xi}^\star, \boldsymbol{\upsilon}^\star$

In solving (13), we note that the constraints C1-a, C3 and C4 are non-convex. Therefore, we need to find a convexified version of (13).We use CCP [32] to convexify (13). In doing so, we reformulate C1-a as:

$$R_k \geq \frac{D_k}{T_k + \alpha_k - T^{\text{prop},i}_k - T^{\text{exe},i}_k}. \tag{14}$$

where $T^{\text{exe},i}_k$ and $T^{\text{prop},i}_k$ are the execution latency and propagation latency of task k obtained from the heuristic method in i^{th} iteration, respectively. In order to convexify (14), we need a concave approximation of R_k with respect to $\boldsymbol{\rho}$. The rate R_k is:

$$R_k = W \log_2 \left(\frac{\sum_{j \in \mathcal{K}} \frac{|\mathbf{h}^{\text{H}}_{u,k} \mathbf{h}_{u,j}|^2}{|\mathbf{h}_{u,j}|^2} \rho_j + \sigma^2_{\text{n}}}{\sum_{j \in \mathcal{K} \setminus \{k\}} \frac{|\mathbf{h}^{\text{H}}_{u,k} \mathbf{h}_{u,j}|^2}{|\mathbf{h}_{u,j}|^2} \rho_j + \sigma^2_{\text{n}}} \right), \quad k \in \mathcal{K}_u, \tag{15}$$

which is equivalent to:

$$R_k = \underbrace{W \log_2 \left(\sum_{u=1}^{U} \sum_{j \in \mathcal{K}_u} \frac{|\mathbf{h}^{\text{H}}_{u,k} \mathbf{h}_{u,j}|^2}{|\mathbf{h}_{u,j}|^2} \rho_j + \sigma^2_{\text{n}} \right)}_{h_k(\boldsymbol{\rho})} - \underbrace{W \log_2 \left(\sum_{u=1}^{U} \sum_{j \in \mathcal{K}_u \setminus \{k\}} \frac{|\mathbf{h}^{\text{H}}_{u,k} \mathbf{h}_{u,j}|^2}{|\mathbf{h}_{u,j}|^2} \rho_j + \sigma^2_{\text{n}} \right)}_{g_k(\boldsymbol{\rho})}. \tag{16}$$

Both $h_k(\boldsymbol{\rho})$ and $g_k(\boldsymbol{\rho})$ are concave functions of $\boldsymbol{\rho}$. Thus, we need to find a linear approximation of $g_k(\boldsymbol{\rho})$, which is $\hat{g}_k(\boldsymbol{\rho}) = g_k(\boldsymbol{\rho}^0) + \nabla g_k(\boldsymbol{\rho}^0)^{\mathrm{T}}(\boldsymbol{\rho} - \boldsymbol{\rho}^0)$, where:

$$[\nabla g_k(\boldsymbol{\rho})]_i = \begin{cases} \dfrac{W \sum_{u=1}^{U} I_u^i \frac{|\mathbf{h}_{u,k}^{\mathrm{H}} \mathbf{h}_{u,i}|^2}{|\mathbf{h}_{u,i}|^2}}{\ln(2)\left(\sum_{u=1}^{U} \sum_{j \in \mathcal{K}_u \setminus \{k\}} \frac{|\mathbf{h}_{u,k}^{\mathrm{H}} \mathbf{h}_{u,j}|^2}{|\mathbf{h}_{u,j}|^2} \rho_j + \sigma_{\mathrm{n}}^2\right)}, i \in \mathcal{K} \setminus \{k\}, \\ 0, \qquad i = k. \end{cases} \tag{17}$$

Next, we focus on the convex approximation of C3 and C4. To this aim, we find a convex approximation of R_k, which is found by linear approximation of $h_k(\boldsymbol{\rho})$. Thus, we have $\hat{h}_k(\boldsymbol{\rho}) = h_k(\boldsymbol{\rho}^0) + \nabla h_k(\boldsymbol{\rho}^0)^{\mathrm{T}}(\boldsymbol{\rho} - \boldsymbol{\rho}^0)$, where:

$$[\nabla h_k(\boldsymbol{\rho})]_i = \frac{W \sum_{u=1}^{U} I_u^i \frac{|\mathbf{h}_{u,k}^{\mathrm{H}} \mathbf{h}_{u,i}|^2}{|\mathbf{h}_{u,i}|^2}}{\ln(2)\left(\sum_{u=1}^{U} \sum_{j \in \mathcal{K}_u} \frac{|\mathbf{h}_{u,k}^{\mathrm{H}} \mathbf{h}_{u,j}|^2}{|\mathbf{h}_{u,j}|^2} \rho_j + \sigma_{\mathrm{n}}^2\right)}, \quad i \in \mathcal{K}, \tag{18}$$

Finally, the convexified version of (12) is:

$$\begin{aligned} &\text{find} \quad \boldsymbol{\rho} \\ &\text{s.t.} \quad \text{C1-b} \quad h_k(\boldsymbol{\rho}) - \hat{g}_k(\boldsymbol{\rho}) \geq \tfrac{D_k}{T_k + \alpha_k - T_k^{\mathrm{prop},i} - T_k^{\mathrm{exe},i}}, \quad \forall k \in \mathcal{K} \\ &\qquad \text{C3-a:} \quad \textstyle\sum_{k \in \mathcal{K}} \sum_{n \in \mathcal{N}} \sum_{b \in \mathcal{B}_n} I_{p_n^b}^{(m,m')} \xi_{p_n^b}^k \left(\hat{h}_k(\boldsymbol{\rho}) - g_k(\boldsymbol{\rho})\right) \leq B_{(m,m')}, \quad \forall (m,m') \in \mathcal{E} \\ &\qquad \text{C4-a:} \quad \textstyle\sum_{k \in \mathcal{K}_u} \left(\hat{h}_k(\boldsymbol{\rho}) - g_k(\boldsymbol{\rho})\right) \leq B_{\mathrm{f},u}, \forall u \\ &\qquad \text{C5:} \quad \rho_k \leq P_k^{\max}, \forall k, \end{aligned} \tag{19}$$

under variable: $\boldsymbol{\rho} \geq 0$. Note that, based on CCP, any feasible solution of (19) is also feasible in (13) [32]. The feasibility problem (8) is solved by alternatively solving (11) and (19). Then, we reject the task that makes (7) infeasible. According to Algorithm 2, we find the value of the maximum non-negative variable. If the value is positive, its associated task is rejected, the set of served tasks is updated, and (8) is solved for updated set of tasks. This procedure continues until all non-negative variables are zero. The output of Algorithm 2 is feasible subset of tasks $\mathcal{K}^\star$ as well as the solution of (8), i.e., the values of $\boldsymbol{\xi}^{\mathrm{ini}}, \boldsymbol{\rho}^{\mathrm{ini}}$, and $\boldsymbol{\upsilon}^{\mathrm{ini}}$, which are utilized as initialization for solving (7).

B. Optimization

Given the feasible solution $\boldsymbol{\xi}^{\mathrm{ini}}, \boldsymbol{\rho}^{\mathrm{ini}}, \boldsymbol{\upsilon}^{\mathrm{ini}}$, and the set of accepted tasks $\mathcal{K}^\star$, we seek for the solution of (7). Similar to Algorithm 2, we decouple power allocation from task placement and

Algorithm 2: JTO Feasibility Analysis for Solving (8).

Initialize: $\mathcal{K}=\{1,\cdots,K\}, \quad \boldsymbol{\xi}=\mathbf{0}, \quad \boldsymbol{\rho}^0$: very small

repeat

  $i=0$

  repeat

    % Allocate transmit power, computational resources, and place the tasks

    Solve (11) via Algorithm 1 and return $\boldsymbol{\upsilon}^{i+1}$, $\boldsymbol{\xi}^{i+1}$, and $\boldsymbol{\alpha}^{i+1}$

    Solve (19) and return $\boldsymbol{\rho}^{i+1}$

    $i=i+1$

  until $\sum_{k\in\mathcal{K}}\alpha_k^i-\sum_{k\in\mathcal{K}}\alpha_k^{i+1}\leq\epsilon$ *or* $i\geq I_{\max}$

  % Discard the infeasible task

  $k^\star=\arg\max_{k\in\mathcal{K}}\alpha_k$

  if $\alpha_{k^\star}>0$ **then**

    $\mathcal{K}=\mathcal{K}\backslash\{k^\star\}$

until $\sum_{k\in\mathcal{K}}\alpha_k=0$

Output: $\boldsymbol{\xi}^{\text{ini}}=\boldsymbol{\xi}^{i+1}, \boldsymbol{\rho}^{\text{ini}}=\boldsymbol{\rho}^{i+1}, \boldsymbol{\upsilon}^{\text{ini}}=\boldsymbol{\upsilon}^{i+1}$, and $\mathcal{K}^\star=\mathcal{K}$

coumputational resource allocation. The optimization problemof task placement and coumputational resource allocation is:

$$\begin{aligned}\min_{\boldsymbol{\upsilon},\boldsymbol{\xi}} \quad & \textstyle\sum_{n\in\mathcal{N}}\sum_{k\in\mathcal{K}}\sum_{b\in\mathcal{B}_n}\Lambda_n\xi^k_{p_n^b}{\upsilon_k}^3\\ \text{s.t.} \quad & \text{C1-C3, and C6,}\end{aligned}\tag{20}$$

which is non-convex. Note that the objective of (20) is an increasing function of υ_k and allocating lower computational resources to task k decreases the power consumption. But, allocating lower computational resources increases execution latency and violates the E2E latency constraint. As a result, we need to find nodes with smaller propagation latency to compensate for increased execution latency. In doing so, we find a subset of nodes with smaller propagation latency than the current executing server and with sufficient capacity of links terminating at that nodes. This set of nodes is $\mathcal{N}'_k=\{n'\in\mathcal{N}|\exists b': R_k\leq\tilde{B}^k_{(m,m')}\forall(m,m')\in\mathcal{E}_{p^{b'}_{n'}}$ and $T_k^{\text{prop}}(p^{b'}_{n'})\leq T_k^{\text{prop}}(p^b_n)\}$, where we assume task k is previously placed through path p^b_n. For each node in $\mathcal{N}'_k$, we calculate the minimum computational resources that satisfy the E2E latency constraint, i.e., $\upsilon_{\text{temp}}=\frac{L_k}{T_k-T_k^{\text{tx}}-T_k^{\text{prop}}(p^{b'}_{n'})}$. When $\tilde{\Upsilon}^k_{n'}\geq\upsilon_{\text{temp}}$ and $\Lambda_{n'}\upsilon^3_{\text{temp}}\leq\Lambda_n\upsilon_k^3$, we ensure that task placement through $p^{b'}_{n'}$ and computational resource allocation υ_{temp} are feasible and result in

lower power consumption. Therefore, we set $\upsilon_k = \upsilon_{\text{temp}}$. Otherwise, we reinstate υ_k for task k. Algorithm 3 begins with the tasks with larger power consumption, i.e., $\Lambda_{n_k}\upsilon_k^3$, where n_k denotes the executing server of task k. This procedure is repeated for all accepted tasks.

Algorithm 3: Heuristic Algorithm for Solving (20).

Input: $\boldsymbol{\xi}^{\text{ini}}, \boldsymbol{\rho}^{\text{ini}}, \boldsymbol{\upsilon}^{\text{ini}}$

sort: $\Lambda_{[1]}\upsilon_{[1]}^3 \leq \Lambda_{[2]}\upsilon_{[2]}^3 \leq \cdots \Lambda_{[K]}\upsilon_{[K]}^3$

for $k = [1] : [|\mathcal{K}|]$ **do**

  % Find a feasible node according to capacity of paths terminated at that node

  $\mathcal{N}'_k = \{n' \in \mathcal{N} | \exists b' : R_k \leq \tilde{B}^k_{(m,m')}, \forall (m,m') \in \mathcal{E}_{p^{b'}_{n'}}$ and $T_k^{\text{prop}}(p^{b'}_{n'}) \leq T_k^{\text{prop}}(p^b_n)\}$

  for $n' \in \mathcal{N}'_k$ **do**

    $\upsilon_{\text{temp}} = \frac{L_k}{T_k - T_k^{\text{tx}} - T_k^{\text{prop}}(p^{b'}_{n'})}$

    $\tilde{\Upsilon}^k_{n'} = \Upsilon_{n'} - \sum_{j \in \mathcal{K}\setminus\{k\}} \sum_{b \in \mathcal{B}_{n'}} \upsilon_j \xi^j_{p^b_{n'}}$

    if $\upsilon_{\text{temp}} \geq \tilde{\Upsilon}^k_{n'}$ and $\Lambda_{n'}\upsilon_{\text{temp}}^3 \leq \Lambda_{n^\star}\upsilon_k^3$ **then**

      set $\upsilon_k^\star = \upsilon_{\text{temp}}$

      set $\xi^{k\star}_{p^{b'}_{n'}} = 1$ and $\xi^{k\star}_{p^b_n} = 0, \forall (n,b) \neq (n',b')$

      break

Output: $\boldsymbol{\xi}^\star, \boldsymbol{\upsilon}^\star$

The sub-problem of transmit power allocation, after convexification, is:

$$\begin{aligned} \min_{\boldsymbol{\rho}} \quad & \textstyle\sum_{k \in \mathcal{K}} \rho_k \\ \text{s.t.} \quad & \text{C1-c:} \quad h_k(\boldsymbol{\rho}) - \hat{g}_k(\boldsymbol{\rho}) \geq \frac{D_k}{T_k - T_k^{\text{prop},i} - T_k^{\text{exe},i}}, \forall k \in \mathcal{K} \\ & \text{C3-a, C4-a, and C5.} \end{aligned} \tag{21}$$

Based on CCP in Algorithm 4 and starting from $\boldsymbol{\rho}^0 = \boldsymbol{\rho}^{\text{ini}}$, an iterative solution of (21) provides a sub-optimal transmit power allocation. Finally, optimization problem (7) is solved via Algorithm 5, which alternatively solves optimization problem (11) via Algorithm 3 and optimization problem (21) via Algorithm 4.

From the implementation point of view, BBU is responsible for gathering the required information, performing resource allocation, and sending the decisions to the associated entities. Specifically, in JTO, BBU needs to acquire CSI of UEs and the available computational resources in the NFV-enabled nodes. CSI of each UE is estimated at its serving RRH and is forwarded through fronthaul links with negligible latency. In addition, each NFV-enabled node sends the

Algorithm 4: Power Allocation in JTO.

Input: $\boldsymbol{\rho}^0 = \boldsymbol{\rho}^{\text{ini}}$, $i = 0$, $\epsilon = 10^{-3}$, $I^{\rho}_{\max} = 10^2$

repeat

  % Allocate power to users

  Solve (21) and return $\boldsymbol{\rho}^{i+1}$

  $i = i + 1$

until $\sum_{k\in\mathcal{K}} \rho^i_k - \sum_{k\in\mathcal{K}} \rho^{i+1}_k \leq \epsilon$ or $i \geq I^{\rho}_{\max}$

Output: $\boldsymbol{\rho}^{\star} = \boldsymbol{\rho}^{i+1}$

available computational resources to the BBU through the transport network. After performing JTO, BBU transmits the value of allocated powers to RRHs. Next, BBU forwards the received data of tasks as well as the obtained computational resources to associated NFV-enabled nodes based on task placement variables. In the downlink, the processed data of tasks are sent to BBU, which in turn transmits UEs processed data to their serving RRH.

Algorithm 5: JTO Optimization Algorithm for Solving (7).

Input: $\boldsymbol{\xi}^0 = \boldsymbol{\xi}^{\text{ini}}, \boldsymbol{\rho}^0 = \boldsymbol{\rho}^{\text{ini}}, \boldsymbol{\upsilon}^0 = \boldsymbol{\upsilon}^{\text{ini}}, \mathcal{K}^{\star}$, $i = 0$

repeat

  % Place the tasks and allocate the computational resources

  Solve (20) via Algorithm 3 and return $\boldsymbol{\upsilon}^{i+1}$ and $\boldsymbol{\xi}^{i+1}$

  % Allocate the transmit power

  Solve (21) via CCP in Algorithm 4 and return $\boldsymbol{\rho}^{i+1}$

  $i = i + 1$

until $\mathcal{E}(\boldsymbol{\xi}^i, \boldsymbol{\upsilon}^i, \boldsymbol{\rho}^i) - \mathcal{E}(\boldsymbol{\xi}^{i+1}, \boldsymbol{\upsilon}^{i+1}, \boldsymbol{\rho}^{i+1}) \leq \epsilon$ *or* $i \geq I_{\max}$

Output: $\boldsymbol{\xi}^{\star}, \boldsymbol{\rho}^{\star}, \boldsymbol{\upsilon}^{\star}$

C. Convergence analysis

In this subsection, we prove the convergence of Algorithms 2 and 5.

Theorem 1. *Algorithm 2 is convergent.*

Proof. We show that the objective value of (8), i.e., $\sum_{k\in\mathcal{K}} \alpha_k$, is non-increasing in each step of Algorithm 2 and since the objective value is lower bounded by zero, Algorithm 2 is convergent. In i^{th} iteration of Algorithm 2, Algorithm 1 sets α^{i+1}_k either equal to 0 when E2E latency of task k is guaranteed or equal to $T^{\text{tx}}_k + T^{\text{exe}}_k + T^{\text{prop}}_k - T_k$ when E2E latency is larger than its maximum acceptable value. Therefore, we have $\alpha^{i+1}_k = [T^{\text{exe}}_k + T^{\text{prop}}_k + T^{\text{tx}}_k - T_k]^+$. Hence, we need to show

that $\sum_{k\in\mathcal{K}}(T_k^{\mathrm{tx}}+T_k^{\mathrm{exe}}+T_k^{\mathrm{prop}})$ does not increase after i^{th} iteration. Algorithm 1 affloads task k so that $T_k^{\mathrm{exe}}+T_k^{\mathrm{prop}}$ in the objective of (11) is minimized (line 5 in Algorithm 1). As a result, Algorithm 1 does not increase the objective value of (11), i.e., $\sum_{k\in\mathcal{K}}(T_k^{\mathrm{prop}}(\boldsymbol{\xi}^{i+1})+T_k^{\mathrm{exe}}(\boldsymbol{\upsilon}^{i+1}))\leq \sum_{k\in\mathcal{K}}(T_k^{\mathrm{prop}}(\boldsymbol{\xi}^{i})+T_k^{\mathrm{exe}}(\boldsymbol{\upsilon}^{i}))$. Moreover, as discussed in subsection IV-A, Algorithm 1 makes C1-a active, i.e., $T_k^{\mathrm{tx}}(\boldsymbol{\rho}^i)=T_k+\alpha_k^{i+1}-T_k^{\mathrm{exe}}(\boldsymbol{\upsilon}^{i+1})-T_k^{\mathrm{prop}}(\boldsymbol{\xi}^{i+1})$, and therefore, any feasible solution to (13) does not increase the objective vlaue of (12), i.e., $\sum_{k\in\mathcal{K}}T_k^{\mathrm{tx}}(\boldsymbol{\rho}^{i+1})\leq\sum_{k\in\mathcal{K}}T_k^{\mathrm{tx}}(\boldsymbol{\rho}^{i})$, which gives $\sum_{k\in\mathcal{K}}(T_k^{\mathrm{exe}}(\boldsymbol{\upsilon}^{i+1})+T_k^{\mathrm{prop}}(\boldsymbol{\xi}^{i+1})+T_k^{\mathrm{tx}}(\boldsymbol{\rho}^{i+1}))\leq\sum_{k\in\mathcal{K}}(T_k^{\mathrm{exe}}(\boldsymbol{\upsilon}^{i})+T_k^{\mathrm{prop}}(\boldsymbol{\xi}^{i})+T_k^{\mathrm{tx}}(\boldsymbol{\rho}^{i}))$. As a result, we have $\sum_{k\in\mathcal{K}}\alpha_k^{i+1}\leq\sum_{k\in\mathcal{K}}\alpha_k^{i}$, that is, Algorithm 2 is convergent.

Note that Algorithm 2 may eliminate the task with maximum non-negative variable. This elimination is equivalent to removing the constraints of (8) associated with the eliminated task. Note that, eliminating a task increases the available capacity of links in transport network and available computational resources in NFV-enabled nodes. As a result, a search space of Algorithm 1 increases, which may result in lower propagation and execution latencies. Moreover, eliminating a task extends the feasible set of (13). Therefore, data rate of users may increase, which in turn may decrease $\sum_{k\in\mathcal{K}}T_k^{\mathrm{tx}}$. As a result, eliminating the task with maximum non-negative variable does not increase the objective of (8). □

Theorem 2. *Algorithm 5 is convergent.*

Proof. Algorithm 5 solves (7) by alternating minimization of (20) and (21). Therefore, we need to show that Algorithm 3 (which solves (20)) and Algorithm 4 (which solves (21)) do not increase the objective value of (7). According to line 7 of Algorithm 3, computational resource allocation and task placement do not increase the objective value of (20). In addition, based on [32], convergence of Algorithm 4 is guaranteed and CCP does not increase the objective of (21). As a result, the objective value of (7) is non-increasing in each iteration, and since $\Psi(\boldsymbol{\xi},\boldsymbol{\upsilon},\boldsymbol{\rho})$ is lower bounded by zero, Algorithm 5 is convergent. □

D. Summary of JTO

Herein we summarize JTO. We obtain a set of feasible tasks by solving (8). In doing so, we decouple the power allocation from task placement and computational resource allocation, which are performed by solving (13) and Algorithm (1), respectively. Then, we solve (7) for feasible tasks via Algorithm 5, which includes the alternating minimization of (20) and (21) via Algorithm 3 and Algorithm 4, respectively.

E. Computational Complexity (CC) Analysis

In this section, we analyze CC of the proposed algorithms. CC of JTO is equal to CC of feasibility analysis in Algorithm 2 and CC of optimization in Algorithm 5. Algorithm 2 includes two nested while loops and CC of the inner loop is equal to CC of Algorithm 1 and CC of solving (19). CC of Algorithm 1 depends on the required computations for calculation of the parameters in Algorithm 1, which are provided in Table II where B is the maximum number of the paths between any node and $\bar{n}$ whereas E is the total number of the edges in network graph G. Hence, CC of Algorithm 1 is $CC^1 = \mathcal{O}(K^2NBE)$.

TABLE II: CC of obtaining parameters in Algorithm 1.

Parameter	**CC**	**Parameter**	**CC**
$\mathcal{N}^k$	$N \times E$	$\tilde{B}^k_{(m,m')}$	$N \times B \times (K-1)$
$\tilde{\Upsilon}^k_n$	$B \times (K-1)$	$(n^\star, b^\star)$	$N \times B$

Problem (19) is solved via CVX, which exploits IPM for finding the optimal solution [33]. Based on [34] and [35], the required number of iterations for IPM to converge is $\frac{\log N_{\rm c}}{t^0 \varrho \log \varsigma}$, where $N_{\rm c} = 2K+U+E$ is the total number of constraints in (19), t^0 is the initial point for approximation of the barrier function, ϱ is the desired accuracy of convergence and $0 < \varsigma \ll 1$ is used for updating the stepsize of the barrier function accuracy. Note that the inner loop in Algorithm 2 is repeated at most $KI_{\max}$ times. As a result, CC of Algorithm 2 is

$$CC^2 = K \times I_{\max} \left(\frac{\log(2K+U+E)}{t^0 \varrho \log \varsigma} + \mathcal{O}(K^2NBE) \right). \tag{22}$$

Algorithm 5 includes alternating minimization of (20) and (21) via Algorithm 3 and Algorithm 4, respectively. Note that CC of Algorithm 3 is in the same order as that of Algorithm 1, i.e., $CC^3 = CC^1$. Algorithm 4 solves (21) at most $I^\rho_{\max}$ times and CC of solving (21) is equal to CC of (19). As a result, CC of Algorithm 4 is

$$CC^4 = I^\rho_{\max} \frac{\log(2K+U+E)}{t^0 \varrho \log \varsigma}. \tag{23}$$

Algorithms 3 and 4 are executed at most $I_{\max}$ times. Therefore, CC of Algorithm 5 is $CC^5 = I_{\max}\left(CC^3 + CC^4\right)$. Finally, CC of JTO is $CC^{\rm JTO} = CC^2 + CC^5$. Note that Algorithm 1, Algorithm 3, and IPM are of polynomial time complexity. Therfore, JTO is also of polynomial time complexity.

V. DISJOINT TASK OFFLOADING (DTO)

In DTO, transmit power allocation is independent of task placement and computational resource allocation. The transmit power is allocated under a radio latency constraint, i.e., $T_k^{\mathrm{tx}} \leq T_k^{\mathrm{RAN}}$. Then, the task placement and computational resource allocation are performed so that $T_k^{\mathrm{prop}} + T_k^{\mathrm{exe}} \leq T_k - T_k^{\mathrm{RAN}}$. The convexified sub-problem of the transmit power allocation is:

$$\begin{aligned} \min_{\boldsymbol{\rho}} \quad & \textstyle\sum_{k\in\mathcal{K}} \rho_k \\ \text{s.t.} \quad & \text{C1-d:} \quad h_k(\boldsymbol{\rho}) - \hat{g}_k(\boldsymbol{\rho}) \geq \frac{D_k}{T_k^{\mathrm{RAN}}}, \quad \forall k \in \mathcal{K} \\ & \text{C4-a, and C5.} \end{aligned} \tag{24}$$

According to discussion on discussion on (7), a feasibility analysis is needed for (24). Similar to JTO, the feasibility problem of (24) is:

$$\begin{aligned} \text{find} \quad & \boldsymbol{\rho} \\ \text{s.t.} \quad & \text{C1-e:} \quad h_k(\boldsymbol{\rho}) - \hat{g}_k(\boldsymbol{\rho}) \geq \frac{D_k}{T_k^{\mathrm{RAN}}+\alpha_k}, \quad \forall k \in \mathcal{K} \\ & \text{C4-a, and C5,} \end{aligned} \tag{25}$$

which is solved via CVX. Next, the non-negative variables are updated as $\alpha_k = [T_k^{\mathrm{tx}} - T_k^{\mathrm{RAN}}]^+$ and the task with maximum non-negative variable is eliminated. This procedure is repeated until a feasible subset of tasks for transmit power allocation is obtained. After this step, (24) is solved with the feasible subset of tasks. The transmit power allocation phase of DTO is provided in Algorithm 6.

Having obtained transmit power $\boldsymbol{\rho}$, task placement and computational resource allocation are performed, whose associated sub-problem is:

$$\begin{aligned} \min_{\boldsymbol{\xi},\boldsymbol{\upsilon}} \quad & \textstyle\sum_{n\in\mathcal{N}} \sum_{k\in\mathcal{K}} \sum_{b\in\mathcal{B}_n} \Lambda_n \xi_{p_n^b}^k {\upsilon_k}^3 \\ \text{s.t.} \quad & \text{C1-f:} \quad T_k^{\mathrm{prop}} + T_k^{\mathrm{exe}} \leq T_k - T_k^{\mathrm{RAN}}, \quad \forall k \in \mathcal{K}, \\ & \text{C2, C3, and C6.} \end{aligned} \tag{26}$$

A feasibility analysis is also needed for solving (26). Similar to the transmit power allocation, we introduce a set of non-negative variables. The resulting sub-problem is similar to (11) by replacing C1-a with C1-f, which is solved by algorithm 1. After obtaining a set of feasible tasks, (26) is solved via Algorithm 3. The feasibility analysis and optimization of DTO is provided in Algorithm 7.

Algorithm 6: DTO Transmit Power Allocation.

Input: $\mathcal{K}=\{1,\cdots,K\}, \boldsymbol{\alpha}^0$: very large, $\boldsymbol{\rho}^0$: very small, $T_k^{\mathrm{RAN}}=(0,T_k)$

repeat
  $i=0$
  repeat
    % Allocate the transmit power to users
    Solve (25) via CVX and set $\boldsymbol{\rho}^{i+1}=\boldsymbol{\rho}^{\star}$
    % Update the non-negative variables
    $\alpha_k^{i+1}=[T_k^{\mathrm{tx}}-T_k^{\mathrm{RAN}}]^+, \quad \forall k\in\mathcal{K}$
    $i=i+1$
  until $\sum_{k\in\mathcal{K}}\alpha_k^i-\sum_{k\in\mathcal{K}}\alpha_k^{i+1}\leq\epsilon$ *or* $i\geq I_{\max}$
  $k^{\star}=\arg\max_{k\in\mathcal{K}}\alpha_k$
  % Discard the infeasible task
  if $\alpha_{k^{\star}}>0$ **then**
    $\mathcal{K}=\mathcal{K}\backslash\{k^{\star}\}$
until $\sum_{k\in\mathcal{K}}\alpha_k=0$
% Minimize the transmit power
Solve (24) via CCP in Algorithm 4 and return $\boldsymbol{\rho}^{\star}$

Output: $\boldsymbol{\rho}^{\star}, \mathcal{K}^{\mathrm{RAN}}=\mathcal{K}$

A. CC of DTO

CC of DTO is equal to CC of Algorithms 6 and 7. Algorithm 6 includes two nested loops, wherein the inner loop includes solving (25) via CVX, whose CC is $\frac{\log(2K+U)}{t^0\varrho\log\varsigma}$. Moreover, Algorithm 6 minimizes the transmit power via CCP, whose CC is derived in subsection IV-E. As a result, CC of Algorithm 6 is $CC^6=(KI_{\max}+I_{\max}^{\rho})\frac{\log(2K+U)}{t^0\varrho\log\varsigma}$.

Algorithm 7 includes two nested while loops, in which Algorithm 1 is executed. Based on CC of Algorithm 1 derived in subsection IV-E, CC of feasibility analysis in Algorithm 7 is $KI_{\max}\mathcal{O}(K^2NBE)$. Algorithm 7 also includes executing Algorithm 3 at most $I_{\max}$ times. As a result, CC of Algorithm 7 is $CC^7=(K+1)I_{\max}\mathcal{O}(K^2NBE)$. Based on above, CC of DTO is $CC^{\mathrm{DTO}}=CC^6+CC^7$. Note that although CC^{DTO} is larger than CC^{JTO}, they are in the same order of complexity. The difference of CCs comes from the fact that CCP is performed at most $I_{\max}$ times in JTO and only once in DTO.

Algorithm 7: DTO Computational Resource Allocation and Task Placement.

Input: $\mathcal{K}^{\mathrm{RAN}}$, $\boldsymbol{\xi} = \mathbf{0}$

repeat
  $i = 0$
  repeat
    % Allocate transmit power, computational resources, and place the tasks
    Solve (26) via Algorithm 1 given $\boldsymbol{\upsilon}^i$, $\boldsymbol{\xi}^i$, and $\boldsymbol{\alpha}^i$ and return $\boldsymbol{\upsilon}^{i+1}$, $\boldsymbol{\xi}^{i+1}$, and $\boldsymbol{\alpha}^{i+1}$
    $i = i + 1$
  until $\sum_{k\in\mathcal{K}} \alpha_k^i - \sum_{k\in\mathcal{K}} \alpha_k^{i+1} \leq \epsilon$ *or* $i \geq I_{\max}$
  % Find the task with maximum non-negative variable
  $k^\star = \arg\max_{k\in\mathcal{K}} \alpha_k$
  if $\alpha_{k^\star} > 0$ **then**
    % Discard the infeasible task
    $\mathcal{K} = \mathcal{K} \backslash \{k^\star\}$
until $\sum_{k\in\mathcal{K}} \alpha_k = 0$
$i = 0$
repeat
  % Allocate computational resources and place the tasks
  Given $\boldsymbol{\upsilon}^i$ and $\boldsymbol{\xi}^i$, solve (26) via Algorithm 3 and return $\boldsymbol{\upsilon}^{i+1}$ and $\boldsymbol{\xi}^{i+1}$
until $\Psi(\boldsymbol{\xi}^i, \boldsymbol{\upsilon}^i, \boldsymbol{\rho}^\star) - \Psi(\boldsymbol{\xi}^{i+1}, \boldsymbol{\upsilon}^{i+1}, \boldsymbol{\rho}^\star) \leq \epsilon$ *or* $i \geq I_{\max}$

Output: $\boldsymbol{\xi}^\star, \boldsymbol{\upsilon}^\star$

VI. Lower Bound on Optimal Solution (LTO)

Since the optimization problem (8) is non-convex, without loss of the optimality, we make some assumptions to resolve the non-convexity of (8). First, we note that it is very likely for the fiber-optic links to have sufficient capacity for carrying the traffic of UEs, which is the case for frontahul links and any wired link in the transport network. As a result, we relax the constraints C3 and C4 from (8). Note that the relaxation of C3 and C4 extends the feasible set of (8), resulting in a lower bound on the optimal solution to (8). In addition, with a large number of antenna elements at RRHs, the channel vectors between different RRHs and a specific user are approximately orthogonal, i.e., $|\mathbf{h}_{u,k}^{\mathrm{H}} \mathbf{h}_{u,j}| \approx 0$ for all $j \neq k$ [30]. Therefore, the interference in wireless channels is negligible and (15) becomes:

$$R_k = W \log_2 \left(1 + \frac{|\mathbf{h}_{u,k}|^2}{\sigma_{\mathrm{n}}^2} \rho_k\right), \quad k \in \mathcal{K}_u. \tag{27}$$

The elimination of the interference increases R_k with the same amount of power allocated to each UE, which again results in a lower bound on the optimal solution to (8). Based on the fact that $\min\limits_{\boldsymbol{\alpha},\boldsymbol{\xi},\boldsymbol{\upsilon},\boldsymbol{\rho}} \sum_{k\in\mathcal{K}} \alpha_k = \min\limits_{\boldsymbol{\alpha},\boldsymbol{\xi},\boldsymbol{\upsilon}} \left(\min\limits_{\boldsymbol{\rho}} \sum_{k\in\mathcal{K}} \alpha_k \right)$, the optimal power allocation is the solution to:

$$\begin{aligned} \min_{\boldsymbol{\rho}} \quad & \textstyle\sum_{k\in\mathcal{K}} \alpha_k \\ \text{s.t.} \quad & \text{C1:} \quad T_k^{\text{exe}} + T_k^{\text{prop}} + T_k^{\text{tx}} \leq T_k + \alpha_k, \quad \forall k, \\ & \text{C5:} \quad \rho_k \leq P_k^{\max}, \quad \forall k. \end{aligned} \tag{28}$$

The data rate in (27) removes the cross-coupling impact of the allocated power to different users. Hence, without loss of optimality, (28) is solved for each ρ_k independently. The associated power allocation problem is:

$$\begin{aligned} \min_{\rho_k} \quad & T_k^{\text{tx}} \\ \text{s.t.} \quad & \text{C1:} \quad T_k^{\text{exe}} + T_k^{\text{prop}} + T_k^{\text{tx}} \leq T_k + \alpha_k, \quad \forall k, \\ & \text{C5:} \quad \rho_k \leq P_k^{\max}, \quad \forall k, \end{aligned} \tag{29}$$

in which α_k in the objective is replaced with T_k^{tx}. Note that minimizing T_k^{tx} is equivalent to maximizing $\frac{R_k}{D_k}$. Since R_k in (27) is increasing with ρ_k, the optimal solution of (29) is $\rho_k^\star = P_k^{\max}$. Note that feasibility of C1 is ensured by optimizing other variables.

Next, we deal with the binary optimization variable $\boldsymbol{\xi}$. We propose an exhaustive search over all possible values of $\boldsymbol{\xi}$ to avoid any performance loss due to non-convexity of (8), stemmed from binary $\boldsymbol{\xi}$. The number of all possible combinations of task placement decisions equals $|\mathcal{B}|^{|\mathcal{K}|}$ where $|\mathcal{B}| = \sum_n |\mathcal{B}_n|$. Thus, we solve (8) for $\boldsymbol{\alpha}$ and $\boldsymbol{\upsilon}$ for each task placement decision and select the decision that results in lowest $\sum_k \alpha_k$ as the optimal decision. Note that the exhaustive search may impose an excessive computational complexity. However, LTO is developed as a baseline for performance evaluation and it is not supposed to work in real-time.

The optimization problem for solving $\boldsymbol{\alpha}$ and $\boldsymbol{\upsilon}$ is:

$$\begin{aligned} \min_{\boldsymbol{\upsilon},\boldsymbol{\alpha}} \quad & \textstyle\sum_{k\in\mathcal{K}} \alpha_k \\ \text{s.t.} \quad & \text{C1-a:} \quad \frac{L_k}{\upsilon_k} \leq \tilde{T}_k + \alpha_k, \quad \forall k \in \mathcal{K} \\ & \text{C2:} \quad \textstyle\sum_{k\in\mathcal{K}_n} \upsilon_k \leq \Upsilon_n, \quad \forall n, \end{aligned} \tag{30}$$

where $\tilde{T}_k = T_k - T_k^{\text{prop}} - T_k^{\text{tx}}$ and $\mathcal{K}_n$ is the set of tasks to be executed at executing server n. Problem (30) is convex in both $\boldsymbol{\alpha}$ and $\boldsymbol{\upsilon}$. As a result, the KKT conditions determine the optimal solution. To derive the KKT conditions, we first write the Lagrangian function as follows:

$$\mathcal{L} = \sum_{k\in\mathcal{K}} \left(\alpha_k + \gamma_k \left(\frac{L_k}{\upsilon_k} - \tilde{T}_k - \alpha_k \right) - \eta_k \alpha_k - \mu_k \upsilon_k \right) + \sum_{n\in\mathcal{N}} \lambda_n \left(\sum_{k\in\mathcal{K}_n} \upsilon_k - \Upsilon_n \right). \tag{31}$$

By derivating the Lagrangian function with respect to α_k and υ_k we have:

$$\frac{\partial \mathcal{L}}{\partial \alpha_k} = 1 - \gamma_k - \eta_k = 0, \quad \forall k \in \mathcal{K}, \tag{32}$$

and

$$\frac{\partial \mathcal{L}}{\partial \upsilon_k} = -\gamma_k \frac{L_k}{\upsilon_k^2} - \mu_k + \lambda_n = 0, \quad \forall k \in \mathcal{K}_n. \tag{33}$$

In addition, the complementary slackness conditions are:

$$\gamma_k \left(\frac{L_k}{\upsilon_k} - \tilde{T}_k - \alpha_k\right) = 0, \quad \forall k \in \mathcal{K}, \tag{34}$$

$$\lambda_n \left(\sum_{k \in \mathcal{K}_n} \upsilon_k - \Upsilon_n\right) = 0, \quad \forall n \in \mathcal{N}, \tag{35}$$

$$\eta_k \alpha_k = 0, \quad \forall k \in \mathcal{K}, \tag{36}$$

$$\mu_k \upsilon_k = 0, \quad \forall k \in \mathcal{K}. \tag{37}$$

Constraint C1-a implies $\upsilon_k > 0$. Hence, from (37) we have $\mu_k = 0$ and condition (33) results in:

$$\upsilon_k = \sqrt{\frac{L_k}{\lambda_n}}, \quad \forall k \in \mathcal{K}_n, \tag{38}$$

which implies $\lambda_n > 0$. Thus, (35) gives:

$$\sum_{k \in \mathcal{K}_n} \upsilon_k = \Upsilon_n, \quad \forall n \in \mathcal{N}. \tag{39}$$

On the other hand, when (7) is infeasible, we get $\alpha_k > 0$. Thus, (36) leads to $\eta_k = 0$ and condition (32) results in $\gamma_k = 1$. As a result, from (34) we get:

$$\alpha_k = \frac{L_k}{\upsilon_k} - \tilde{T}_k, \quad \forall k \in \mathcal{K}. \tag{40}$$

Having $\alpha_k \geq 0$ and (38), the optimal non-negative variable is:

$$\alpha_k = [\sqrt{L_k \lambda_n} - \tilde{T}_k]^+, \quad \forall k \in \mathcal{K}_n, \tag{41}$$

wherein λ_n is found such that:

$$\sum_{k \in \mathcal{K}_n} \frac{L_k}{\tilde{T}_k + \alpha_k} = \Upsilon_n, \quad \forall n \in \mathcal{N}. \tag{42}$$

Then, the optimal values of α_k and υ_k are found as in (41) and (38), respectively. Having the optimal solution of (30) for all possible $\boldsymbol{\xi}$, the optimal solution of (8) is the solution with lowest objective of (30).

TABLE III: Simulation Setup.

Parameter	Value	Parameter	Value
L_k	10^6 CPU Cycles	$\delta_{(m,m')}$	10 ms
M	32 Antennas	Λ_n	10^{-28} [18]
D_k	0.1 Mbits	Path Loss	$128.1 + 37.6 \log Q$ [21]
Υ_n	10^9 CPU Cycles per Second [6]	U	4
$P_k^{\max}$	0.5 Watt	ISD	100 m
$B_{(m,m')}$	0.4 Gbps	W	20 MHz [21]
$B_{\mathrm{f},u}$	0.6 Gbps	Noise power	-150 dBm/Hz [21]

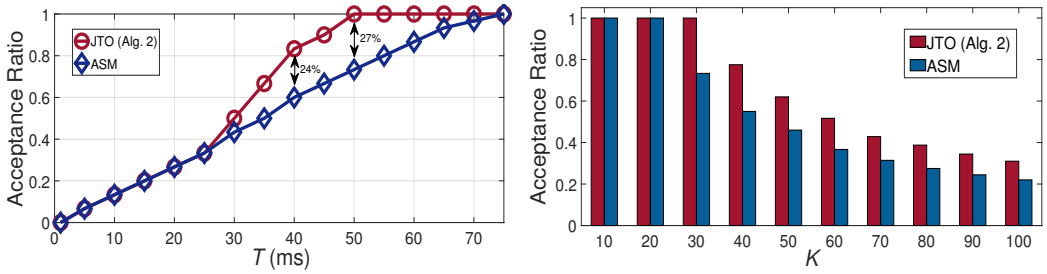
VII. Simulation Results

In this section, we evaluate the performance of JTO. The setup of the simulation is presented in Table III. We assume that $U = 4$ RRHs are placed with inter-site distance of 100 m and all users are served in an area of 100 m radius with a given user-RRH assignment. The nodes in the transport network are divided into three tiers based on their distance from UEs: the local tier, the regional tier, and the national tier. Although the number of serving nodes is very large, there are some distant nodes in each tier that impose a large propagation latency. Hence, we only incorporate the nodes with reasonable propagation latency in the transport network [7]. Network graph G consists of $N = 6$ nodes: $\bar{n}$ at the local tier with zero propagation latency, three nodes at the regional tier with relatively low propagation latency, and two distant nodes at the national tier. For simplicity of comparison, we assume that all nodes have the same computational capacity and all tasks are of the same size, load, and maximum acceptable latency, i.e., $D_k = D$, $L_k = L$, and $T_k = T$, $\forall k$. Moreover, we assume equal propagation latency and capacity for the network links. Note that the relatively low value of link capacity (0.4 Gbps) is the amount of capacity solely reserved for task offloading. Finally, the simulations are performed on a 3.30 GHz Core i5 CPU and 16 GB RAM.

Fig. 3 (a) reports the performance of the feasibility analysis in JTO, showing the acceptance ratio versus T. The acceptance ratio is defined as the ratio of accepted services by the feasibility analysis over the total number of the requested tasks. Note that the acceptance ratio increases by increasing T. This is due to the fact that the tasks with higher T need less transmit power and computational resources to be served. Moreover, for higher T, a larger number of nodes are available for task offloading. In addition, we solve (8) by the alternate search method (ASM), in which (8) is alternatively solved for each variable. Note that the sub-problem of $\boldsymbol{v}$ is solved by CVX and the sub-problem of $\boldsymbol{\xi}$ is solved by MOSEK (details are not provided due to space

limitation). The effectiveness of JTO against ASM is also shown in Fig. 3 (a). Note that for latencies smaller than 75 ms, JTO outperforms ASM. Moreover, the performance of both methods is identical for low values of T. This is due to the fact that the set of accessible NFV-enabled nodes for low values of T is restricted to $\bar{n}$ and therefore, JTO is not able to offload the tasks to more distant NFV-enabled nodes because their propagation latencies violate the E2E latency constraints.

The acceptance ratio of JTO for different number of tasks is shown in Fig. 3 (b). Since the amount of available resources is limited, the acceptance ratio is decreasing with the increase in the total number of tasks. Again, the superiority of JTO over ASM is observed.

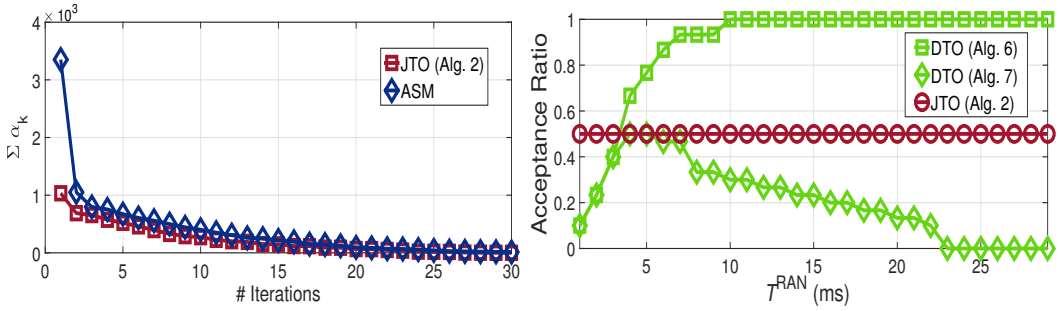


(a) Acceptance ratio vs. T for $K = 30$.

(b) Acceptance ratio vs. K for $T = 40$ ms.

Fig. 3: Acceptance ratio vs. T and K

The convergence of Algorithm 2 is shown in Fig. 4 (a). As expected, the sum of non-negative variables is decreasing in each iteration. Furthermore, Algorithm 2 converges faster than ASM, which stems from higher acceptance ratio of JTO.



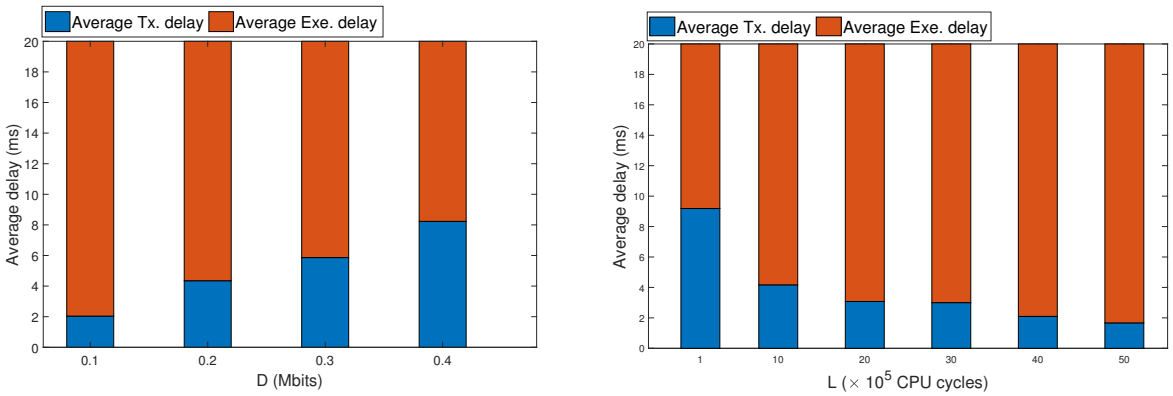
(a) Convergence of admission control algorithm for $T = 20$ ms and $K = 30$.

(b) Acceptance ratio of joint vs. disjoint methods in terms of T^{RAN} for $T = 30$ ms and $K = 30$ users.

Fig. 4: Convergence and acceptance ratio of the proposed methods.

The acceptance ratio of JTO is compared with DTO in Fig. 4 (b). The acceptance ratio of JTO and DTO is depicted for $T = 30$ ms. For DTO, we obtain the acceptance ratio for different values of $T^{\mathrm{RAN}} \in (0, T)$. Moreover, the acceptance ratio of the feasibility analysis in the transmit power allocation phase of DTO, i.e., Algorithm 6, is depicted. The acceptance ratio of DTO is increasing for small values of T^{RAN}, that is, the small values of T^{RAN} impose high rates on users, which is impossible due to either insufficient bandwidth or limited fronthaul capacity. On the other hand, for larger values of T^{RAN}, the acceptance ratio of Algorithm 6 is 1 but the task placement and computational resource allocation restricts the number of accepted tasks. Furthermore, JTO outperforms DTO in different values of T^{RAN}.

Fig. 5 (a) shows the average radio transmission latency, i.e., $\frac{1}{K}\sum_{k\in\mathcal{K}} T_k^{\mathrm{tx}}$, and the average execution latency of tasks, i.e., $\frac{1}{K}\sum_{k\in\mathcal{K}} T_k^{\mathrm{exe}}$ for different values of D given $T = 20$ ms. The average radio transmission latency increases by increasing D and subsequently the average execution latency is decreased to maintain the maximum acceptable latency. Therefore, it is inferred that JTO efficiently manages the transmit power and the computational resources. Similarly, according to Fig. 5 (b), the average execution latency increases by increasing L and subsequently this increase is compensated with lower radio transmission latency.



(a) Average radio transmission and execution latencies vs. D for $T = 20$ ms and $K = 30$.

(b) Average radio transmission and execution latencies vs. L for $T = 20$ ms and $K = 30$.

Fig. 5: Average radio transmission and execution latencies in JTO.

In Fig. 6, we assume there are three classes of tasks (each including 10 tasks) with three different maximum acceptable latencies, i.e., $T^{(1)} = 10$ ms, $T^{(2)}$ =50 ms, and $T^{(3)} = 100$ ms. The classes (1), (2), and (3) are considered as the sets of tasks with low, medium, and high latency requirements, respectively. Moreover, we assume there are three nodes (shown by rectangles):

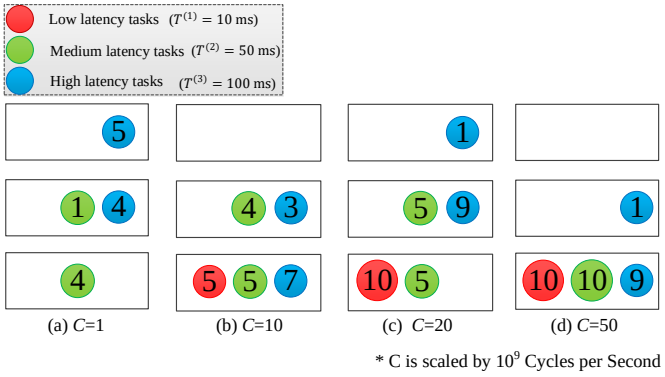


Fig. 6: Placement of the different classes of tasks at three different tiers of nodes for $K = 30$.

a local node (i.e., $\bar{n}$) with zero propagation latency, a regional node with 20 ms propagation latency, and a national node with 40 ms propagation latency. The propagation latencies are the summation of uplink and downlink propagation latencies. Fig. 6 shows the task placement for different values of the processing capacity of nodes $C = \Upsilon_n, \forall n$. When $C = 1$, none of the nodes is able to serve the tasks in class (1) due to their high resource utilization. However, the tasks in class (2) are mainly served at the local node and class (3) tasks are placed at regional and national nodes. When $C = 10$, some of the tasks in class (1) are placed at the local node. Moreover, some tasks in class (2) and (3) are served at the local node as well. Furthermore, the national node does not serve any task because JTO places the tasks at the nearest nodes in order to reduce the transmit power. When $C = 20$, more tasks in class (1) are served at the local node and the acceptance ratio reaches 1. Finally, when $C = 50$, almost all of the tasks are placed at the local node to reduce the transmit power consumption.

Table IV shows the acceptance ratio of each class for different values C. Note that the acceptance ratio of all classes is increased by increasing C. Moreover, the acceptance ratio of class (1) is lower than that of classes (2) and (3). The reason is twofold, one is due to high resource utilization by tasks of this class and another is due to limited number of available nodes for tasks with low latency requirement (only node $\bar{n}$ in this example).

Fig. 7 shows the acceptance ratio of LTO and JTO for different values of maximum acceptable latency T. Due to the high computational complexity of exhaustive search in LTO, we consider a simple network graph comprised of two nodes connected with a single link. Moreover, the total number of tasks $|\mathcal{K}|$ is 20. The acceptance ratio of both JTO and LTO is lower for larger computational loads. Meanwhile, the acceptance ratio of JTO is almost the same as LTO for

TABLE IV: Acceptance ratio of JTO for different task classes vs. processing capacity of nodes.

Computational capacity (10^9 CPU cycles/sec)	**Maximum acceptable latency (ms)**		
	$T^{(1)} = 10$	$T^{(2)} = 50$	$T^{(3)} = 100$
$C = 1$	0	0.5	0.9
$C = 10$	0.5	0.9	1
$C = 20$	1	1	1

different values of T and L.

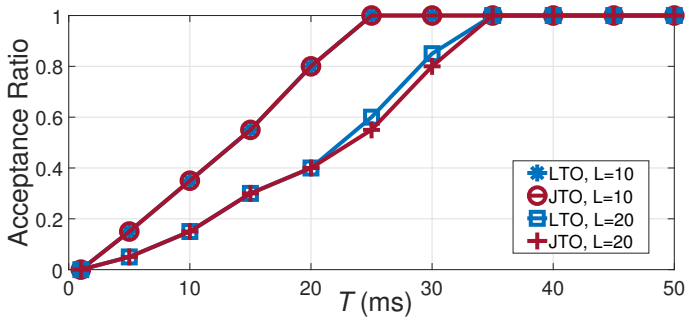


Fig. 7: Acceptance ratio of LTO and JTO vs. maximum acceptable latency.

VIII. Conclusions and Future Work

In this paper, we considered an energy-efficient task offloading problem under E2E latency constraints. We investigated the joint impact of radio transmission, propagation of tasks through the transport network, and execution of tasks on the experienced latency of tasks. Due to the non-convexity of the optimization problem, we decoupled the transmit power allocation from task placement and computational resource allocation. The transmit power allocation was solved by adopting CCP to convexify the sub-problem. The task placement and computational resource allocation were solved via our proposed heuristic method, which minimizes the sum of propagation and execution latencies. Furthermore, to ensure the feasibility of the optimization problem, we proposed a feasibility analysis that eliminates the tasks causing infeasibility. Simulation results showed the superiority of JTO over both DTO and ASM. The performance of DTO depended on the part of latency required to be met in the radio access network, i.e., T^{RAN}. However, JTO showed higher acceptance ratios for different values of T^{RAN}. As future work, we plan to incorporate task scheduling into JTO. Moreover, the investigation of an innovative solution that divides the required computational load of each task among several nodes will be an interesting future research activity.

REFERENCES

[1] B. Yi, X. Wang, K. Li, S. K. Das, and M. Huang, "A comprehensive survey of network function virtualization," *Computer Networks*, vol. 133, pp. 212–262, 2018.

[2] P. Mach and Z. Becvar, "Mobile edge computing: A survey on architecture and computation offloading," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1628–1656, 2017.

[3] ETSI, "Mobile Edge Computing (MEC); Framework and reference architecture," *ETSI Group Specification MEC 003*, 2016.

[4] H. Guo, J. Liu, and J. Zhang, "Computation offloading for multi-access mobile edge computing in ultra-dense networks," *IEEE Communications Magazine*, vol. 56, no. 8, pp. 14–19, 2018.

[5] W. Almughalles, R. Chai, J. Lin, and A. Zubair, "Task execution latency minimization-based joint computation offloading and cell selection for MEC-enabled HetNets," in *2019 28th Wireless and Optical Communications Conference (WOCC)*, pp. 1–5, IEEE, 2019.

[6] L. Yang, H. Zhang, M. Li, J. Guo, and H. Ji, "Mobile edge computing empowered energy efficient task offloading in 5G," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 7, pp. 6398–6409, 2018.

[7] T. X. Tran and D. Pompili, "Joint task offloading and resource allocation for multi-server mobile-edge computing networks," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 1, pp. 856–868, 2019.

[8] T. Q. Dinh, J. Tang, Q. D. La, and T. Q. Quek, "Offloading in mobile edge computing: Task allocation and computational frequency scaling," *IEEE Transactions on Communications*, vol. 65, no. 8, pp. 3571–3584, 2017.

[9] B. Yang, W. K. Chai, Z. Xu, K. V. Katsaros, and G. Pavlou, "Cost-efficient NFV-enabled mobile edge-cloud for low latency mobile applications," *IEEE Transactions on Network and Service Management*, vol. 15, no. 1, pp. 475–488, 2018.

[10] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Transactions on Networking*, vol. 24, no. 5, pp. 2795–2808, 2015.

[11] K. Zhang, Y. Mao, S. Leng, Q. Zhao, L. Li, X. Peng, L. Pan, S. Maharjan, and Y. Zhang, "Energy-efficient offloading for mobile edge computing in 5G heterogeneous networks," *IEEE access*, vol. 4, pp. 5896–5907, 2016.

[12] T. Li, C. S. Magurawalage, K. Wang, K. Xu, K. Yang, and H. Wang, "On efficient offloading control in cloud radio access network with mobile edge computing," in *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, pp. 2258–2263, IEEE, 2017.

[13] P. Zhao, H. Tian, C. Qin, and G. Nie, "Energy-saving offloading by jointly allocating radio and computational resources for mobile edge computing," *IEEE Access*, vol. 5, pp. 11255–11268, 2017.

[14] X. Zhang, Y. Mao, J. Zhang, and K. B. Letaief, "Multi-objective resource allocation for mobile edge computing systems," in *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, pp. 1–5, IEEE, 2017.

[15] J. Zhang, X. Hu, Z. Ning, E. C.-H. Ngai, L. Zhou, J. Wei, J. Cheng, and B. Hu, "Energy-latency tradeoff for energy-aware offloading in mobile edge computing networks," *IEEE Internet of Things Journal*, vol. 5, no. 4, pp. 2633–2645, 2017.

[16] C. Wang, F. R. Yu, C. Liang, Q. Chen, and L. Tang, "Joint computation offloading and interference management in wireless cellular networks with mobile edge computing," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 8, pp. 7432–7445, 2017.

[17] C. You, K. Huang, H. Chae, and B.-H. Kim, "Energy-efficient resource allocation for mobile-edge computation offloading," *IEEE Transactions on Wireless Communications*, vol. 16, no. 3, pp. 1397–1411, 2016.

[18] F. Zhou, Y. Wu, R. Q. Hu, and Y. Qian, "Computation rate maximization in UAV-enabled wireless-powered mobile-edge computing systems," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 9, pp. 1927–1941, 2018.

[19] A. Khalili, S. Zarandi, and M. Rasti, "Joint resource allocation and offloading decision in mobile edge computing," *IEEE Communications Letters*, vol. 23, no. 4, pp. 684–687, 2019.

[20] J. Zhang, W. Xia, F. Yan, and L. Shen, "Joint computation offloading and resource allocation optimization in heterogeneous networks with mobile edge computing," *IEEE Access*, vol. 6, pp. 19324–19337, 2018.

[21] W. Xia, J. Zhang, T. Q. Quek, S. Jin, and H. Zhu, "Power minimization-based joint task scheduling and resource allocation in downlink C-RAN," *IEEE Transactions on Wireless Communications*, vol. 17, no. 11, pp. 7268–7280, 2018.

[22] M.-H. Chen, M. Dong, and B. Liang, "Resource sharing of a computing access point for multi-user mobile cloud offloading with delay constraints," *IEEE Transactions on Mobile Computing*, vol. 17, no. 12, pp. 2868–2881, 2018.

[23] S. Li, N. Zhang, S. Lin, L. Kong, A. Katangur, M. K. Khan, M. Ni, and G. Zhu, "Joint admission control and resource allocation in edge computing for internet of things," *IEEE Network*, vol. 32, no. 1, pp. 72–79, 2018.

[24] J. Guo, Z. Song, Y. Cui, Z. Liu, and Y. Ji, "Energy-efficient resource allocation for multi-user mobile edge computing," in *GLOBECOM 2017-2017 IEEE Global Communications Conference*, pp. 1–7, IEEE, 2017.

[25] M.-H. Chen, B. Liang, and M. Dong, "Joint offloading decision and resource allocation for multi-user multi-task mobile cloud," in *2016 IEEE International Conference on Communications (ICC)*, pp. 1–6, IEEE, 2016.

[26] A. Al-Shuwaili and O. Simeone, "Energy-efficient resource allocation for mobile edge computing-based augmented reality applications," *IEEE Wireless Communications Letters*, vol. 6, no. 3, pp. 398–401, 2017.

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

[28] J. Liu, Y. Mao, J. Zhang, and K. B. Letaief, "Delay-optimal computation task scheduling for mobile-edge computing systems," in *2016 IEEE International Symposium on Information Theory (ISIT)*, pp. 1451–1455, IEEE, 2016.

[29] J. Li, H. Gao, T. Lv, and Y. Lu, "Deep reinforcement learning based computation offloading and resource allocation for MEC," in *2018 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1–6, IEEE, 2018.

[30] L. Lu, G. Y. Li, A. L. Swindlehurst, A. Ashikhmin, and R. Zhang, "An overview of massive MIMO: Benefits and challenges," *IEEE journal of selected topics in signal processing*, vol. 8, no. 5, pp. 742–758, 2014.

[31] J. W. Chinneck, *Feasibility and Infeasibility in Optimization*. Algorithms and Computational Methods, Springer, 2008.

[32] T. Lipp and S. Boyd, "Variations and extension of the convex–concave procedure," *Optimization and Engineering*, vol. 17, no. 4, pp. 263–287, 2016.

[33] CVX Research, Inc., "CVX: Matlab software for disciplined convex programming." http://cvxr.com/cvx, 2017.

[34] N. Mokari, F. Alavi, S. Parsaeefard, and T. Le-Ngoc, "Limited-feedback resource allocation in heterogeneous cellular networks," *IEEE Transactions on Vehicular Technology*, vol. 65, pp. 2509–2521, April 2016.

[35] S. Boyd and L. Vandenberghe, *Convex Optimization*. New York, NY, USA: Cambridge University Press, 2004.