Minimum Overhead Beamforming and Resource Allocation in D2D Edge Networks

Junghoon Kim, *Student Member, IEEE,* Taejoon Kim, *Senior Member, IEEE,* Morteza Hashemi, *Member, IEEE,* Christopher G. Brinton, *Senior Member, IEEE,* and David J. Love, *Fellow, IEEE*

Abstract—Device-to-device (D2D) communications is expected to be a critical enabler of distributed computing in edge networks at scale. A key challenge in providing this capability is the requirement for judicious management of the heterogeneous communication and computation resources that exist at the edge to meet processing needs. In this paper, we develop an optimization methodology that considers the network topology jointly with device and network resource allocation to minimize total D2D overhead, which we quantify in terms of time and energy required for task processing. Variables in our model include task assignment, CPU allocation, subchannel selection, and beamforming design for multiple-input multiple-output (MIMO) wireless devices. We propose two methods to solve the resulting non-convex mixed integer program: semi-exhaustive search optimization, which represents a "best-effort" at obtaining the optimal solution, and efficient alternate optimization, which is more computationally efficient. As a component of these two methods, we develop a novel coordinated beamforming algorithm which we show obtains the optimal beamformer for a common receiver characteristic. Through numerical experiments, we find that our methodology yields substantial improvements in network overhead compared with local computation and partially optimized methods, which validates our joint optimization approach. Further, we find that the efficient alternate optimization scales well with the number of nodes, and thus can be a practical solution for D2D computing in large networks.

Index Terms—Wireless edge networks, device-to-device (D2D) communications, multiple-input-multiple-output (MIMO), beamforming, network optimization.

I. INTRODUCTION

THE number of wireless devices is now over 8.6 billion, and with the advent of new 5G-and-beyond technologies, this is expected to grow to 12.3 billion by 2022 [2]. Many of these devices will be data-processing-capable nodes in the hands of users that facilitate rapidly growing data-intensive applications running at the network edge, e.g., social networking, video streaming, and distributed data analytics. Given the bursty nature of user demands, when certain devices are occupied with processing for computationally-intensive applications, e.g., face recognition, 3D modeling/gaming, and augmented/virtual

Junghoon Kim, Christopher G. Brinton, and David J. Love are with the Department of Electrical and Computer Engineering, Purdue University, West Lafayette, IN, 47907 USA (e-mail: kim3220@purdue.edu; cgb@purdue.edu; djlove@purdue.edu).

Taejoon Kim and Morteza Hashemi are with the Department of Electrical Engineering and Computer Science, University of Kansas, KS, 66045 USA (email: taejoonkim@ku.edu; mhashemi@ku.edu).
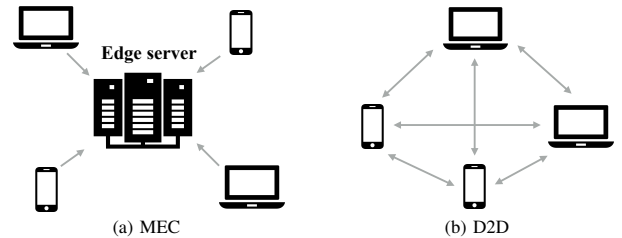

Fig. 1: High-level comparison between the topologies of (a) mobile edge computing (MEC) systems and (b) device-to-device (D2D) networks. MEC topology is typically fixed and predetermined, while D2D topology is not and can support offloading between devices.

reality (AR/VR), it may be desirable for them to offload their data to devices with underutilized resources [3]–[5]. Traditionally, cloud computing architectures, such as Amazon Web Services and Microsoft Azure, have been adopted for such data intensive applications, but the exponential rise in data generation at the edge is making centralized architectures infeasible for providing latency-sensitive quality of service at scale [2].

As a current trend in wireless networks is reducing cell sizes [6], many 5G networks will be dense with small cell coverage areas and networks composed of several smaller subsets [7]. Networks of small subnets combined with improved computational and storage capabilities of edge devices are enabling mobile edge computing (MEC) architectures. At a high level, MEC leverages radio access networks (RANs) to increase the amount of computing power located close to the end user, which enables the end user to offload computations (e.g., using a central processing entity) as shown in Fig. 1(a) [8]–[17]. In an MEC architecture, the edge servers have high-performance computing units which can process large amounts of computationally intensive tasks efficiently. This concept has been extended to "helper" edge server architectures as well, where devices with idle computation resources become (small) edge servers [18]–[23].

The current trend in distributed computing, though, is a migration to architectures that are more decentralized than MEC. This is due to the fact that all edge nodes can take part in data offloading at different times because of the advances in 5G communication technologies in conjunction with improved computational capabilities of individual devices. For this reason, device-to-device (D2D) network architectures (in Fig. 1(b)) that were previously studied in 4G LTE standards now hold the promise of providing distributed computing at scale [24].

Unlike the MEC system in Fig. 1(a), distributed computing in the D2D network of Fig. 1(b) will have more complicated

topology management needs that must be considered together with the management of device resources. From a computation perspective, the edge nodes that receive offloaded tasks must have a suitable strategy for allocating its central processing unit (CPU) and/or storage resources to the tasks. From a communication perspective, wireless transmissions between edge nodes will inevitably incur inter-channel interference due to concurrent data offloading. In contemporary wireless networks, devices are often equipped with multiple antennas to support multiple input multiple output (MIMO) communications, and the ability to communicate over different subchannels. Theoretically, having both multiple transmit and multiple receive antennas (i.e., a MIMO channel) provides an additional spatial dimension for communication and yields a degree-of-freedom gain [25], which can be leveraged to mitigate such inter-channel interference. This motivates interference management techniques that consider the joint effects of subchannel allocations, transmission powers, MIMO beamforming, and other device transmit resources.

The focus of this paper is on addressing these challenges. Specifically, we develop methodologies that jointly optimize computation and communication resources together with topology configuration in D2D wireless edge computing systems. These methodologies will aim to minimize the *overhead* incurred from communication and computation, measured in terms of time delay/latency and energy consumption incurred from processing tasks in the D2D network.

A. Related Work and Differentiation

We discuss related works on task offloading, resource management, and edge computing. We divide our analysis into two main categories: MEC and D2D.

1) MEC systems: Researchers have developed methods for resource management and offloading decision-making to maximize MEC system performance. Offloading decisions were thoroughly studied in [8], where management of device resources is assumed to be fixed. On the other hand, under the assumption that offloading decisions are given, studies have considered optimal allocations of CPU and subchannel resources [9], and have also considered these together with beamformer design for MIMO systems [14], [15]. Recently, offloading decisions have been considered together with management of resources in MEC systems such as CPU [10]–[13], subchannels [12], [17], transmit powers [12], [13], [16], and beamformer design [13].

Though we focus on D2D in this paper, as mentioned previously, newer MEC architectures allow idle devices in close proximity to be dedicated computing nodes. Therefore, optimization in MEC systems can be viewed as a special case of D2D networks, where offloading is restricted to specific devices unidirectionally. On the contrary, D2D networks allow multi-directional task offloading between devices. This requires additional optimization components to capture the multi-directional task offloading, CPU allocation across possibly multiple tasks at each device, and MIMO combiner design at receive devices, which are not considered in MEC systems.

2) D2D networks: Several prior works have focused on optimizing communication quality in D2D systems, where the objectives have been to maximize sum-rate [26]–[31], spectral efficiency [32], or signal-to-noise ratio (SINR) [33], with consideration of device and channel resources such as subchannels [26], [27], [29]–[31], transmit powers [27], [30], [31], and beamformer design for MIMO systems [32], [33]. For beamforming specifically, a coordinated beamforming strategy, such as the weighted minimum mean square error (WMMSE) approach, can be used to maximize standard communication measures (e.g., sum-rate) [34]. However, conventional beamforming strategies are not designed to consider time delay and energy consumption in a joint overhead metric for D2D. Furthermore, in D2D edge computing, the total network overhead is also impacted by other system parameters, such as subchannels, CPU allocation, and topology configuration.

Works on D2D in edge computing have primarily focused on D2D-enabled (or D2D-assisted) MEC systems where several helper nodes are available as dedicated nodes for computing together with the edge server. In this respect, within a fixed topology, [18] investigated energy minimization based on CPU and transmission power allocation, and [19] studied joint time and energy minimization based on CPU, subchannel, and transmission power allocation. On the other hand, for a given set of system resources, the strategy of topology reconfiguration was discussed to minimize total energy in [20]. Some recent works have addressed topology configuration together with the allocation of specific device resources such as CPU [21]–[23] and power [21], [22]. Advanced offloading strategies for topology configuration have also been introduced in vehicular fog computing [35], [36]. In [35], a task offloading method leveraging pricing-based matching was proposed to minimize the total network delay under utility constraints. In [36], an ADMM-based approach was proposed for energy minimization via partial offloading decisions subject to computing capability and latency constraints. Overall, we are not aware of any work that has addressed computation, communication, and topology configuration together in a unified optimization model for D2D edge computing, which is the focus of our paper.

B. Summary of Contributions

Compared to the related works discussed in Section I-A, the contributions of this paper are as follows:

- We formulate a unified optimization model for D2D edge computing networks that minimizes total network overhead, defined as the weighted sum of time and energy consumption required to process a given task. In doing so, we consider different dimensions of MIMO wireless for combating interference caused by concurrent data offloading, where each device is equipped with multiple antennas. Our model includes a framework for joint topology configuration, CPU allocation, subchannel allocation, and beamformer design for MIMO systems (Sections II and III).

- We propose two methods for minimizing the total network overhead in our model, which we refer to as semi-exhaustive search optimization and efficient alternate optimization. We compare these two methods in terms of optimality guarantees and computational complexity in solving our non-convex problem. While the semi-exhaustive search optimization can be viewed as a "best effort" to obtaining the optimal solution,

its complexity becomes problematic as the network size grows, which motivates the efficient alternate optimization (Section IV).

- In developing these methods, we study the decomposition of the optimization into several subproblems: topology design, CPU allocation, and beamformer design. In doing so, we develop a novel beamforming algorithm, minimum communication overhead beamforming (MCOB), that can be seen as a generalization of the WMMSE technique to utility functions incorporating both time and energy overhead. We prove that MCOB obtains the optimal beamformer for an MMSE receiver (Section IV).

- We conduct several numerical experiments to evaluate the performance of our network overhead optimization methodology. Our results show, for example, that our efficient alternate optimization algorithm can reduce the total overhead in D2D networks by 20%-30% compared to computation without offloading (Section V).

II. WIRELESS DEVICE-TO-DEVICE (D2D) NETWORK MODEL

In this section, we develop our models for computational tasks, wireless signals, and the allocation of network resources in D2D systems.

A. Task Model

We let $\mathcal{K} = \{1, 2, ..., K\}$ be the set of nodes in the D2D network, with a total of K nodes. Each node $k \in \mathcal{K}$ has a *task* to be completed, consisting of computational work involved in data processing, where the objective of the data processing is to perform a transformation from input to output data. In this paper, we adopt an indivisible task model [37], where a task is processed as a whole.[1] A task is considered to be *completed* when the input data is successfully processed to the desired output. In general, task completion requires computational resources including CPU, RAM, and storage. In this paper, similar to previous works [11]–[13], [21]–[23], we focus on CPU as the computation resource. In case of mobile devices, many of today's tasks require computation-intensive processing with high CPU requirements, such as 3D-gaming and location-based augmented/virtual reality (AR/VR) [3]–[5].

To quantify the complexity of the task for node k (which we will refer to as task k for brevity), we introduce the *task size* I_k (in bits), which is the length of the bit stream of input data consisting of task k. In other words, the bit stream of input data is represented as $\{0, 1\}^{I_k}$. Then, the *task workload* is denoted as $\mu_k I_k$ (in cycles), where μ_k (in cycles/bit) is the processing density, meaning how many CPU cycles are required to process a bit of data. That is, $\mu_k I_k$ represents total number of CPU cycles required to complete task k. The processing density μ_k depends on the application; for example, in the case of the audio signal detection in [38], since 500 cycles are required for processing 1 bit of data, μ_k is 500.

[1]The computation task model widely adopted in edge computing includes two categories: a divisible task model and an indivisible task model. The former supports fine-grained task partitions composed of multiple parallel segments, and the latter supports highly integrated or relatively simple tasks that cannot be partitioned for execution. In this paper, we focus on the latter.
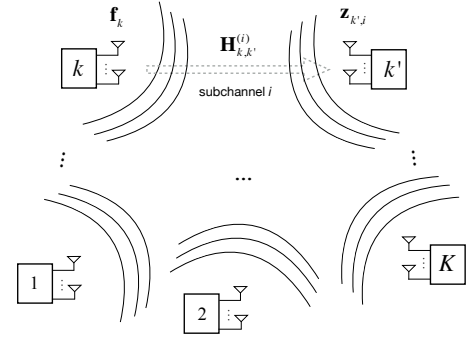

Fig. 2: Wireless device-to-device (D2D) network model among K nodes. Node k transmits with a beamformer $\mathbf{f}_k$ to receive node k' through subchannel i characterized as $\mathbf{H}_{k,k'}^{(i)}$, which is decoded with a receive combiner $\mathbf{z}_{k',i}$.

B. Signal Model

Fig. 2 demonstrates our wireless D2D network model among a set of K nodes. We assume that the nodes can transmit using multiple antennas on S subchannels, where the set of subchannels is denoted $\mathcal{S} = \{1, 2, ..., S\}$. Each node $k' \in \mathcal{K}$ receives a signal $\mathbf{y}_{k',i}$ through subchannel $i \in \mathcal{S}$ as

$$\mathbf{y}_{k',i} = \sum_{k=1}^{K} b_{k,i} \mathbf{H}_{k,k'}^{(i)} \mathbf{f}_k s_k + \mathbf{n}_{k',i} \in \mathbb{C}^{N_{k'}}, \qquad (1)$$

where $N_{k'}$ is the number of antennas of node k'. The scalar $s_k \in \mathbb{C}$ denotes the transmit signal sent by node k with unit power $\mathbb{E}[|s_k|^2] = 1$, where s_k can be understood as a single channel use of a Gaussian codeword vector that is encoded with I_k bits per channel use. The vector $\mathbf{f}_k \in \mathbb{C}^{N_k}$ is the *transmit beamformer* of node k with transmission power constraint P_k, i.e., $||\mathbf{f}_k||_2^2 \leq P_k$. Also, the matrix $\mathbf{H}_{k,k'}^{(i)} \in \mathbb{C}^{N_{k'} \times N_k}$ denotes a *multiple-input multiple-output (MIMO) channel* from transmit node k to receive node k' through subchannel i. The noise vector $\mathbf{n}_{k',i} \in \mathbb{C}^{N_{k'}}$ is assumed to be complex additive Gaussian noise with zero mean and identity covariance matrix scaled by the noise power σ^2, i.e., $\mathbf{n}_{k',i} \sim \mathcal{CN}(\mathbf{0}, \sigma^2 \mathbf{I})$. The *subchannel allocation variable* $b_{k,i} \in \{0, 1\}$ denotes whether transmit node k uses subchannel i for transmission. We assume that the transmit node k uses only one subchannel for transmission; if $b_{k,i} = 1$, then $b_{k,j} = 0 \ \forall j \neq i$.

At receive node k' on subchannel i, we consider a linear receive combiner $\mathbf{z}_{k',i} \in \mathbb{C}^{N_{k'}}$ so that the estimated value $\hat{y}_{k',i}$ is given by

$$\hat{y}_{k',i} = \mathbf{z}_{k',i}^H \mathbf{y}_{k',i}, \qquad (2)$$

where the superscript H denotes the conjugate transpose.

C. Task and Resource Allocation

The assignment of tasks to either offloading or local processing determines the D2D network topology. Constraints on how subchannels and processing resources are allocated must be specified based on these assignments.

1) Task assignment: Each task k can be either processed locally at node k or offloaded to another node k' for processing. We define $a_{k,k'} \in \{0, 1\}$ as the *task assignment variable* of whether task k is assigned to node k' for $k, k' \in \mathcal{K}$. If $a_{k,k} = 1$, then we have *local processing* of task k at node k. On the other hand, if $a_{k,k'} = 1$ for some $k' \neq k$, then we have *offloaded*

processing where task k is offloaded from k to k' and processed at node k'. The set of task assignments is denoted by

$$\mathcal{A} = \{(k,k') : a_{k,k'} = 1 \ \forall k, k' \in \mathcal{K}\}. \tag{3}$$

Since each task should be processed as a whole, task k should be assigned to only one node, i.e.,

$$\sum_{k'=1}^{K} a_{k,k'} = 1 \ \forall k. \tag{4}$$

For efficient offloading decisions, it is important to consider the channel matrices between nodes, as well as the subchannels and available computing capabilities of nodes. [2]

2) Subchannel allocation: The task assignment specifies the configuration of how the K nodes communicate with each other. Therefore, the subchannel allocation variable $b_{k,i}$ is related to task assignment variable $a_{k,k'}$ as

$$\sum_{i=1}^{S} b_{k,i} = \begin{cases} 1 \ \text{for} \ k \ \text{with} \ a_{k,k} = 0 \\ 0 \ \text{for} \ k \ \text{with} \ a_{k,k} = 1. \end{cases} \tag{5}$$

$a_{k,k} = 0$ implies node k is a *transmit node*, because task k is not locally processed at node k, implying transmission to another node. In this case, transmit node k uses one of the subchannels for transmission, i.e., $\sum_{i=1}^{S} b_{k,i} = 1$. On the other hand, if node k is not a transmit node, then $a_{k,k} = 1$ and there is no subchannel allocation for node k, i.e., $\sum_{i=1}^{S} b_{k,i} = 0$.

Each of the S subchannels is assumed to have equal and non-overlapping bandwidth of width W. Consider, however, the case that node k' receives multiple tasks from multiple transmit nodes. If same subchannel i is used by these transmitters, the receive node must jointly decode the data of tasks, which leads to degraded decoding performance. Therefore, in this paper, we follow prior work and assume that the transmit nodes that transmit to the same receive node use different subchannels [17]. In other words, for each receive node k', we restrict the number of transmitters on subchannel i according to

$$\sum_{k \in \mathcal{A}_{k'}} b_{k,i} \le 1 \ \forall k', i, \tag{6}$$

where $\mathcal{A}_{k'}$ denotes the set of transmit nodes that transmit to the receive node k' given by

$$\mathcal{A}_{k'} = \{k : a_{k,k'} = 1 \ \forall k \in \mathcal{K} \ \text{and} \ k \ne k'\}. \tag{7}$$

3) Computational resource allocation: Consider that node k' has multiple tasks to complete (its own and/or those offloaded to it). Its *computational resource* (CPU) $F_{k'}$ will be shared across these multiple tasks, where $F_{k'}$ (in cycles/sec or Hz) denotes the available CPU of node k'. We define the amount of CPU resource of node k' allocated to task k as $F_{k,k'}$, which is subject to the constraints

$$\sum_{k=1}^{K} F_{k,k'} \le F_{k'} \ \forall k', \tag{8}$$

$$F_{k,k'} = 0 \ \text{if} \ a_{k,k'} = 0, \quad F_{k,k'} \ge 0 \ \forall k, k'. \tag{9}$$

In (8), the total CPU resource allocated cannot exceed the available CPU resource for each node k'. In (9), $a_{k,k'} = 0$ implies that task k has not been assigned to node k', so no CPU resources will be allocated to task k. In addition, the allocated CPU $F_{k,k'}$ is restricted to a positive real value.

III. D2D Network Optimization Model

In this section, we formulate the optimization problem for minimizing D2D network task completion overhead. We define the total network overhead as a cost function to be minimized, consisting of both computation and communication overhead.

A. Computation Overhead

We first define the computation overhead associated with node k offloading to node k'. Based on the models from Section II, we can compute the *computation time* $T_{\text{comp}}(k,k')$ (in seconds) of task k computed at node k' according to

$$T_{\text{comp}}(k,k') = \frac{\mu_k I_k}{F_{k,k'}}. \tag{10}$$

The *computation energy consumption* $E_{\text{comp}}(k,k')$ (in Joules) can be computed as

$$E_{\text{comp}}(k,k') = \kappa_{k'} F_{k,k'}^2 \mu_k I_k, \tag{11}$$

where $\kappa_{k'}$ is the energy coefficient (in Joules $\cdot$ seconds^2/cycles^3) of node k' that depends on the processor chip architecture [39]. Here, $\kappa_{k'} F_{k,k'}^2$ denotes the energy consumption per cycle (in units of Joules/cycle).

We define the *computation overhead* $Y_{\text{comp}}(k,k')$ as the weighted sum of time and energy consumption, given by

$$Y_{\text{comp}}(k,k') = (1 - \beta_k) T_{\text{comp}}(k,k') + \beta_k E_{\text{comp}}(k,k')$$
$$= \left((1 - \beta_k)\frac{1}{F_{k,k'}} + \beta_k \kappa_{k'} F_{k,k'}^2\right) \mu_k I_k, \tag{12}$$

where $\beta_k \in [0,1]$ is a demand overhead factor. From (10) and (11), note that the time consumption $T_{\text{comp}}(k,k')$ and energy consumption $E_{\text{comp}}(k,k')$ have tradeoff relationship with respect to computation resources: as more computation resources $F_{k,k'}$ are used, computation time $T_{\text{comp}}(k,k')$ decreases while computation energy $E_{\text{comp}}(k,k')$ increases. The overhead factor β_k trades off the importance of these two factors, and should be determined by the requirement of task k. Note that $Y_{\text{comp}}(k,k)$ gives the local computation overhead in the case that task k is locally processed at node k.

B. Communication Overhead

We now define the communication overhead associated with transmission of a task from node k to k'. When $k \ne k'$, we can write the *signal to interference plus noise ratio* (SINR) from node k to node k' on subchannel i as

$$\text{SINR}_{k,k'}^{(i)} = \frac{b_{k,i}\left|\mathbf{z}_{k',i}^H \mathbf{H}_{k,k'}^{(i)} \mathbf{f}_k\right|^2}{\sum\limits_{\ell \ne k}^{K} b_{\ell,i}\left|\mathbf{z}_{k',i}^H \mathbf{H}_{\ell,k'}^{(i)} \mathbf{f}_\ell\right|^2 + \sigma^2 \|\mathbf{z}_{k',i}\|_2^2}, \tag{13}$$

where all other transmit nodes $\ell \ne k$ using subchannel i are interferences to the data stream of node k on subchannel i.

Assuming perfect channel state information (CSI), we can write the *maximum achievable data rate* $R_{k,k'}^{(i)}$ (in bits/second) from node k to node k' on subchannel i as

$$R_{k,k'}^{(i)} = W \log_2\big(1 + \text{SINR}_{k,k'}^{(i)}\big), \tag{14}$$

where W is the bandwidth of each frequency subchannel. Then, the total maximum achievable data rate from node k to node k' over all subchannels is

$$R_{k,k'} = \sum_{i=1}^{S} R_{k,k'}^{(i)}. \tag{15}$$

When node k is a transmitter, by (5), only one subchannel is active. In other words, when $b_{k,i} = 1$, $b_{k,j} = 0$ for $j \neq i$, leading to $R_{k,k'}^{(j)} = 0$. Letting $i(k)$ be the active subchannel for node k, i.e., satisfying $b_{k,i(k)} = 1$, the achievable rate is

$$R_{k,k'} = W \log_2 \left(1 + \frac{\left| \mathbf{z}_{k',i(k)}^H \mathbf{H}_{k,k'}^{(i(k))} \mathbf{f}_k \right|^2}{\sum_{\ell \neq k}^{K} b_{\ell,i(k)} \left| \mathbf{z}_{k',i(k)}^H \mathbf{H}_{\ell,k'}^{(i(k))} \mathbf{f}_\ell \right|^2 + \sigma^2 \left\| \mathbf{z}_{k',i(k)} \right\|_2^2} \right). \tag{16}$$

Given the data rate, we can compute the *communication time* $T_{\text{comm}}(k,k')$ (in seconds) for offloading task k to node k' as

$$T_{\text{comm}}(k,k') = \frac{I_k}{R_{k,k'}}. \tag{17}$$

The *communication energy consumption* for node k corresponding to the link from k to k' is

$$E_{\text{comm}}(k,k') = (\|\mathbf{f}_k\|_2^2 + P_{\text{c}}) \frac{I_k}{R_{k,k'}}, \tag{18}$$

where P_{c} is the circuit power including power dissipations in the transmit filter, mixer, and digital-to-analog converter, which are independent of the actual transmit power $\|\mathbf{f}_k\|_2^2$.

With these expressions for $T_{\text{comm}}(k,k')$ and $E_{\text{comm}}(k,k')$, the *communication overhead* $Y_{\text{comm}}(k,k')$ is defined with respect to the overhead factor β_k as

$$Y_{\text{comm}}(k,k') = (1 - \beta_k) T_{\text{comm}}(k,k') + \beta_k E_{\text{comm}}(k,k')$$
$$= (1 - \beta_k + \beta_k \|\mathbf{f}_k\|_2^2 + \beta_k P_{\text{c}}) \frac{I_k}{R_{k,k'}}. \tag{19}$$

We allocate the same β_k to the computation and communication overhead definitions (in (12) and (19)) because β_k is the weighting factor between time delay and energy consumption for task k, which intuitively should be the same for both types of overhead.[3] There is a tradeoff between $T_{\text{comm}}(k,k')$ and $E_{\text{comm}}(k,k')$ with respect to the transmit power $\|\mathbf{f}_k\|_2^2$: as more power $\|\mathbf{f}_k\|_2^2$ is applied, $T_{\text{comm}}(k,k')$ decreases due to the increasing data rate $R_{k,k'}$ in (16), while $E_{\text{comm}}(k,k')$ increases because $\|\mathbf{f}_k\|_2^2 / R_{k,k'}$ increases.

C. Total Network Overhead

Recall that there are two possibilities for task k: (i) local processing, i.e., $a_{k,k} = 1$, and (ii) offloaded processing, i.e., $a_{k,k'} = 1$ for some $k' \neq k$. Local processing only incurs computation overhead $Y_{\text{comp}}(k,k)$ while offloaded processing incurs both communication and computation overhead, $Y_{\text{comm}}(k,k') + Y_{\text{comp}}(k,k')$. With this, for a given D2D network topology configuration, we can write the *total network overhead* to complete all tasks in the network as

---

[3]If different overhead factors are preferred due to different resource restrictions on communication and computation, our methodology can be easily extended by defining β_k^{comp} in (12) and β_k^{comm} in (19).

$$Y_{\text{total}} = \sum_{k=1}^{K} \Bigg(a_{k,k} Y_{\text{comp}}(k,k) + \sum_{k' \neq k}^{K} a_{k,k'} \Big(Y_{\text{comm}}(k,k') + Y_{\text{comp}}(k,k') \Big) \Bigg). \tag{20}$$

D. Optimization Formulation

We now formulate the problem of jointly optimizing the D2D network parameters to achieve the minimum total network overhead Y_{total}. The degrees of freedom available are the task assignments $\{a_{k,k'}\}$, computational resource allocations $\{F_{k,k'}\}$, subchannel allocations $\{b_{k,i}\}$, and beamformer design variables involving transmit beamformers $\{\mathbf{f}_k\}$ and receive combiners $\{\mathbf{z}_{k',i}\}$. The optimization problem is given by:

$$\underset{\{\mathbf{f}_k\}, \{\mathbf{z}_{k',i}\}, \{F_{k,k'}\}, \{a_{k,k'}\}, \{b_{k,i}\}}{\text{minimize}} \quad Y_{\text{total}} \text{ in (20)} \tag{21}$$

subject to $\sum_{k'=1}^{K} a_{k,k'} = 1 \; \forall k, \quad a_{k,k'} \in \{0,1\} \; \forall k,k',$ (22)

$$\sum_{i=1}^{S} b_{k,i} = \begin{cases} 1 & \forall k \text{ with } a_{k,k} = 0 \\ 0 & \forall k \text{ with } a_{k,k} = 1, \end{cases} \tag{23}$$

$$\sum_{k \in \mathcal{A}_{k'}} b_{k,i} \leq 1 \; \forall k', i, \quad b_{k,i} \in \{0,1\} \; \forall k, i, \tag{24}$$

$R_{k,k'}$ defined in (16), $\quad \|\mathbf{f}_k\|_2^2 \leq P_k \; \forall k,$ (25)

$$\sum_{k=1}^{K} F_{k,k'} \leq F_{k'} \; \forall k', \tag{26}$$

$$F_{k,k'} = 0 \text{ if } a_{k,k'} = 0, \quad F_{k,k'} \geq 0 \; \forall k, k' \tag{27}$$

Constraints (22)-(24) and (26)-(27) account for task assignment, subchannel allocation, and CPU allocation requirements, which were described in Section II-C. (25) captures the constraint for the transmission power budget P_k of an individual node. Note that there is no constraint on $\{\mathbf{z}_{k',i}\}$ such as a maximum magnitude restriction because the data rate $R_{k,k'}$ is not affected by the magnitude of $\mathbf{z}_{k',i}$.

Assuming all nodes have N antennas, meaning that $N_k = N$ for all k, the optimization is a mixed integer program (MIP) with $K(N + NS + K)$ non-integer variables from $\{\mathbf{f}_k\}$, $\{\mathbf{z}_{k',i}\}$, $\{F_{k,k'}\}$, and $K(K + S)$ integer variables from $\{a_{k,k'}\}$, $\{b_{k,i}\}$. The function $Y_{\text{comm}}(k,k')$ is non-convex with respect to $\{\mathbf{f}_k\}$ and $\{\mathbf{z}_{k',i}\}$, which makes the problem a non-convex MIP. Existing solvers for non-convex MIPs do not scale well with the number of variables [40], and even in a relatively small D2D setting with $K = 20$ nodes, $S = 5$ subchannels, and $N = 10$ antennas, our problem has already more than 2000 variables. In Section IV, we turn to addressing the challenge of solving this optimization at scale.

E. D2D Network Optimization Assumptions

A few assumptions made on the D2D model in this section are noteworthy. First, although the network states will be dynamic over time, we assume a quasi-static scenario with K active nodes and fixed channels during one codeword block, similar to previous works [9], [12]–[15], [18], [19], [21], [22]. The algorithms we develop for solving the optimization (21)-(27) in Section IV could then be applied to each quasi-static scenario as the number of nodes and channel conditions change, or at some suitable time interval. Related to this, we focus

our optimization on the tasks generated at nodes in a single time period. A straightforward way to extend this approach to a dynamic task generation scenario is through queuing, where new tasks are queued at nodes and processed by the optimization in a series of frames. We provide an experiment on this in Appendix D-C.

Second, we consider operator-assisted D2D networks [41]–[43], where a network operator functions as a centralized controller for resource management. The network operator, e.g., a base station (BS), executes our algorithms based on knowledge of the required network information, such as task size, CSI, and availability of subchannels and CPUs. This aligns with the current framework of cellular systems found in practice, where mobile devices typically have their radio access and resource allocation controlled through information sharing with the BS [44]. Our methodology can thus be seen as enabling distributed task processing in the data plane via centralized optimization for variable design in the control plane. The tasks generated by the nodes are processed in a distributed way, i.e., through offloading between devices or local processing, while the network operator will solve the optimization for orchestrating communication and computation resources.

Remark 1. *The BS, as a network operator, can acquire the CSI of devices through standard protocols of existing cellular network architectures. First, for CSI estimation at each device, concurrent channel estimation from multiple transmit devices can be conducted via well-established frameworks in the Long-Term Evolution (LTE) [44] and 5G new radio (5G-NR) [45] standards, using UE-specific reference signals. Since each device's wireless channel is temporally correlated in practice, CSI can be estimated with only a few parameters via channel tracking techniques [46]–[48]. Second, for CSI acquisition at the BS, the BS can obtain measures such as channel quality information (CQI), precoding matrix indicator (PMI), and rank indicator (RI) from the devices, without receiving entire channel matrices, through standard CSI reporting frameworks [44]. Since the number of CSI bits received at the BS would be very small compared to the sizes of modern computing tasks, the communication overhead for CSI acquisition can be safely ignored in the optimization. Although the CSI acquired at the BS will be imperfect, we will demonstrate in Section V that the effect of imperfect CSI (e.g., refer to [49]–[51]) has only a small effect on our proposed network overhead optimization.*

Third, we do not consider the process of transferring the computation result of an offloaded task back to the source node, similar to previous works, e.g., [8], [52]. This assumption is reasonable since for many applications (e.g., facial recognition), the size of the output data (e.g., the recognized individuals) is in general much smaller than the size of the input data (e.g., the original full images). Therefore, we consider that the output data is negligible in size compared with the task so that it can be transferred through the network with minimal load.

IV. Optimization Algorithms

In this section, we develop two methods for solving the minimum overhead optimization problem (21)-(27). The first method, semi-exhaustive search, provides a best-effort attempt to obtain the optimal solution, but has exponential complexity. The second method, efficient alternate optimization, reduces the complexity to polynomial time, for which we use semi-exhaustive search as an optimality benchmark. As a component of these two methods, we will derive a new algorithm for MIMO beamforming (Section IV-A2) which minimizes communication overhead in D2D wireless edge networks.

A. Semi-Exhaustive Search Optimization

Given the task assignments $\{a_{k,k'}\}$ and subchannel allocations $\{b_{k,i}\}$ variables are binary, an intuitive approach to solving the optimization is to exhaustively search through all of their possibilities, so long as the search space is not prohibitively large. Then, for each possibility, we can solve for the non-integer variables $\{\mathbf{f}_k\}$, $\{\mathbf{z}_{k',i}\}$, and $\{F_{k,k'}\}$. We refer to this method as *semi-exhaustive search*. The overall procedure is described in Algorithm 1: each choice of $\{a_{k,k'}\}$ and $\{b_{k,i}\}$ satisfying constraints (22)-(24) is considered. For given task assignments $\{a_{k,k'}\}$, we solve the CPU allocation problem for the processing resources $\{F_{k,k'}\}$, which is a convex problem. In addition, for fixed task assignments $\{a_{k,k'}\}$ and subchannel allocations $\{b_{k,i}\}$, we solve the problem with respect to the beamformers $\{\mathbf{f}_k\}$ and combiners $\{\mathbf{z}_{k',i}\}$, which is a beamformer design problem. We will develop solutions to these two problems in the following.

1) CPU allocation: With task assignments $\{a_{k,k'}\}$ determined, the optimization problem (21)-(27) with respect to CPU allocations $\{F_{k,k'}\}$ can be reduced to

$$\underset{\{F_{k,k'}\}}{\text{minimize}} \quad \sum_{k'=1}^{K}\sum_{k=1}^{K} a_{k,k'} Y_{\text{comp}}(k,k') \qquad (28)$$

$$\text{subject to} \qquad \text{Constraints } (26)-(27) \qquad (29)$$

The problem can be decomposed into K independent subproblems: each node can allocate its own CPU regardless of the others. For each node $k' \in \mathcal{K}$, the optimization problem is given as

$$\underset{F_{k,k'} \; \forall k}{\text{minimize}} \quad \sum_{k=1}^{K} a_{k,k'}\left((1-\beta_k)\frac{1}{F_{k,k'}} + \beta_k \kappa F_{k,k'}^2\right)\mu_k I_k \quad (30)$$

$$\text{subject to} \quad \sum_{k=1}^{K} F_{k,k'} \leq F_{k'}, \quad F_{k,k'} \geq 0 \; \forall k, \qquad (31)$$

$$F_{k,k'} = 0 \text{ if } a_{k,k'} = 0 \qquad (32)$$

Note that $Y_{\text{comp}}(k,k')$ is convex with respect to $\{F_{k,k'}\}$ (since all parameters in $Y_{\text{comp}}(k,k')$ are positive) and the constraints (26)-(27) are also convex. Therefore, optimization (28)-(29) is convex. The decomposed subproblem (30)-(32) for each k' is also a convex problem that can be solved accordingly.

2) Beamformer design: With task assignments $\{a_{k,k'}\}$ and subchannel allocations $\{b_{k,i}\}$ determined, the optimization problem (21)-(27) with respect to the beamformer design variables $\mathbf{f}_k$ and $\mathbf{z}_{k',i} \; \forall k, k' \in \mathcal{K}, i \in \mathcal{S}$, can be reduced to

$$\underset{\mathbf{f}_k, \; \mathbf{z}_{k',i} \; \forall k,k'\in\mathcal{K}, \; i\in\mathcal{S}}{\text{minimize}} \quad \sum_{k=1}^{K}\sum_{k'\neq k}^{K} a_{k,k'} Y_{\text{comm}}(k,k') \qquad (33)$$

$$\text{subject to} \qquad \text{Constraint } (25) \qquad (34)$$

Algorithm 1 Semi-exhaustive search optimization

1: **Initialization.** Set $\mathcal{G}^\star = \emptyset$ and $Y^\star_{\text{total}} = \Upsilon$ (e.g., $\Upsilon = 10^5$).
2: **repeat**
3:      Generate new $\{a_{k,k'}\}$ and $\{b_{k,i}\}$ which satisfy (22)-(24).
4:      *CPU allocation:* Solve for $\{F_{k,k'}\}$ with $\{a_{k,k'}\}$ from (30)-(32)
5:      *Beamformer design:* Solve for $\{\mathbf{f}_k\}$ and $\{\mathbf{z}_{k',i}\}$ with $\{a_{k,k'}\}$ and $\{b_{k,i}\}$ from Algorithm 2.
6:      Calculate Y_{total} in (20) with the solution set $\mathcal{G} = \{\{a_{k,k'}\}, \{b_{k,i}\}, \{\mathbf{f}_k\}, \{\mathbf{z}_{k',i}\}, \{F_{k,k'}\}\}$.
7:      **if** $Y_{\text{total}} < Y^\star_{\text{total}}$ **then**
8:          Update $Y^\star_{\text{total}} = Y_{\text{total}}$ and $\mathcal{G}^\star \leftarrow \mathcal{G}$.
9:      **end if**
10: **until** There is no possible case of $\{a_{k,k'}\}$ and $\{b_{k,i}\}$
11: **return** $\{a_{k,k'}\}, \{b_{k,i}\}, \{\mathbf{f}_k\}, \{\mathbf{z}_{k',i}\}, \{F_{k,k'}\}$ in $\mathcal{G}^\star$

We refer to this as the *minimum communication overhead beamforming* (MCOB) problem. Conventionally, objective functions in beamforming resource allocation problems take the form of sum rate or sum harmonic rate utility functions [53]. In our D2D setting, the objective instead becomes the weighted sum of time and energy consumption for transmission.

We are interested in determining the variables $\mathbf{f}_k$ and $\mathbf{z}_{k',i}$ related to active data streams, i.e., for k, k', and i with $a_{k,k'} = 1$ and $b_{k,i} = 1$. Denote the set of all transmit nodes as $\mathcal{K}_{\text{Tx}} = \bigcup_{k' \in \mathcal{K}} \mathcal{A}_{k'} \subset \mathcal{K}$ from (7). Since each node $k \in \mathcal{K}_{Tx}$ offloads to one k' on one subchannel i, we index this datastream as the tuple (k, k', i).[4] Our problem can be then rewritten as

$$\underset{\mathbf{f}_k, \mathbf{z}_{k',i}}{\text{minimize}} \sum_{\forall k \in \mathcal{K}_{\text{Tx}}} (1 - \beta_k + \beta_k \|\mathbf{f}_k\|_2^2 + \beta_k P_c) \frac{I_k}{R_{k,k'}} \quad (35)$$

$$\text{subject to} \quad \|\mathbf{f}_k\|_2^2 \leq P_k \quad \forall k \in \mathcal{K}_{\text{Tx}} \quad (36)$$

This communication overhead minimization problem is non-convex and hard to solve due to the logarithm term in the data rate $R_{k,k'}$ in (16). However, if the beamformers $\{\mathbf{f}_k\}$ are fixed,[5] minimizing (35) leads to the well known minimum mean square error (MMSE) receiver. If we restrict ourselves to using MMSE receiver, we can transform the data rate into a quadratic form with the following lemma.

Lemma 1. *With an MMSE-designed receiver, the data rate in (16) can be represented in quadratic form as*

$$R_{k,k'} = \max_{\mathbf{z}_{k',i}, w_k} u_k(\{\mathbf{f}_k\}, \mathbf{z}_{k',i}, w_k), \quad (37)$$

where

$$u_k(\{\mathbf{f}_k\}, \mathbf{z}_{k',i}, w_k) = -w_k^{-1} e_k^{\text{mse}}(\{\mathbf{f}_k\}, \mathbf{z}_{k',i}) - \log w_k + \sigma^2, \quad (38)$$

$w_k \in \mathbb{R}^+$ *is an auxiliary variable, and the term e_k^{mse} is the MSE of receive node k' given by*

$$e_k^{\text{mse}}(\{\mathbf{f}_k\}, \mathbf{z}_{k',i}) = (1 - \mathbf{z}_{k',i}^H \mathbf{H}_{k,k'}^{(i)} \mathbf{f}_k)^H (1 - \mathbf{z}_{k',i}^H \mathbf{H}_{k,k'}^{(i)} \mathbf{f}_k)$$
$$+ \mathbf{z}_{k',i}^H (\sum_{\ell \neq k}^K b_{\ell,i} \mathbf{H}_{\ell,k'}^{(i)} \mathbf{f}_\ell \mathbf{f}_\ell^H \mathbf{H}_{\ell,k'}^{(i)H} + \sigma^2 \mathbf{I}) \mathbf{z}_{k',i}. \quad (39)$$

---

[4]Once $\{a_{k,k'}\}$ and $\{b_{k,i}\}$ are determined, the tuple (k, k', i) is specified by k and can be written as $(k, k'(k), i(k))$. For convenience, we are omitting the dependency of k' and i on k.

[5]In this case, the notation $\{\mathbf{f}_k\}$ is short for $\{\mathbf{f}_k\}_{k \in \mathcal{K}_{\text{Tx}}}$ denoting all variables $\mathbf{f}_k$ with $k \in \mathcal{K}_{\text{Tx}}$. Throughout the paper, the context will make the distinction clear. The same simplification is applied for $\{\mathbf{z}_{k',i}\}$, $\{w_k\}$, $\{\lambda_k\}$, and $\{\gamma_k\}$.

The proof follows from Theorem 1 in [34]. Since u_k in (38) is concave with respect to each of the variables $\{\mathbf{f}_k\}$, $\mathbf{z}_{k',i}$ and w_k, the optimal solution to (37) is

$$\mathbf{z}_{k',i}^\star = \mathbf{J}_k^{-1} \mathbf{H}_{k,k'}^{(i)} \mathbf{f}_k, \quad (40)$$

$$w_k^\star = e_k^{\text{mse}}(\{\mathbf{f}_k\}, \mathbf{z}_{k',i}^\star), \quad (41)$$

where $\mathbf{J}_k = \sum_{\ell=1}^K b_{\ell,i} \mathbf{H}_{\ell,k'}^{(i)} \mathbf{f}_\ell \mathbf{f}_\ell^H \mathbf{H}_{\ell,k'}^{(i)} + \sigma^2 \mathbf{I}$. Note that $\mathbf{z}_{k',i}^\star$ is the MMSE receiver solution.

Using the formulation in Lemma 1, the optimization problem (35)-(36) can be written as

$$\underset{\{\mathbf{f}_k\}, \{\mathbf{z}_{k',i}\}, \{w_k\}}{\text{minimize}} \sum_{k \in K_{\text{Tx}}} I_k \frac{g_k(\mathbf{f}_k)}{u_k(\{\mathbf{f}_k\}, \mathbf{z}_{k',i}, w_k)} \quad (42)$$

$$\text{subject to} \quad \|\mathbf{f}_k\|_2^2 \leq P_k \quad \forall k \in \mathcal{K}_{\text{Tx}} \quad (43)$$

where

$$g_k(\mathbf{f}_k) = 1 - \beta_k + \beta_k \|\mathbf{f}_k\|_2^2 + \beta_k P_c. \quad (44)$$

For a given $\{\mathbf{f}_k\}$, the optimal solutions of $\mathbf{z}_{k',i}$ and w_k for (42)-(43) are given by (40) and (41). Moreover, for given $\mathbf{z}_{k',i}$ and w_k, the function g_k is convex and u_k is concave with respect to $\{\mathbf{f}_k\}$. Optimization (42)-(43) with respect to $\{\mathbf{f}_k\}$ is thus a convex-concave multiple-ratio fractional programming problem [54], which is not convex. Motivated by [55], we will exploit the fractional programming approach to solve it.

Specifically, we have the following theorem, which introduces an equivalent problem that is convex with respect to each individual set of variables $\{\mathbf{f}_k\}$, $\{\mathbf{z}_{k',i}\}$, and $\{w_k\}$ when two other sets of variables $\{\lambda_k\}$ and $\{\gamma_k\}$ are introduced.

Theorem 1. *Consider the optimization problem*

$$\underset{\{\mathbf{f}_k\}, \{\mathbf{z}_{k',i}\}, \{w_k\}}{\text{minimize}} \sum_{k \in \mathcal{K}_{\text{Tx}}} \lambda_k (g_k(\mathbf{f}_k) - \gamma_k u_k(\{\mathbf{f}_k\}, \mathbf{z}_{k',i}, w_k)) \quad (45)$$

$$\text{subject to} \quad \|\mathbf{f}_k\|_2^2 \leq P_k \quad \forall k \in \mathcal{K}_{\text{Tx}} \quad (46)$$

and the system equations

$$\lambda_k = \frac{I_k}{u_k(\{\mathbf{f}_k\}, \mathbf{z}_{k',i}, w_k)}, \quad \gamma_k = \frac{g_k(\mathbf{f}_k)}{u_k(\{\mathbf{f}_k\}, \mathbf{z}_{k',i}, w_k)}. \quad (47)$$

If $\{\tilde{\mathbf{f}}_k\}$, $\{\tilde{\mathbf{z}}_{k',i}\}$, and $\{\tilde{w}_k\}$ are solutions of the problem (45)-(46) and also simultaneously satisfy the system equations in (47), then they are optimal solutions to (42)-(43).

The proof of Theorem 1 is relegated to Appendix A. Optimization (42)-(43) is equivalent to (45)-(47) in the sense that they have the same globally optimal solutions. Using the fact that optimization (45)-(46) is convex with respect to each set of variables $\{\mathbf{f}_k\}$, $\{\mathbf{z}_{k',i}\}$, and $\{w_k\}$, we will solve for each set, iteratively, which will yield solutions with $\{\lambda_k\}$ and $\{\gamma_k\}$ being fixed. Specifically, we propose an iterative algorithm to solve (45)-(46) and satisfy the system equations (47) simultaneously: given $\{\lambda_k\}$ and $\{\gamma_k\}$, we solve for $\{\mathbf{f}_k\}$, $\{\mathbf{z}_{k',i}\}$, and $\{w_k\}$, and then update $\{\lambda_k\}$ and $\{\gamma_k\}$ from the updated variables $\{\mathbf{f}_k\}$, $\{\mathbf{z}_{k',i}\}$, and $\{w_k\}$.

To solve (45)-(46) for fixed $\{\lambda_k\}$ and $\{\gamma_k\}$, we use the block coordinate descent (BCD) method, where each set of the variables is solved fixing the other two. In particular, with $\{\mathbf{f}_k\}$ and $\{w_k\}$ fixed, the optimal solution of each $\mathbf{z}_{k',i}$ is given in (40). With $\{\mathbf{f}_k\}$ and $\{\mathbf{z}_{k',i}\}$ fixed, the optimal solution of each

Algorithm 2 Minimum communication overhead beamforming (MCOB) algorithm

1: **Initialization.** Choose arbitrary $\{\mathbf{f}_k^{(0)}\}$ with $||\mathbf{f}_k^{(0)}||_2^2 = P_k$ where $\mathbf{f}_k^{(0)} \in \mathbb{C}^{N_k}$. Update $\{\mathbf{z}_{k',i}^{(0)}\}$ and $\{w_k^{(0)}\}$ from (40) and (41). Update the system equations $\{\lambda_k^{(0)}\}$ and $\{\gamma_k^{(0)}\}$ from (47) with $\{\mathbf{f}_k^{(0)}\}$, $\{\mathbf{z}_{k',i}^{(0)}\}$, and $\{w_k^{(0)}\}$. Set $\rho^{(0)} = 1$. Set the iteration number $j = 1$.
2: **repeat**
3:     Solve for $\{\mathbf{f}_k^{(j)}\}$ from (50)-(51).
4:     Update $\{\mathbf{z}_{k',i}^{(j)}\}$ and $\{w_k^{(j)}\}$ from (40) and (41).
5:     Calculate the objective function $\rho^{(j)}$ in (45) with $\{\mathbf{f}_k^{(j)}\}$, $\{\mathbf{z}_{k',i}^{(j)}\}$, and $\{w_k^{(j)}\}$.
6:     Update the system equations $\{\lambda_k^{(j)}\}$ and $\{\gamma_k^{(j)}\}$ from (47) with $\{\mathbf{f}_k^{(j)}\}$, $\{\mathbf{z}_{k',i}^{(j)}\}$, and $\{w_k^{(j)}\}$.
7:     Calculate the system equation error $\zeta^{(j)}$ from (52).
8:     Set $j = j + 1$.
9: **until** $\left|\rho^{(j)} - \rho^{(j-1)}\right| \leq \varepsilon$ and $\zeta^{(j)} \leq \varepsilon$ (e.g., $\varepsilon = 10^{-4}$)
10: Obtain the solutions, $\{\mathbf{f}_k\} = \{\mathbf{f}_k^{(j)}\}$ and $\{\mathbf{z}_{k',i}\} = \{\mathbf{z}_{k',i}^{(j)}\}$
11: **return** $\{\mathbf{f}_k\}, \{\mathbf{z}_{k',i}\}$

---

w_k is given in (41). The remaining part is to solve for $\{\mathbf{f}_k\}$ with $\{\mathbf{z}_{k',i}\}$ and $\{w_k\}$ fixed, which is not-trivial. To solve for $\{\mathbf{f}_k\}$, we reorganize the objective function in (45) by replacing u_k and g_k with (38) and (44):

$$\sum_{k \in \mathcal{K}_{\text{Tx}}} \lambda_k g_k(\mathbf{f}_k) - \sum_{k \in K_{\text{Tx}}} \lambda_k \gamma_k u_k(\{\mathbf{f}_k\}, \mathbf{z}_{k',i}, w_k)$$

$$= \sum_{k \in \mathcal{K}_{\text{Tx}}} \lambda_k \left(1 - \beta_k + \beta_k P_{\text{c}} - \gamma_k w_k^{-1} - \gamma_k w_k^{-1} \mathbf{z}_{k',i}^H \mathbf{z}_{k',i} \right.$$

$$\left. - \gamma_k \log w_k + \gamma_k \right) + \sum_{k \in \mathcal{K}_{\text{Tx}}} \mathbf{f}_k^H \mathbf{\Sigma}_k \mathbf{f}_k$$

$$+ \sum_{k \in \mathcal{K}_{\text{Tx}}} \lambda_k \left(\beta_k ||\mathbf{f}_k||_2^2 - 2\gamma_k w_k^{-1} \text{Re}[\mathbf{z}_{k',i}^H \mathbf{H}_{k,k'}^{(i)} \mathbf{f}_k] \right) \quad (48)$$

where

$$\mathbf{\Sigma}_k = \sum_{\ell \in \mathcal{K}_{\text{Tx}}} \lambda_\ell \gamma_\ell w_\ell^{-1} b_{\ell,i(\ell)} \mathbf{H}_{k,k'(\ell)}^{(i(\ell))H} \mathbf{z}_{k'(\ell),i(\ell)} \mathbf{z}_{k'(\ell),i(\ell)}^H \mathbf{H}_{k,k'(\ell)}^{(i(\ell))}.$$

$$(49)$$

In (49), for the tuple $(\ell, k'(\ell), i(\ell))$, $k'(\ell)$ denotes the receive node of the transmit node ℓ and $i(\ell)$ denotes the subchannel that ℓ uses. Since the first term in (48) is constant with respect to $\{\mathbf{f}_k\}$, we are only interested in the second and third terms. The optimization can be decoupled into $|\mathcal{K}_{\text{Tx}}|$ independent subproblems, one for each $\mathbf{f}_k$, as

$$\underset{\mathbf{f}_k}{\text{minimize}} \quad \lambda_k \beta_k ||\mathbf{f}_k||_2^2 - 2\lambda_k \gamma_k w_k^{-1} \text{Re}[\mathbf{z}_{k',i}^H \mathbf{H}_{k,k'}^{(i)} \mathbf{f}_k]$$

$$+ \mathbf{f}_k^H \mathbf{\Sigma}_k \mathbf{f}_k \quad (50)$$

$$\text{subject to} \quad ||\mathbf{f}_k||_2^2 \leq P_k \quad (51)$$

A closed-form solution can be derived for (50)-(51) with the Karush–Kuhn–Tucker (KKT) conditions [56], since the reduced problem (50)-(51) is a quadratically constrained quadratic program (QCQP). The detailed derivation of this procedure is given in Appendix C. Through this process, we have transformed the original communication overhead minimization problem (33)-(34) to a form in (40)-(41), (50)-(51) which allows each beamformer to be designed individually, under the BCD framework. The solution $\mathbf{f}_k$ minimizes the communication

Algorithm 3 Efficient alternate optimization

1: **Initialization.** Set $Y_{\text{total}}^{\text{cur}} = \Upsilon$ (e.g., $\Upsilon = 10^5$). Generate arbitrary $\{a_{k,k'}\}$ and $\{b_{k,i}\}$, which satisfy (22)-(24).
2: **repeat**
3:     Update $Y_{\text{total}}^{\text{prev}} = Y_{\text{total}}^{\text{cur}}$.
4:     *Beamformer design*: Solve for $\{\mathbf{f}_k\}$ and $\{\mathbf{z}_{k',i}\}$ with $\{a_{k,k'}\}$ and $\{b_{k,i}\}$, using Algorithm 2.
5:     *Greedy algorithm*: Solve for $\{a_{k,k'}\}$, $\{b_{k,i}\}$, and $\{F_{k,k'}\}$ with $\{\mathbf{f}_k\}$ and $\{\mathbf{z}_{k',i}\}$, using Algorithm 4.
6:     Calculate $Y_{\text{total}}^{\text{cur}}$ in (20) with $\{a_{k,k'}\}$, $\{b_{k,i}\}$, $\{\mathbf{f}_k\}$, $\{\mathbf{z}_{k',i}\}$, and $\{F_{k,k'}\}$.
7: **until** $|Y_{\text{total}}^{\text{cur}} - Y_{\text{total}}^{\text{prev}}| < \varepsilon$ (e.g., $\varepsilon = 10^{-4}$)
8: **return** $\{a_{k,k'}\}, \{b_{k,i}\}, \{\mathbf{f}_k\}, \{\mathbf{z}_{k',i}\}, \{F_{k,k'}\}$

---

overhead caused by concurrent task offloading, for a given topology configuration, subchannel allocation, and MMSE combiner at the receive nodes.

With $\{\mathbf{f}_k\}$, $\{\mathbf{z}_{k',i}\}$, and $\{w_k\}$ in hand, we can then update $\{\lambda_k\}$ and $\{\gamma_k\}$ using (47). The overall MCOB algorithm is demonstrated in Algorithm 2, which determines $\{\mathbf{f}_k\}$, $\{\mathbf{z}_{k',i}\}$, $\{w_k\}$, $\{\lambda_k\}$, and $\{\gamma_k\}$ that are the solutions to (45)-(47). The algorithm runs until the objective function value ρ in (45) changes less than a threshold and the system equation error is also less than that. We define the system equation error as

$$\zeta^{(j)} = \sum_{k \in \mathcal{K}_{\text{Tx}}} \left(\left|\lambda_k^{(j)} - \lambda_k^{(j-1)}\right|^2 + \left|\gamma_k^{(j)} - \gamma_k^{(j-1)}\right|^2 \right). \quad (52)$$

B. Efficient Alternate Optimization

In this section, we propose a computationally efficient alternative to the semi-exhaustive search optimization (Algorithm 1) that avoids the brute force strategy of handling the binary variables $\{a_{k,k'}\}$ and $\{b_{k,i}\}$. This method, which we term *efficient alternate optimization*, is demonstrated in Algorithm 3. The key idea is that we divide the optimization (21)-(27) into two subproblems and solve them alternately. The first problem is the beamformer design for the variables $\{\mathbf{f}_k\}$ and $\{\mathbf{z}_{k',i}\}$ given task assignments $\{a_{k,k'}\}$ and subchannel allocations $\{b_{k,i}\}$, which we already developed in Algorithm 2. The second problem is the resource allocation design for $\{a_{k,k'}\}$, $\{b_{k,i}\}$, and CPU allocation $\{F_{k,k'}\}$ with given beamformer design variables $\{\mathbf{f}_k\}$ and $\{\mathbf{z}_{k',i}\}$. While each sub-algorithm is dedicated to each sub-problem, the overall composition via iterative alternating optimization is aiming to solve the overall problem (21)-(27). The experiments in Section V demonstrate that this efficient alternate optimization achieves a substantial reduction in network overhead compared to local processing.

Algorithm 4 demonstrates our approach for the resource allocation problem. The key idea is that at each step, we determine the data stream tuple (k, k', i) that provides the most reduction in overhead, and allocate these resources accordingly. The process continues until there are no cases that any tuple will improve the optimization objective. The maximizer for the current step is determined as

$$(\tilde{k}, \tilde{k}', \tilde{i}) = \underset{\substack{k \in K_{\text{Tx}}, k' \in K_{\text{Rx}}, k \neq k', i \in I, \\ k,k',i \text{ satisfy } (22)-(24)}}{\arg\max} \eta_{k,k',i}, \quad (53)$$

where $\mathcal{K}_{\text{Tx}}$ denotes the candidate set of transmit nodes, $\mathcal{K}_{\text{Rx}}$ denotes the candidate set of receive nodes, and $\eta_{k,k',i}$ is the *offloading benefit* provided by tuple (k, k', i). The offloading benefit is defined as

Algorithm 4 Greedy algorithm for task assignment, subchannel allocation, and CPU allocation

1: **Initialization.** Set $\mathcal{K}_{\mathrm{Tx}} = \mathcal{K}$, $\mathcal{K}_{\mathrm{Rx}} = \mathcal{K}$, $\mathcal{A} = \emptyset$, and $\mathcal{B} = \emptyset$.
2: **repeat**
3:    $(\tilde{k}, \tilde{k}', \tilde{i}) = \underset{\substack{k \in K_{\mathrm{Tx}},\, k' \in K_{\mathrm{Rx}},\, k \neq k',\, i \in I, \\ k, k', i \text{ satisfy } (22)-(24)}}{\arg\max} \eta_{k,k',i}$, where
   $\eta_{k,k',i} = Y^{\mathrm{loc}} - Y^{\mathrm{off}}$. The Y^{loc} and Y^{off} are given in (55)&(57).
4:    **if** $\eta_{\tilde{k},\tilde{k}',\tilde{i}} \leq 0$ **then**
5:       Update $\mathcal{A} \leftarrow \mathcal{A} \cup \{(k,k) : k \in \mathcal{K}_{\mathrm{Tx}}\}$ and terminate the algorithm (set $\mathcal{K}_{\mathrm{Tx}} = \emptyset$).
6:    **else**
7:       Update $\mathcal{K}_{\mathrm{Tx}} \leftarrow \mathcal{K}_{\mathrm{Tx}} \setminus \{\tilde{k}, \tilde{k}'\}$,
      $\mathcal{A} \leftarrow \mathcal{A} \cup \{(\tilde{k}, \tilde{k}'), (\tilde{k}', \tilde{k}')\}$, and $\mathcal{B} \leftarrow \mathcal{B} \cup \{(\tilde{k}, \tilde{i})\}$.
8:    **end if**
9: **until** $\mathcal{K}_{\mathrm{Tx}} = \emptyset$
10: Update $\{a_{k,k'}\}$ with $a_{k,k'} = 1$ for $(k,k') \in \mathcal{A}$ and $a_{k,k'} = 0$ otherwise. Update $\{b_{k,i}\}$ with $b_{k,i} = 1$ for $(k,i) \in \mathcal{B}$ and $b_{k,i} = 0$ otherwise. Update $\{F_{k,k'}\}$ from (28)-(29).
11: **return** $\{a_{k,k'}\}, \{b_{k,i}\}, \{F_{k,k'}\}$

$$\eta_{k,k',i} = Y^{\mathrm{loc}} - Y^{\mathrm{off}}, \tag{54}$$

which quantifies the reduction in network overhead by offloading from node k to k' on subchannel i on top of the current resource allocations. Y^{loc} denotes the total network overhead in case of no offloading from k to k', while Y^{off} denotes the total network overhead in case of offloading.

Algorithm 4 begins with $\mathcal{K}_{\mathrm{Tx}} = \mathcal{K}$, $\mathcal{K}_{\mathrm{Rx}} = \mathcal{K}$, meaning that all of the nodes are candidates for transmit and receive. With $\mathcal{A}$ denoting the task assignment set $\mathcal{A} = \{(k,k') : a_{k,k'} = 1\}$ and $\mathcal{B}$ denoting the subchannel allocation set $\mathcal{B} = \{(k,i) : b_{k,i} = 1\}$, initially $\mathcal{A} = \mathcal{B} = \emptyset$.

For a given $\mathcal{A}$ and $\mathcal{B}$, Y^{loc} is computed as

$$Y^{\mathrm{loc}} = \sum_{(k,k') \in \mathcal{A}^{\mathrm{loc}}} Y_{\mathrm{comp}}^{\star}(k,k') + \sum_{\substack{(k,k') \in \mathcal{A}^{\mathrm{loc}}, \\ k \neq k', (k,i) \in \mathcal{B}}} Y_{\mathrm{comm}}(k,k'), \tag{55}$$

where $Y_{\mathrm{comp}}^{\star}(k,k')$ is the value of $Y_{\mathrm{comp}}(k,k')$ obtained by the optimal solution to (28)-(29) for the allocation set $\mathcal{A}^{\mathrm{loc}}$, and

$$\mathcal{A}^{\mathrm{loc}} = \mathcal{A} \cup \{(k,k), (k',k')\}. \tag{56}$$

$\mathcal{A}^{\mathrm{loc}}$ denotes the new task assignment set when node k and k' process locally. In Algorithm 4, k' is added as a local processing node. Otherwise, it might happen that at current step, task k occupies all of the CPU of node k' without consideration of allocating CPU to task k'. Then, k' has no choice but to offload to other nodes at the next step. To overcome this, we consider the local processing of task k' when task k is being considered for offloading to node k'.

On the other hand, Y^{off} is given by

$$Y^{\mathrm{off}} = \sum_{(k,k') \in \mathcal{A}^{\mathrm{off}}} Y_{\mathrm{comp}}^{\star}(k,k') + \sum_{\substack{(k,k') \in \mathcal{A}^{\mathrm{off}}, \\ k \neq k', (k,i) \in \mathcal{B}^{\mathrm{off}}}} Y_{\mathrm{comm}}(k,k'), \tag{57}$$

where $Y_{\mathrm{comp}}^{\star}(k,k')$ is the optimal value for the allocation set $\mathcal{A}^{\mathrm{off}}$, and

$$\mathcal{A}^{\mathrm{off}} = \mathcal{A} \cup \{(k,k'), (k',k')\}, \quad \mathcal{B}^{\mathrm{off}} = \mathcal{B} \cup \{(k,i)\}. \tag{58}$$

$\mathcal{A}^{\mathrm{off}}$ denotes the new task assignment set when node k offloads to k'. $\mathcal{B}^{\mathrm{off}}$ denotes the new subchannel allocation set when node k uses subchannel i for offloading.

In each step of Algorithm 4, as long as the best data stream $(\tilde{k}, \tilde{k}', \tilde{i})$ from (53) has a positive offloading benefit $\eta_{\tilde{k},\tilde{k}',\tilde{i}}$, then

these resources are allocated. This means task $\tilde{k}$ is offloaded to node $\tilde{k}'$ on subchannel $\tilde{i}$, and node $\tilde{k}'$ locally processes its own task $\tilde{k}'$. As a result, we update $\mathcal{A} \leftarrow \mathcal{A} \cup \{(\tilde{k}, \tilde{k}'), (\tilde{k}', \tilde{k}')\}$ and $\mathcal{B} \leftarrow \mathcal{B} \cup \{(\tilde{k}, \tilde{i})\}$. Since nodes $\tilde{k}$ and $\tilde{k}'$ are no longer candidate transmit nodes, we update $\mathcal{K}_{\mathrm{Tx}} \leftarrow \mathcal{K}_{\mathrm{Tx}} \setminus \{\tilde{k}, \tilde{k}'\}$. Once there is no data stream with positive offloading benefit, the algorithm is terminated, and all remaining candidate transmit nodes are assigned to local processing.

C. Discussion of Optimality

As mentioned previously, Algorithm 1 (semi-exhaustive search) represents a best-effort approach for solving the optimization (21)-(27) with manageable complexity for small networks. We will explain this reasoning now. Then, in Section IV-D, we will compare the computational complexities between Algorithms 1 and 3.

The optimal solution to (21)-(27) can be obtained (in theory) by solving for the non-integer variables for all possible combinations of integer variables. If we represent the objective function Y_{total} in its functional form $Y_{\mathrm{total}}(\{a_{k,k'}\}, \{b_{k,i}\}, \{F_{k,k'}\}, \{\mathbf{f}_k\}, \{\mathbf{z}_{k',i}\})$, then by fixing the binary variables as $\{\bar{a}_{k,k'}\}$ and $\{\bar{b}_{k,i}\}$, we are left with

$$\underset{\{F_{k,k'}\}, \{\mathbf{f}_k\}, \{\mathbf{z}_{k',i}\}}{\mathrm{minimize}}\ Y_{\mathrm{total}}(\{\bar{a}_{k,k'}\}, \{\bar{b}_{k,i}\}, \{F_{k,k'}\}, \{\mathbf{f}_k\}, \{\mathbf{z}_{k',i}\}), \tag{59}$$

subject to the constraints. Since the CPU allocation variable $\{F_{k,k'}\}$ is not affected by the beamformer design variables $\{\mathbf{f}_k\}$ and $\{\mathbf{z}_{k',i}\}$, and vice versa, this optimization can be divided into two independent problems given by

$$\underset{\{F_{k,k'}\}}{\mathrm{minimize}}\ Y_{\mathrm{total}}(\{\bar{a}_{k,k'}\}, \{\bar{b}_{k,i}\}, \{F_{k,k'}\}, \{\mathbf{f}_k\}, \{\mathbf{z}_{k',i}\}), \tag{60}$$

$$\underset{\{\mathbf{f}_k\}, \{\mathbf{z}_{k',i}\}}{\mathrm{minimize}}\ Y_{\mathrm{total}}(\{\bar{a}_{k,k'}\}, \{\bar{b}_{k,i}\}, \{F_{k,k'}\}, \{\mathbf{f}_k\}, \{\mathbf{z}_{k',i}\}). \tag{61}$$

In summary, the optimization variables in (59) are separable, and the problem can be decomposed into (60) and (61) for every combination of $\{\bar{a}_{k,k'}\}$ and $\{\bar{b}_{k,i}\}$.

Consider how the proposed semi-exhaustive search optimization addresses (60) and (61). Problem (60) is convex: for this, we arrive at the convex problem (30)-(32) decomposed across nodes. Thus, we obtain the optimal solution $\{F_{k,k'}^{\star}\}$ for this set of integer variables as

$$\{F_{k,k'}^{\star}\} = \underset{\{F_{k,k'}\}}{\arg\min}\ Y_{\mathrm{total}}(\{\bar{a}_{k,k'}\}, \{\bar{b}_{k,i}\}, \{F_{k,k'}\}, \{\mathbf{f}_k\}, \{\mathbf{z}_{k',i}\}). \tag{62}$$

In contrast, problem (61) is non-convex. To solve it, we developed the MCOB algorithm for optimizing the receive combiner $\{\mathbf{z}_{k',i}\}$ fixing the transmit beamformer $\{\mathbf{f}_k\}$ and vice versa (see Algorithm 2). The solution for $\{\mathbf{z}_{k',i}^{\star}\}$ for a fixed $\{\bar{\mathbf{f}}_k\}$ based on an MMSE receiver is given in (40), such that

$$\{\mathbf{z}_{k',i}^{\star}\} = \underset{\{\mathbf{z}_{k',i}\}}{\arg\min}\ Y_{\mathrm{total}}(\{\bar{a}_{k,k'}\}, \{\bar{b}_{k,i}\}, \{F_{k,k'}\}, \{\bar{\mathbf{f}}_k\}, \{\mathbf{z}_{k',i}\}). \tag{63}$$

The solution $\{\mathbf{f}_k^{\star}\}$ for a fixed $\{\bar{\mathbf{z}}_{k',i}\}$ is given in (50)-(51), such that

$$\{\mathbf{f}_k^{\star}\} = \underset{\{\mathbf{f}_k\}, \{P_k\}}{\arg\min}\ Y_{\mathrm{total}}(\{\bar{a}_{k,k'}\}, \{\bar{b}_{k,i}\}, \{F_{k,k'}\}, \{\mathbf{f}_k\}, \{\bar{\mathbf{z}}_{k',i}\}). \tag{64}$$

Although $\{\mathbf{f}_k^\star\}$ and $\{\mathbf{z}_{k',i}^\star\}$ are not guaranteed to be optimal solutions to the non-convex optimization in (61), they are practical solutions that have an efficient tradeoff between optimality and computational complexity. Similar tradeoffs have been made in related works [34], [55], [57], [58] for this reason. However, $\{\mathbf{z}_{k',i}^\star\}$ is an optimal solution for a given $\{\bar{\mathbf{f}}_k\}$, and $\{\mathbf{f}_k^\star\}$ is an optimal solution for a given $\{\bar{\mathbf{z}}_{k',i}\}$, which is one of the main contributions of this paper.

D. Computational Complexity

The semi-exhaustive search optimization still requires significant computation due to the large potential number of combinations of $\{a_{k,k'}\}$ and $\{b_{k,i}\}$. The efficient alternate optimization is much more computationally efficient, and as we will see in Section V-B, its observed solutions have comparable performance to that of the semi-exhaustive search optimization.

Considering the computational complexities of both algorithms with respect to the integer variables, we have the following lemma:

Lemma 2. *With respect to the task assignment and subchannel allocation variables, the semi-exhaustive search optimization (Algorithm 1) has $\mathcal{O}((KS-S+1)^K)$ and the efficient alternate optimization (Algorithm 3) has $\mathcal{O}(K^3 S)$, where K and S are the number of nodes and number of subchannels, respectively.*

The proof is relegated to Appendix B. The computational complexity of the semi-exhaustive search optimization is worse than exponential in the number of nodes, while that of the efficient alternate optimization is polynomial. For example, if we consider $K = 10$ and $S = 2$, the semi-exhaustive search optimization already has up to 19^{10} combinations of binary variables to consider (depending on condition (23)), and the optimization for non-integer variables will be performed for each combination. In contrast, the efficient alternate optimization limits the number of combinations to at most 2000, depending on how many combinations provide a positive offloading benefit. Further, the full optimization over non-integer variables is performed once the best combination is determined, i.e., it is not performed for every binary combination.

V. PERFORMANCE EVALUATION AND DISCUSSION

In this section, we conduct experiments to validate our methods for minimizing the total network overhead in D2D networks. After discussing our setup (Section V-A), in Section V-B, we will quantify improvements relative to local processing and compare the efficient alternate optimization to the semi-exhaustive search optimization. Then, in Sections V-C to V-F, we will evaluate the performance of the efficient alternate optimization under variation of different network parameters.

A. Experimental Setup

1) Parameter values: For all of our experiments, we select values that are common for mobile computing environments [16], [59]. Each channel $\mathbf{H}_{k,k'}^{(i)}$ is modeled as a Rayleigh fading channel where the entries are i.i.d. following $\mathcal{CN}(0, \beta_{k,k'})$ [25]. Here, $\beta_{k,k'} = \beta_0 - 10\alpha \log_{10}(d_{k,k'}/d_0)$ (dB) denotes the large-scale fading factor between nodes k and k', where $\beta_0 = -30$

dB is the path loss at the distance $d_0 = 1$ m, and $\alpha = 3.5$ is the path loss exponent in cellular networks [60]. The distance between nodes k and k', $d_{k,k'}$, is randomly generated as $d_{k,k'} \sim \mathcal{U}(10, 30)$ (m), where $\mathcal{U}(a, b)$ denotes the uniform distribution on the interval $[a, b]$. We assume that the individual transmit power limit is $P_k = P = 3$ dBW [16] for $k \in \mathcal{K}$, the noise power is $\sigma^2 = -90$ dBW [44], the circuit power is $P_c = -20$ dBW [59], and the subchannel bandwidth is $W = 1$ MHz. The beamformers $\mathbf{f}_k$ and combiners $\mathbf{z}_{k',i}$ are initially generated to be uniformly distributed on the complex sphere [61] with radius $\sqrt{P}$ and 1, respectively, for $k, k' \in \mathcal{K}$ and $i \in \mathcal{S}$.

As a computation task, we consider image recognition [8], [52], which is a popular application on mobile devices (e.g., for user authentication via facial recognition). To emulate heterogeneous devices, we consider different task sizes and CPUs across the nodes. For the task size, we sample $I_k \sim \mathcal{U}(1, 8)$ in Mbits, i.e., $\mathcal{U}(0.128, 1)$ in Mbytes, for each node k. These represent common image sizes found on mobile devices [8], [52]. For CPU, we consider a bimodal distribution for each node k': $F_{k'} \sim \frac{3}{4}\mathcal{U}(0.1, 0.2) + \frac{1}{4}\mathcal{U}(0.9, 1)$, with units of GHz. This selection generates a composition of resource-hungry and resource-rich devices, reflecting common processor clock speeds found in mobile devices [8], [16], [52]. We assume constant processing density $\mu_k = 200$ cycles/bit, and energy coefficients $\kappa_{k'} = 3.5 \times 10^{-27}$ across all nodes, as in [16]. The overhead factor β_k is assumed to be the same for all nodes, i.e., $\beta_k = \beta$ for all k. Unless otherwise stated, $\beta = 0.5$. All nodes are considered to have N transmit and receive antennas, i.e., $N_k = N$ for all k. Each experiment is averaged over 20 different samplings of task sizes, CPUs, and channel realizations. For the efficient alternate optimization, we consider 10 different initializations of $\{a_{k,k'}\}$ and $\{b_{k,i}\}$, and choose the best solution. The threshold for Algorithms 2&3 is $\varepsilon = 10^{-4}$.

2) Baselines: We compare the proposed algorithms with three different baselines. The first baseline is local computation, where all the nodes locally process their own tasks without offloading. The total network overhead for local processing is

$$Y_{\text{total}} = \sum_{k=1}^{K} Y_{\text{comp}}(k, k). \quad (65)$$

This baseline will be used to assess the improvements obtained via our offloading optimization methodology.

The second baseline is the efficient alternate optimization with the weighted minimum mean square error (WMMSE) approach [34] used in place of Algorithm 2. WMMSE is an existing method for beamformer design with a sum-utility maximization objective, proposed in [34]. Specifically, in place of (33)-(34), with WMMSE, we minimize the total communication time as

$$\underset{\{\mathbf{f}_k\}, \{\mathbf{z}_{k',i}\}}{\text{minimize}} \quad \sum_{k=1}^{K} \sum_{k' \neq k}^{K} a_{k,k'} T_{\text{comm}}(k, k'). \quad (66)$$

This baseline will allow us to assess the importance of balancing time and energy as competing objectives in overhead minimization.

The third baseline is the efficient alternate optimization but with equal CPU allocation. For a given task assignment, the CPU is equally allocated across the requested tasks. Specifically,
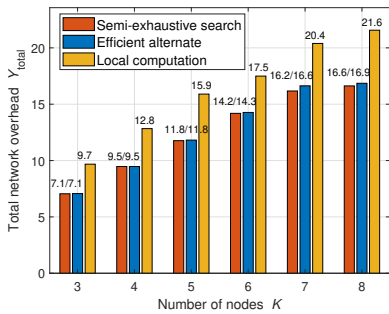
Fig. 3: The total network overhead obtained by the semi-exhaustive search optimization, the efficient alternate optimization and local computation where $S = 2$ and $N = 5$.
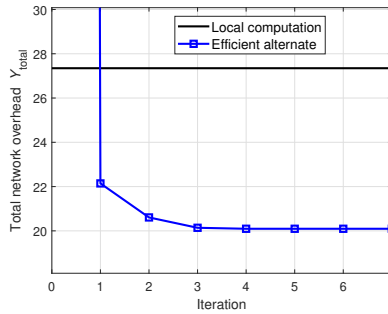

Fig. 4: Convergence behavior of the efficient alternate optimization algorithm when $K = 10$, $S = 2$, and $N = 5$. The total network overhead converges within a few iterations.
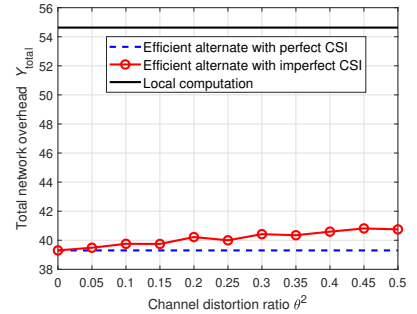

Fig. 5: Effect of imperfect CSI on performance of the efficient alternate optimization with $K = 20$, $S = 2$, and $N = 5$. Significant improvements are still obtained as θ^2 increases.

in Algorithm 4, we do not consider the minimization problem with respect to $\{F_{k,k'}\}$ in (55) and (57). This baseline, together with the second baseline, will assess the importance of our formulation as a joint optimization over communication and computation resources.

B. Optimality, Convergence, and Imperfect CSI

Our first experiment compares the total network overhead incurred by semi-exhaustive search, efficient alternate optimization, and local computation for different numbers of nodes K. Fig. 3 shows the results as K varies from 3 to 8 in a small network with $S = 2$ and $N = 5$. Compared to the local computation, the offloading through our methodology results in a significant decrease between 19% and 27% in the total network overhead even for small D2D networks. The semi-exhaustive search optimization provides a lower bound on the minimum achievable overhead required by the efficient alternate optimization, as discussed in Section IV-C. However, we find that the implementation of the semi-exhaustive search is computationally infeasible for more than $K = 8$ nodes, consistent with its computational complexity given in Lemma 2. Furthermore, the efficient alternate optimization gives almost the same overhead performance as the semi-exhaustive search optimization. Henceforth, we will present results based on the efficient alternate optimization. An experiment on the runtime growth rate of the efficient alternate optimization is provided in Appendix D-A, verifying its polynomial complexity.

Fig. 4 shows the convergence behavior of the efficient alternate optimization, plotting the total network overhead obtained after each iteration of Algorithm 3, for the same settings in Fig. 3 and $K = 10$ nodes. After the first iteration, the total network overhead decreases dramatically due to the high reduction in communication overhead obtained from the beamformer design. We observe in our experiments that, the objective function generally converges within a few iterations.

Fig. 5 demonstrates the effect of imperfect channels on our proposed framework and algorithms. Imperfect CSI is modeled by adding a channel distortion value to the actual CSI. Each distortion value follows a Gaussian distribution with zero mean and variance of $\theta^2 \|\mathbf{H}_{k,k'}^{(i)}\|_2^2$, i.e., $\mathcal{CN}(0, \theta^2 \|\mathbf{H}_{k,k'}^{(i)}\|_2^2)$, where θ^2 denotes the channel distortion ratio. As the distortion ratio increases to $\theta^2 = 0.5$, the overhead improvement relative to local computing only decreases from 28% to 25%. This shows that our methodology is still applicable for minimizing total D2D network overhead in the presence of imperfect CSI.

C. Communication-Computation Overhead Tradeoff

Our next experiment assesses the benefit provided by each offloading node that the greedy algorithm adds in the efficient alternate optimization. Specifically, Fig. 6 shows the change in overhead as more data streams (k, k', i) are added for offloading in Algorithm 4, for $K = 30$, $S = 2$, and $N = 5$. We show the evolution of the communication overhead $Y_{\text{comm}} = \sum_{k=1}^{K} \sum_{k' \neq k}^{K} a_{k,k'} Y_{\text{comm}}(k, k')$, the computation overhead $Y_{\text{comp}} = \sum_{k=1}^{K} \sum_{k'=1}^{K} a_{k,k'} Y_{\text{comp}}(k, k')$, and the total overhead $Y_{\text{total}} = Y_{\text{comm}} + Y_{\text{comp}}$. Overall, we see that the total network overhead is decreasing at each iteration, which is consistent with the operation of the greedy algorithm. This is obtained by trading an increase in communication overhead for a more substantial decrease in computation overhead. The algorithm successively exploits low-cost opportunities for offloading from resource constrained to resource-rich nodes, until such opportunities are no longer cost-effective. In this case, 23% of the nodes (7 out of 30) become offloading nodes by the time the algorithm terminates.

D. Varying Interference Management Resources

Our next experiments assess the communication overhead reduction obtained by our methodology from leveraging interference management resources. When the number of subchannels S and number of antennas N are limited, we expect that communication overhead will be higher due to decreasing transmission data rates from inter-channel interferences. Fig. 7 shows the effect of S on Y_{comm} for both the efficient alternate optimization and the baseline using WMMSE, when $N = 5$ and there are $K = 30$ devices. We see that the total communication overhead decreases as the number of subchannels increases because more subchannels enable avoiding interferences by allocating non-overlapping subchannels to different data streams. Moreover, the efficient alternate optimization with MCOB gives better performance than that with WMMSE – with improvements of roughly 37% for each choice of S – because MCOB is designed to minimize the total communication overhead, while WMMSE minimizes only the total communication time.

Fig. 8 shows the effect on communication overhead as more antennas are employed for $K = 30$ and $S = 1$. With a limited number of subchannels available ($S = 1$), the beamforming strategy plays a significant role in communication overhead reduction. As N increases, our methodology suppresses the
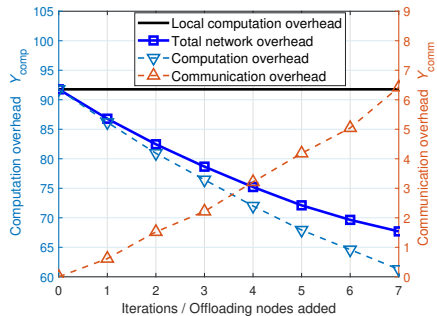
Fig. 6: Evolution of the computation (left axis), communication (right), and total (left axis) network overheads after each iteration of the greedy algorithm, for $K = 30$, $S = 2$, and $N = 5$.
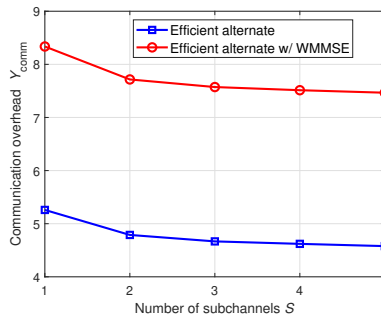

Fig. 7: Communication overhead with varying subchannels S for $K = 30$ and $N = 5$. Our method leverages additional subchannels for overhead reduction via interference mitigation.
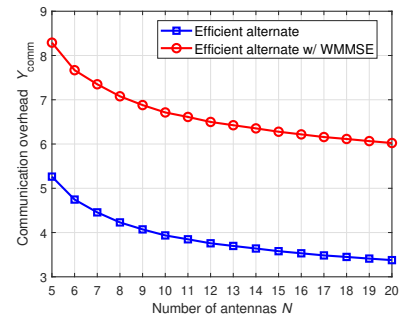

Fig. 8: Communication overhead with varying antennas N for $K = 30$ and $S = 1$. Interferences can be suppressed further with a larger number of antennas.

interference further due to the increased spatial degrees of freedom. The gap in communication overhead between the efficient alternate optimization with MCOB vs. WMMSE increases with more antennas, reaching roughly a 43% improvement.

E. Varying Time/Energy Optimization Importance

We are also interested in the impact of the importance placed on time vs. energy in the total network overhead optimization. Fig. 9 demonstrates the effect of the overhead demand factor β on the total time delay and energy consumption incurred, aggregating all tasks and over both communication and computation overhead components, for $K = 10$, $S = 2$, and $N = 5$. The individual time delay and energy consumption incurred by each task for different β is provided in Appendix D-B. The results in Fig. 9 are consistent with design of the optimization objective: compared to the case of all local computation, total time is reduced the most (36%) when $\beta = 0.2$, while more energy reduction (61%) is achieved when $\beta = 0.8$. This experiment confirms that the objective can be adjusted according to the tradeoff importance requirement of a specific use case.

Now, we consider the communication overhead optimization to evaluate the performance of MCOB, according to different values of β. Fig. 10 shows total communication overhead as β varies from 0 to 1 with MCOB and WMMSE, for $K = 10$, $S = 2$, and $N = 5$. When β is small, the performances of MCOB and WMMSE are almost identical, as the emphasis is on completion time minimization; in other words, WMMSE is a special case of MCOB for $\beta = 0$. Since WMMSE does not factor in energy consumption minimization, at any value of $\beta > 0.15$, MCOB shows a substantial improvement compared to WMMSE. This emphasizes the importance of considering both time delay and energy consumption as components of the overhead model. Note that the communication overhead of WMMSE increases linearly as β increases. This is because the values of T_{comm} and E_{comm} in (19) are constant as a result of WMMSE, and E_{comm} is larger than T_{comm} in this specific D2D scenario. Additionally, the achievable communication overhead by MCOB drops as β increases beyond 0.15. This is also due to the specific D2D scenario being considered: in (19), as a result of MCOB, the value of E_{comm} (in Joules) is further reduced than T_{comm} (in seconds) as β increases.

F. Varying the Number of Nodes

In this experiment, we compare the total network overhead obtained by efficient alternate optimization and the three

baselines as the size of the D2D network changes. Fig. 11 plots Y_{total} as K increases for $S = 2$ and $N = 5$. Compared to local computation, the other three schemes each yield significant reduction in the total network overhead due to the benefit of offloading. Furthermore, the offloading-enabled methods scale better as the size of the network increase: the performance gap widens and the improvement of the efficient alternate optimization stays around 20% to 30% consistently. With more nodes, there are offloading opportunities, leading to more overhead reduction.

In comparing the offloading-enabled methods, we note that the efficient alternate optimization consistently outperforms the equal CPU allocation and WMMSE baselines (by 7% and 6%, respectively), which are partially optimized solutions. This emphasizes the importance of considering a joint optimization of communication and computation resources to obtain the lowest overhead in an environment of heterogeneous wireless devices. The equal CPU allocation baseline is a lower complexity algorithm, however, given it does not solve the CPU optimization problem. This could be a necessary tradeoff if optimization speed is critical, which depends on the timescale at which the solver is employed in practice. We also provide a comparison between the theoretical and observed computational complexity of our method in Appendix D-A.

VI. CONCLUSION AND FUTURE WORK

In this paper, we proposed a novel optimization methodology that minimizes the total network overhead required to process a set of tasks in wireless D2D edge networks. Our optimization model consists of several computation and communication resources including topology configuration, CPU allocations, subchannel allocations, and beamforming design for MIMO transmitters and receivers. Given that the problem is a non-convex MIP, we proposed two methods to solve it: semi-exhaustive search optimization and efficient alternate optimization. In analyzing the optimality and computational complexity of the proposed methods, we showed that the semi-exhaustive search can be regarded as a best effort for optimality, while the efficient alternate optimization has much smaller computational complexity. Through our numerical experiments, we showed the total network overhead can be reduced significantly by leveraging offloading opportunities to resource-rich nodes in D2D networks. Further, in comparison with solutions that only optimize a subset of the variables, our results showed that
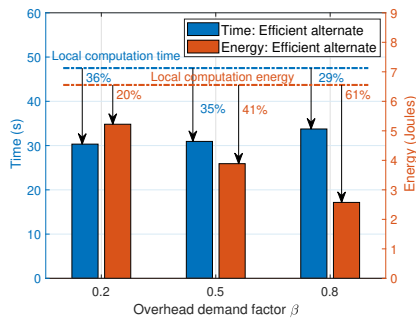
Fig. 9: Total time delay and energy consumption with different β for $K = 10$, $S = 2$, and $N = 5$. The tradeoff in the optimization objective is adjusted according to β.
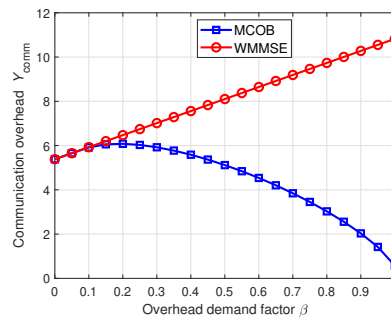

Fig. 10: Communication overhead varying the overhead factor β for $K = 30$, $S = 2$, and $N = 5$. The optimization with MCOB outperforms WMMSE as β increases.
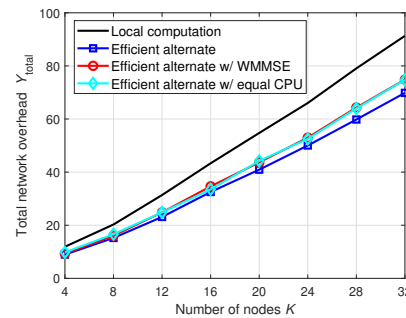

Fig. 11: Impact of the number of nodes K on the total network overhead for $S = 2$ and $N = 5$. The efficient alternate emphasizes the benefit of our holistic optimization approach.

joint communication and computation resource optimization is critical to obtaining the highest reductions in network overhead.

There are several potential avenues of future work. First, while we have focused on tasks generated in a single time period, dynamic task generation at nodes can also be considered. This can be accomplished by augmenting our methodology with a stochastic optimization that models task generation over a long-term time horizon. Second, while we have focused on centralized control for optimizing distributed data processing, it will be desirable to decentralize this optimization in certain settings. One possibility is to investigate a fully distributed implementation of Algorithm 3, where each node solves its local beamforming and resource design sub-problems based on information provided by adjacent neighbors. Lastly, our overhead model can be extended to include components outside of data transmission and processing, e.g., waiting latency, handover latency, and latency of receiving the processed data result back at the origin.

REFERENCES

[1] J. Kim, T. Kim, M. Hashemi, C. G. Brinton, and D. J. Love, "Joint optimization of signal design and resource allocation in wireless D2D edge computing," in *IEEE Conf. Comput. Commun.*, 2020, pp. 2086–2095.

[2] M. Chiang and T. Zhang, "Fog and IoT: An overview of research opportunities," *IEEE Internet Things J.*, vol. 3, no. 6, pp. 854–864, 2016.

[3] C. Yao, X. Wang, Z. Zheng, G. Sun, and L. Song, "Edgeflow: Open-source multi-layer data flow processing in edge computing for 5G and beyond," *IEEE Network*, vol. 33, no. 2, pp. 166–173, 2019.

[4] X. Yang, Z. Chen, K. Li, Y. Sun, N. Liu, W. Xie, and Y. Zhao, "Communication-constrained mobile edge computing systems for wireless virtual reality: Scheduling and tradeoff," *IEEE Access*, vol. 6, pp. 665–677, 2018.

[5] R. Schmoll, S. Pandi, P. J. Braun, and F. H. P. Fitzek, "Demonstration of VR / AR offloading to mobile edge cloud for low latency 5G gaming application," in *IEEE Consumer Commun. Netw. Conf.*, 2018.

[6] K. Sultan, H. Ali, and Z. Zhang, "Big data perspective and challenges in next generation networks," *Future Internet*, vol. 10, no. 7, 2018.

[7] Cisco, "Cisco visual networking index: Forcast and trends, 2017-2022," 2019.

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

[9] H. Q. Le, H. Al-Shatri, and A. Klein, "Efficient resource allocation in mobile-edge computation offloading: Completion time minimization," in *IEEE Int. Symp. Inf. Theory*, 2017, pp. 2513–2517.

[10] N. Naderializadeh and M. Hashemi, "Energy-aware multi-server mobile edge computing: A deep reinforcement learning approach," in *Asilomar Conference on Signals, Systems, and Computers*, 2019.

[11] 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, 2018.

[12] 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 Things J.*, vol. 5, no. 4, pp. 2633–2645, 2017.

[13] T. T. Nguyen, L. Le, and Q. Le-Trung, "Computation offloading in MIMO based mobile edge computing systems under perfect and imperfect CSI estimation," *IEEE Trans. Serv. Comput.*, 2019.

[14] S. Sardellitti, G. Scutari, and S. Barbarossa, "Joint optimization of radio and computational resources for multicell mobile-edge computing," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 1, no. 2, pp. 89–103, 2015.

[15] S. Barbarossa, E. Ceci, M. Merluzzi, and E. Calvanese-Strinati, "Enabling effective mobile edge computing using millimeterwave links," in *IEEE Conf. Commun. Workshops*, 2017, pp. 367–372.

[16] J. Liu, P. Li, J. Liu, and J. Lai, "Joint offloading and transmission power control for mobile edge computing," *IEEE Access*, vol. 7, pp. 81 640–81 651, 2019.

[17] 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 Trans. Veh. Technol.*, vol. 66, no. 8, pp. 7432–7445, 2017.

[18] X. Cao, F. Wang, J. Xu, R. Zhang, and S. Cui, "Joint computation and communication cooperation for energy-efficient mobile edge computing," *IEEE Internet Things J.*, 2018.

[19] X. Diao, J. Zheng, Y. Wu, and Y. Cai, "Joint computing resource, power, and channel allocations for D2D-assisted and NOMA-based mobile edge computing," *IEEE Access*, vol. 7, pp. 9243–9257, 2019.

[20] X. Chen, L. Pu, L. Gao, W. Wu, and D. Wu, "Exploiting massive D2D collaboration for energy-efficient mobile edge computing," *IEEE Wirel. Commun.*, vol. 24, no. 4, pp. 64–71, 2017.

[21] H. Xing, L. Liu, J. Xu, and A. Nallanathan, "Joint task assignment and resource allocation for D2D-enabled mobile-edge computing," *IEEE Trans. Commun.*, 2019.

[22] Y. He, J. Ren, G. Yu, and Y. Cai, "D2D communications meet mobile edge computing for enhanced computation capacity in cellular networks," *IEEE Trans. Wirel. Commun.*, vol. 18, no. 3, pp. 1750–1763, 2019.

[23] Y. Kai, J. Wang, and H. Zhu, "Energy minimization for D2D-assisted mobile edge computing networks," in *IEEE Conf. Commun.*, 2019.

[24] N. Hassan, K.-L. A. Yau, and C. Wu, "Edge computing in 5G: A review," *IEEE Access*, vol. 7, pp. 127 276–127 289, 2019.

[25] D. Tse and P. Viswanath, *Fundamentals of Wireless Communications*. New York: Cambridge University Press, 2005.

[26] F. Wang, L. Song, Z. Han, Q. Zhao, and X. Wang, "Joint scheduling and resource allocation for device-to-device underlay communication," in *IEEE Wireless Commun. Netw. Conf.*, 2013, pp. 134–139.

[27] T. Kim and M. Dong, "An iterative Hungarian method to joint relay selection and resource allocation for D2D communications," *IEEE Wireless Commun. Lett.*, vol. 3, no. 6, pp. 625–628, 2014.

[28] L. Wei, R. Q. Hu, T. He, and Y. Qian, "Device-to-device (D2D) communications underlaying MU-MIMO cellular networks," in *Proc. IEEE Glob. Commun. Conf.*, 2013, pp. 4902–4907.

[29] M.-H. Han, B.-G. Kim, and J.-W. Lee, "Subchannel and transmission mode scheduling for D2D communication in OFDMA networks," in *IEEE Veh. Technol. Conf. (VTC Fall)*, 2012.

[30] J. Zhao, Y. Liu, K. K. Chai, Y. Chen, and M. Elkashlan, "Joint subchannel and power allocation for NOMA enhanced D2D communications," *IEEE Trans. Commun.*, vol. 65, no. 11, pp. 5081–5094, 2017.

[31] D. Feng, L. Lu, Y. Yuan-Wu, G. Y. Li, G. Feng, and S. Li, "Device-to-device communications underlaying cellular networks," *IEEE Trans. Commun.*, vol. 61, no. 8, pp. 3541–3551, 2013.

[32] X. Lin, R. W. Heath, and J. G. Andrews, "The interplay between massive MIMO and underlaid D2D networking," *IEEE Trans. Wirel. Commun.*, vol. 14, no. 6, pp. 3337–3351, 2015.

[33] H. Tang, C. Zhu, and Z. Ding, "Cooperative MIMO precoding for D2D underlay in cellular networks," in *IEEE Conf. Commun.*, 2013, pp. 5517–5521.

[34] Q. Shi, M. Razaviyayn, Z.-Q. Luo, and C. He, "An iteratively weighted MMSE approach to distributed sum-utility maximization for a MIMO interfering broadcast channel," *IEEE Trans. Signal Process.*, vol. 59, no. 5, Sep 2011.

[35] Z. Zhou, H. Liao, X. Zhao, B. Ai, and M. Guizani, "Reliable task offloading for vehicular fog computing under information asymmetry and information uncertainty," *IEEE Trans. Veh. Technol.*, vol. 68, no. 9, pp. 8322–8335, 2019.

[36] Z. Zhou, J. Feng, Z. Chang, and X. Shen, "Energy-efficient edge computing service provisioning for vehicular networks: A consensus ADMM approach," *IEEE Trans. Veh. Technol.*, vol. 68, no. 5, pp. 5087–5099, 2019.

[37] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Commun. Surv. Tutor.*, vol. 19, no. 4, pp. 2322–2358, 2017.

[38] S. G. Johnson and M. Frigo, "A modified split-radix FFT with fewer arithmetic operations," *IEEE Trans. Signal Process.*, vol. 55, no. 1, pp. 111–119, 2006.

[39] Y. Wen, W. Zhang, and H. Luo, "Energy-optimal mobile application execution: Taming resource-poor mobile devices with cloud clones," in *IEEE Conf. Comput. Commun.*, 2012, pp. 2716–2720.

[40] S. Burer and A. N. Letchford, "Non-convex mixed-integer nonlinear programming: A survey," *Surv. Oper. Res. Manag. Sci.*, vol. 17, no. 2, pp. 97–106, 2012.

[41] M. N. Tehrani, M. Uysal, and H. Yanikomeroglu, "Device-to-device communication in 5G cellular networks: challenges, solutions, and future directions," *IEEE Commun. Mag.*, vol. 52, no. 5, pp. 86–92, 2014.

[42] L. Pu, X. Chen, J. Xu, and X. Fu, "D2D fogging: An energy-efficient and incentive-aware task offloading framework via network-assisted D2D collaboration," *IEE. J. Sel. Areas Commun.*, vol. 34, no. 12, pp. 3887–3901, 2016.

[43] S. Mumtaz and J. Rodriguez, *Smart device to smart device communication*. Springer, 2014.

[44] 3GPP TS 36.211, "LTE: Evolved universal terrestrial radio access (e-utra): Physical channels and modulation," vol. V14.2.0 Release 14, 2017.

[45] S.-Y. Lien, S.-L. Shieh, Y. Huang, B. Su, Y.-L. Hsu, and H.-Y. Wei, "5G new radio: Waveform, frame structure, multiple access, and initial access," *IEEE Commun. Mag.*, vol. 55, no. 6, pp. 64–71, 2017.

[46] S. G. Larew and D. J. Love, "Adaptive beam tracking with the unscented Kalman filter for millimeter wave communication," *IEEE Signal Process. Lett.*, vol. 26, no. 11, pp. 1658–1662, 2019.

[47] M. B. Booth, V. Suresh, N. Michelusi, and D. J. Love, "Multi-armed bandit beam alignment and tracking for mobile millimeter wave communications," *IEEE Commun. Lett.*, vol. 23, no. 7, pp. 1244–1248, 2019.

[48] J. Choi, D. J. Love, and P. Bidigare, "Downlink training techniques for FDD massive MIMO systems: Open-loop and closed-loop training with memory," *IEEE J. Sel. Topics Signal Process.*, vol. 8, no. 5, pp. 802–814, 2014.

[49] D. J. Love, R. W. Heath, V. K. Lau, D. Gesbert, B. D. Rao, and M. Andrews, "An overview of limited feedback in wireless communication systems," *IEEE J. Sel. Areas Commun.*, vol. 26, no. 8, pp. 1341–1365, 2008.

[50] D. J. Love, R. W. Heath, W. Santipach, and M. L. Honig, "What is the value of limited feedback for MIMO channels?" *IEEE Commun. Mag.*, vol. 42, no. 10, pp. 54–59, 2004.

[51] D. J. Love, R. W. Heath, and T. Strohmer, "Grassmannian beamforming for multiple-input multiple-output wireless systems," *IEEE Trans. Inform. Theory*, vol. 49, no. 10, pp. 2735–2747, 2003.

[52] X. Lyu, H. Tian, C. Sengul, and P. Zhang, "Multiuser joint task offloading and resource optimization in proximate clouds," *IEEE Trans. Veh. Technol.*, vol. 66, no. 4, pp. 3435–3447, 2016.

[53] M. Hong and Z.-Q. Luo, "Signal processing and optimal resource allocation for the interference channel," in *Academic Press Library in Signal Processing*. Elsevier, 2014, vol. 2, pp. 409–469.

[54] K. Shen and W. Yu, "Fractional programming for communication systems—part I: Power control and beamforming," *IEEE Trans. Signal Process.*, vol. 66, no. 10, pp. 2616–2630, 2018.

[55] S. He, Y. Huang, L. Yang, and B. Ottersten, "Coordinated multicell multiuser precoding for maximizing weighted sum energy efficiency," *IEEE Trans. Signal Process.*, vol. 62, no. 3, pp. 741–751, 2013.

[56] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[57] S. Sun, T. S. Rappaport, R. W. Heath, A. Nix, and S. Rangan, "MIMO for millimeter-wave wireless communications: Beamforming, spatial multiplexing, or both?" *IEEE Commun. Mag.*, vol. 52, no. 12, pp. 110–121, 2014.

[58] J. Zhang, Y. Wu, S. Zhou, and J. Wang, "Joint linear transmitter and receiver design for the downlink of multiuser MIMO systems," *IEEE Commun. Lett.*, vol. 9, no. 11, pp. 991–993, 2005.

[59] O. Arnold, F. Richter, G. Fettweis, and O. Blume, "Power consumption modeling of different base station types in heterogeneous cellular networks," in *Future Network & Mobile Summit*, 2010.

[60] S. Y. Seidel and T. S. Rappaport, "914 MHz path loss prediction models for indoor wireless communications in multifloored buildings," *IEEE Trans. Antennas Propag.*, vol. 40, no. 2, pp. 207–217, 1992.

[61] C. K. Au-Yeung and D. J. Love, "On the performance of random vector quantization limited feedback beamforming in a MISO system," *IEEE Trans. Wirel. Commun.*, vol. 6, no. 2, pp. 458–462, 2007.

[62] J. H. Ferziger, *Numerical methods for engineering application*. Wiley New York, 1981, vol. 1.

[63] S. M. Ross, *Introduction to probability models*. Academic press, 2014.

Junghoon Kim received the M.S. degree in electrical engineering from Korea Advanced Institute of Science and Technology (KAIST) in 2014. Since 2019, he has worked towards the Ph.D. degree at Purdue University.

Taejoon Kim (Senior Member, IEEE) received the Ph.D. degree in electrical and computer engineering from Purdue University in 2011. He is currently an Assistant Professor of electrical engineering and computer science at the University of Kansas (KU). His research interest includes 5G-and-beyond wireless systems and millimeter-wave and terahertz wireless. He was the recipient of the Miller Faculty Award from the KU School of Engineering and The IEEE Communications Society Stephen O. Rice Prize.

Morteza Hashemi is an Assistant Professor with the Department of Electrical Engineering and Computer Science at the University of Kansas, Lawrence, Kansas. He received his MSc and PhD degrees in Electrical Engineering from Boston University in 2013 and 2015, respectively. Before joining KU in 2019, he was a postdoctoral researcher and senior lecturer at the Ohio State University. His research interests span the areas of wireless communications, information systems, real-time data networking, and networked cyber-physical systems.

Christopher G. Brinton (Senior Member, IEEE) received the Ph.D. degree in electrical engineering from Princeton University in 2016. He is currently an Assistant Professor of electrical and computer engineering with Purdue University. Since joining Purdue University in 2019, he has won several awards including the Seed for Success Award and the Ruth and Joel Spira Outstanding Teacher Award.

David J. Love (S'98 - M'05 - SM'09 - F'15) is the Nick Trbovich Professor of Electrical and Computer Engineering at Purdue University. His research interests are in the design and analysis of broadband wireless communication systems, beyond-5G wireless systems, multiple-input multiple-output (MIMO) communications, millimeter wave wireless, software defined radios and wireless networks, coding theory, and MIMO array processing.

APPENDIX A
PROOF OF THEOREM 1

We first rewrite the problem (42)-(43) to an equivalent form by introducing an auxiliary variable $\gamma_k \in \mathbb{R}^{++}$ for $k \in K_{\text{Tx}}$ as

$$\text{minimize} \quad \sum_{k \in K_{\text{Tx}}} I_k \gamma_k \quad (67)$$

$$\text{subject to} \quad \frac{g_k(\mathbf{f}_k)}{u_k(\{\mathbf{f}_k\}, \mathbf{z}_{k',i}, w_k)} \leq \gamma_k, \quad (68)$$

$$\|\mathbf{f}_k\|_2^2 \leq P_k \quad \forall k \in \mathcal{K}_{\text{Tx}} \quad (69)$$

$$\text{variables} \quad \{\mathbf{f}_k\}, \{\mathbf{z}_{k',i}\}, \{w_k\}, \{\gamma_k\}.$$

Introducing the Lagrange multipliers $\{\lambda_k\}$ and $\{\mu_k\}$ for the two inequality constraints in (68)-(69), we obtain the Lagrange function $\mathcal{L}(\cdot)$ of the problem (67)-(69) as

$$\mathcal{L}(\{\mathbf{f}_k\}, \{\mathbf{z}_{k',i}\}, \{w_k\}, \{\gamma_k\}, \{\lambda_k\}, \{\mu_k\}) = \quad (70)$$

$$\sum_{k \in \mathcal{K}_{\text{Tx}}} I_k \gamma_k + \sum_{k \in \mathcal{K}_{\text{Tx}}} \lambda_k(g_k - \gamma_k u_k) + \sum_{k \in \mathcal{K}_{\text{Tx}}} \mu_k(\|\mathbf{f}_k\|_2^2 - P_k)$$

where $u_k(\{\mathbf{f}_k\}, \mathbf{z}_{k',i}, w_k)$ and $g_k(\mathbf{f}_k)$ are denoted as u_k and g_k for simplicity.

Assuming that $\{\tilde{\mathbf{f}}_k\}$, $\{\tilde{\mathbf{z}}_{k',i}\}$, $\{\tilde{w}_k\}$, and $\{\tilde{\gamma}_k\}$ are the solutions of the problem (67)-(69), they must satisfy the KKT conditions

$$\frac{\partial}{\partial \mathbf{f}_k} \mathcal{L}(\cdot) = \mathbf{0}, \quad \frac{\partial}{\partial \mathbf{z}_{k',i}} \mathcal{L}(\cdot) = \mathbf{0}, \quad \frac{\partial}{\partial w_k} \mathcal{L}(\cdot) = 0,$$

$$\frac{\partial}{\partial \gamma_k} \mathcal{L}(\cdot) = I_k - \lambda_k u_k = 0, \quad (71)$$

$$\lambda_k(g_k - \gamma_k u_k) = 0, \quad \mu_k(\|\mathbf{f}_k\|_2^2 - P_k) = 0, \quad (72)$$

$$g_k \leq \gamma_k u_k, \quad \|\mathbf{f}_k\|_2^2 \leq P_k, \quad (73)$$

$$\lambda_k \geq 0, \quad \mu_k \geq 0 \quad \forall k \in \mathcal{K}_{\text{Tx}}, \quad (74)$$

where (71)-(74) represent the conditions of stationarity, complementary slackness, primal feasibility, and dual feasibility.

From (38) and (44), $u_k \geq 0$ and $g_k > 0$. Furthermore, with the optimal solutions $\{\tilde{\mathbf{f}}_k\}$, $\{\tilde{\mathbf{z}}_{k',i}\}$, and $\{\tilde{w}_k\}$, the inequality $u_k > 0$ is guaranteed. Otherwise, it will drive the value of the objective function in (42) to infinity. Since $u_k > 0$, the last condition in (71) and first condition in (72) yield

$$\lambda_k = \frac{I_k}{u_k}, \quad \gamma_k = \frac{g_k}{u_k}. \quad (75)$$

Then, the remaining conditions, i.e., the first three conditions in (71), the second condition in (72), the second condition in (73), and the second condition in (74), are exactly the KKT conditions of the problem below:

$$\text{minimize} \quad \sum_{k \in K_{\text{Tx}}} \lambda_k(g_k - \gamma_k u_k) \quad (76)$$

$$\text{subject to} \quad \|\mathbf{f}_k\|_2^2 \leq P_k \quad \forall k \in \mathcal{K}_{\text{Tx}} \quad (77)$$

$$\text{variables} \quad \{\mathbf{f}_k\}, \{\mathbf{z}_{k',i}\}, \{w_k\}.$$

In summary, if $\{\tilde{\mathbf{f}}_k\}$, $\{\tilde{\mathbf{z}}_{k',i}\}$, $\{\tilde{w}_k\}$, and $\{\tilde{\gamma}_k\}$ are solutions of the problem (67)-(69), then $\{\tilde{\mathbf{f}}_k\}$, $\{\tilde{\mathbf{z}}_{k',i}\}$, and $\{\tilde{w}_k\}$ are solutions of the problem (76)-(77) while simultaneously satisfying (75). The contrary conclusion can be obtained in the opposite direction, which leads to the proof of Theorem 1. If $\{\tilde{\mathbf{f}}_k\}$, $\{\tilde{\mathbf{z}}_{k',i}\}$, and $\{\tilde{w}_k\}$ are solutions of the problem (76)-(77) and also simultaneously satisfy the system equations with $\tilde{\lambda}_k$ and $\tilde{\gamma}_k$ in (75), then $\{\tilde{\mathbf{f}}_k\}$, $\{\tilde{\mathbf{z}}_{k',i}\}$, $\{\tilde{w}_k\}$, $\{\tilde{\lambda}_k\}$, and $\{\tilde{\gamma}_k\}$ satisfy all of the KKT conditions (71)-(74). This means that $\{\tilde{\mathbf{f}}_k\}$, $\{\tilde{\mathbf{z}}_{k',i}\}$, $\{\tilde{w}_k\}$, and $\{\tilde{\gamma}_k\}$ are the solutions of (67)-(69). It follows that $\{\tilde{\mathbf{f}}_k\}$, $\{\tilde{\mathbf{z}}_{k',i}\}$, $\{\tilde{w}_k\}$ are optimal solutions of (42)-(43).

APPENDIX B
PROOF OF LEMMA 2

For computational complexity of the two proposed methods, we only need to compare how many combinations of the binary variables $\{a_{k,k'}\}$ and $\{b_{k,i}\}$ are addressed for optimization. Note that non-integer variables are optimized when the binary variables are given. First, we deal with the computational complexity of the semi-exhaustive search optimization. From the condition (22), each k must choose one k' where $k' \in \mathcal{K}$. If $k' \neq k$, we also must choose one i from the condition (23) where $i \in \mathcal{S}$. Then, we have $(K-1)S+1$ cases for each k. This is performed for every $k \in \mathcal{K}$, and we get $(KS-S+1)^K$ cases. Therefore, we have $\mathcal{O}((KS - S + 1)^K)$. Although condition (23) can reduce the total number of cases, we consider the worst case scenario for computational complexity.

For the computational complexity of the efficient alternate optimization, a few steps need to be described. As the first iteration of the greedy search, the number of pairs among all K nodes is $K(K-1)$. For each pair, we consider S cases from the subchannel allocation condition (23). Therefore, we obtain $K(K-1)S$ cases at the first iteration. At the second iteration, the transmit candidate set $\mathcal{K}_{\text{Tx}}$ is updated with $|\mathcal{K}_{\text{Tx}}| = K-2$. Note that $|\mathcal{K}_{\text{Rx}}| = K$. Then, we have $(K-2)(K-1)S$ cases. At the third iteration, we have $|\mathcal{K}_{\text{Tx}}| = K - 3$ or $K - 4$. If the larger case $|\mathcal{K}_{\text{Tx}}| = K - 3$ is considered as worst case scenario, total cases will be $(K-3)(K-1)S$. This would continue to $1 \cdot (K-1)S$. We can apply the upper bound and calculate the total combinations approximately as $(K-1)S \sum_{k=1}^{K} k = (K-1)K(K+1)S/2$. Therefore, we have $\mathcal{O}(K^3 S)$.

APPENDIX C
DERIVATION OF THE CLOSED-FORM SOLUTION FOR $\mathbf{f}_k$

To solve problem (50)-(51), for each $\mathbf{f}_k$ where $k \in \mathcal{K}_{\text{Tx}}$, we introduce the Lagrange multiplier ν_k for the inequality constraint (51) and obtain the Lagrangian function as

$$L(\mathbf{f}_k, \nu_k) = \lambda_k \beta_k \mathbf{f}_k^H \mathbf{f}_k - 2\lambda_k \gamma_k w_k^{-1} \text{Re}[\mathbf{z}_{k',i}^H \mathbf{H}_{k,k}^{(i)} \mathbf{f}_k] + \mathbf{f}_k^H \mathbf{\Sigma}_k \mathbf{f}_k + \nu_k(\mathbf{f}_k^H \mathbf{f}_k - P_k). \quad (78)$$

The first-order optimality condition yields

$$\lambda_k \beta_k \mathbf{f}_k - \lambda_k \gamma_k w_k^{-1} \mathbf{H}_{k,k'}^{(i)H} \mathbf{z}_{k',i} + \mathbf{\Sigma}_k \mathbf{f}_k + \nu_k \mathbf{f}_k = \mathbf{0}. \quad (79)$$

We can obtain the solution $\mathbf{f}_k$ as a function of ν_k to be

$$\mathbf{f}_k(\nu_k) = \lambda_k \gamma_k w_k^{-1} (\mathbf{\Sigma}_k + \nu_k \mathbf{I} + \lambda_k \beta_k \mathbf{I})^{-1} \mathbf{H}_{k,k'}^{(i)H} \mathbf{z}_{k',i}. \quad (80)$$

From complementary slackness, dual feasibility, and primal feasibility, the following conditions should be satisfied: $\nu_k(\mathbf{f}_k^H(\nu_k)\mathbf{f}_k(\nu_k) - P_k) = 0$, $\nu_k \geq 0$, and $\mathbf{f}_k^H(\nu_k)\mathbf{f}_k(\nu_k) \leq P_k$. In other words, if $\mathbf{f}_k^H(0)\mathbf{f}_k(0) \leq P_k$, then $\mathbf{f}_k(0)$ is the optimal solution. If $\mathbf{f}_k^H(0)\mathbf{f}_k(0) > P_k$, then $\mathbf{f}_k(\nu_k^\star)$ is the optimal solution where $\mathbf{f}_k^H(\nu_k^\star)\mathbf{f}_k(\nu_k^\star) = P_k$ should be satisfied for some $\nu_k^\star$. In this case, the remaining part is to determine $\nu_k^\star$. Using the eigendecomposition $\Psi_k\Lambda_k\Psi_k^H = \Sigma_k + \lambda_k\beta_k\mathbf{I}$ and denoting $\Phi_k = \Psi_k^H\mathbf{H}_{k,k'}^{(i)H}\mathbf{z}_{k',i}$, we can represent $\mathbf{f}_k^H(\nu_k)\mathbf{f}_k(\nu_k)$ as

$$\mathbf{f}_k^H(\nu_k)\mathbf{f}_k(\nu_k) = \left|\lambda_k\gamma_k w_k^{-1}\right|^2 \sum_{m=1}^{N_k} \frac{|[\Phi_k]_m|^2}{([\Lambda_k]_{m,m} + \nu_k)^2}, \quad (81)$$

where $[\Phi_k]_m$ denotes the m-th element of Φ_k and $[\Lambda_k]_{m,m}$ denotes the m-th diagonal element of Λ_k. The formula in (81) is a non-increasing function of ν_k. Therefore, a finite-interval one-dimensional search method such as the bisection method [62] can be used to determine $\nu_k^\star$ to satisfy $\mathbf{f}_k^H(\nu_k^\star)\mathbf{f}_k(\nu_k^\star) = P_k$.

APPENDIX D
ADDITIONAL EXPERIMENTAL RESULTS

A. Runtime Growth Rate of the Efficient Alternate Optimization

In Fig. 12, we consider the runtime growth rate of the efficient alternate optimization over the number of nodes compared to the theoretical complexity given in Section IV-D. We consider $S = 1$ subchannels and $N = 5$ antennas. The runtime at $K = 4$ is normalized to 1 as a basis, and the growth rate along the number of nodes is plotted. While the growth rate of the worse-case complexity derived in Lemma 2 is K^3, we observe a growth rate is lower than K^3 (approximately $K^{2.4}$ according to polynomial curve fitting).

Overall, we see that the efficient alternate optimization has polynomial time complexity, which is considered a desirable upper bound in algorithm design. We leave additional improvements in runtime, e.g., to linear time complexity, to future work. In this regard, it may be desirable to have a fully distributed implementation of the efficient alternate optimization. To do so, future work can consider distributed algorithms for the (i) beamforming and (ii) resource design sub-problems. For (i), one can investigate a version of MCOB where each node solves for its own beamformer and combiner, leveraging channel reciprocity, with network information provided by neighbors. For (ii), one can investigate a distributed consensus version of the greedy algorithm, where each node chooses its local candidate for link addition and the nodes collaboratively discover a final decision. The incurred communication overhead from such approaches can also be factored into the overall network overhead optimization.

B. Example of Task Offloading and Processing

In Fig. 13, we show a more detailed version of the result in Fig. 9, i.e., individual time and energy consumption incurred by each task for different values of β through our optimization. The size of each task and available CPU at each node are demonstrated in Fig. 13(a). Intuitively, the nodes ($k = 3, 4, 10$)
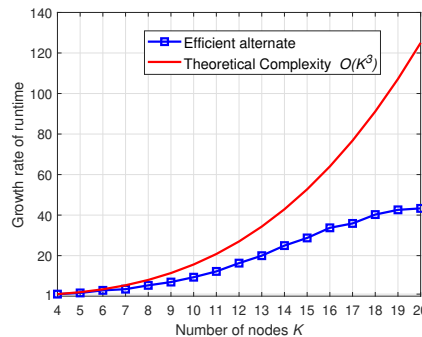
Fig. 12: The runtime growth rate of the efficient alternate optimization along the number of nodes K, for our experimental setting with $S = 1$ and $N = 5$. While the worst-case theoretical complexity is on the order of K^3, we observe a better growth rate in practice (approximately $K^{2.4}$ according to curve fitting).

with low CPU and large data size are more likely to offload tasks to other nodes ($k = 6, 8$) with high CPU as a result of our optimization. Given the computation and communication resources including subchannels, task size, and available CPU, we apply our efficient alternate optimization methodology for each $\beta = 0.2, 0.5, 0.8$, with the results summarized in Fig. 13(b)&(c). Fig. 13(b)&(c) break down the time and energy overhead incurred as a result of communication and computation for each task. For $\beta = 0.2$, tasks $k = 3, 4, 10$ are offloaded to nodes $k = 6, 8, 6$, respectively. For $\beta = 0.5, 0.8$, which increases the emphasis on minimizing energy consumption, tasks $k = 3, 4, 10$ are offloaded to nodes $k = 6, 8, 8$, respectively. The rest of the tasks are processed locally.

In order to explore how the time (latency) and energy consumption for processing the data are changed through task offloading, we take a look the specific behavior of task 4 offloaded to node 8 as an example. In Fig. 13(b), for any β, we see that the time overhead for task 4, including computation and communication overhead, is greatly reduced as compared to local task processing, because task 4 is offloaded to node 8 which has high CPU. This also leads to increasing the computation time overhead for task 8, because less CPU is allocated to it due to CPU sharing with the offloaded task 4. In Fig. 13(c), the energy consumption for processing task 4 increases as compared to local task processing, because a larger amount of CPU is used to process the offloaded task (at node 8). At the same time, the energy consumption for task 8 decreases because less CPU (at node 8) is now allocated to this task due to CPU sharing. Overall, task offloading is conducted to minimize the optimization objective, which is the weighted sum of total time and energy consumption for processing the data throughout the network. The overall performance in terms of the total time and energy consumption is shown in Fig. 9.

C. Dynamic Task Generation

In this experiment, we assess our proposed framework and algorithms under a dynamic task scenario. When a new task is generated at a node, we assume that it is queued until there is a time frame available for it to be processed. Then, the optimization proceeds as a series of frames, where each frame will consider one task from each node in a first in first out

(a) The size of task k and available CPU at node k.

(b) Time delay (latency) for processing each task.

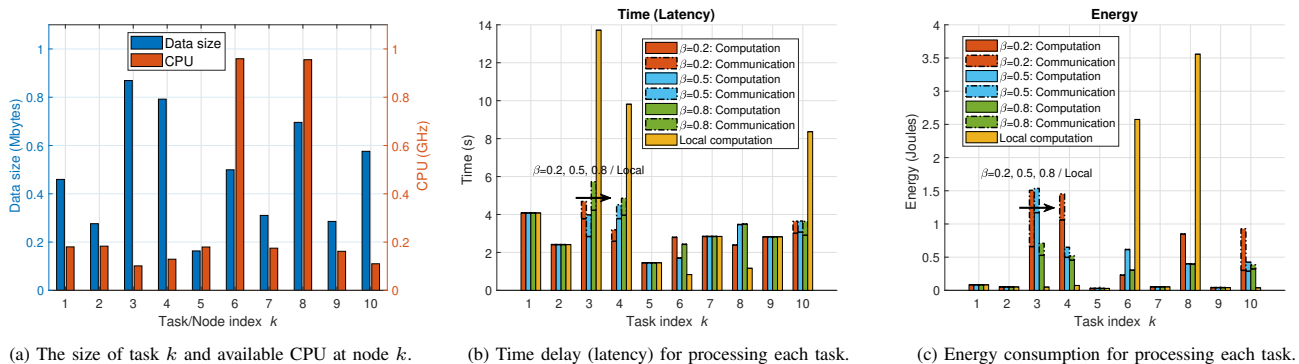(c) Energy consumption for processing each task.

Fig. 13: Specific breakdown of task offloading and processing for the experiment in Fig. 9. From (a), intuitively, the nodes ($k = 3, 4, 10$) with low CPU and large data size are more likely to offload their tasks to other nodes ($k = 6, 8$) with high CPU. After the efficient alternate optimization (Algorithm 3) is executed for each $\beta = 0.2, 0.5, 0.8$, the individual time delay and energy consumption values incurred by each task are depicted in (b) and (c).
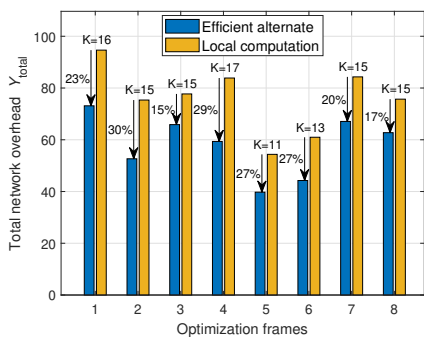

Fig. 14: Total network overhead over multiple optimization frames under a dynamic task generation scenario. The total number of nodes in the D2D network is $K_{\max} = 30$, where some of nodes will participate in each optimization frame if they have a task in their queue. Task generation is assumed to follow a Poisson distribution at each node, with task arrival rate $\phi = 1/10$ (tasks/s).

stochastic optimization to capture dynamic task generation over a long-term time horizon.

(FIFO) manner. Since nodes will not always have tasks in their queue, let $\mathcal{K}(t)$ denote the set of participating nodes in optimization frame t. In Fig. 14, we show a simulation which considers the total network overhead obtained by our efficient alternate optimization algorithm over $t = 1, ..., 8$ optimization frames, where the total nodes is $K_{max} = 30$ and $\mathcal{K}(t)$ varies from tasks being generated according to a standard $M/M/1$ queue model [63]. For the $M/M/1$ queue model, Poisson distributed traffic is considered with task arrival rate $\phi = 1/10$ (tasks/sec), and the traffic is assumed to be independent across nodes. We consider the period of each optimization frame to be $T_s = 5$ sec. The task size is assumed to be $I_k = 1$ Mbytes for each node. In every optimization frame t, the number of participating nodes, $K(t)$, is indicated above the bars in Fig. 14. Overall, the total network overhead is reduced substantially in each time period compared to local computation: between 15% and 30%. This is consistent with our experiments for the non-dynamic case.

The above experiment shows that our framework and algorithms can be applied in the dynamic task setting. Moreover, extensions of the approach we have presented here are noteworthy for future work. For example, optimizing the frame rate T_s over time may provide additional gains in terms of task processing delay and energy efficiency. Doing so would require