Coded Computing and Cooperative Transmission for Wireless Distributed Matrix Multiplication

Kuikui Li, Meixia Tao, Jingjing Zhang, and Osvaldo Simeone

Abstract

Consider a multi-cell mobile edge computing network, in which each user wishes to compute the product of a user-generated data matrix with a network-stored matrix. This is done through task offloading by means of input uploading, distributed computing at edge nodes (ENs), and output downloading. Task offloading may suffer long delay since servers at some ENs may be straggling due to random computation time, and wireless channels may experience severe fading and interference. This paper aims to investigate the interplay among upload, computation, and download latencies during the offloading process in the high signal-to-noise ratio regime from an information-theoretic perspective. A policy based on cascaded coded computing and on coordinated and cooperative interference management in uplink and downlink is proposed and proved to be approximately optimal for a sufficiently large upload time. By investing more time in uplink transmission, the policy creates data redundancy at the ENs, which can reduce the computation time, by enabling the use of coded computing, as well as the download time via transmitter cooperation. Moreover, the policy allows computation time to be traded for download time. Numerical examples demonstrate that the proposed policy can improve over existing schemes by significantly reducing the end-to-end execution time.

Index Terms

Matrix Multiplication, Straggler, Edge Computing, Transmission Cooperation, Coded Computing

I. INTRODUCTION

Motivation and scope: Mobile edge computing (MEC) is an emerging network architecture that enables cloud-computing capabilities at the edge nodes (ENs) of mobile networks [2]–[4]. Through task offloading, MEC makes it possible to offer mobile users intelligent applications, such as recommendation systems or gaming services, that would otherwise require excessive on-device storage and computing resources. Deploying task offloading, however, poses non-trivial design problems. On one hand, task offloading may require a large amount of data to be transferred between users and ENs over uplink or downlink channels, which may suffer severe channel fading and interference conditions, resulting in large communication latencies. On the other hand, edge servers are likely to suffer from the straggling effect, yielding unpredictable computation delays [5]. A key problem in MEC networks, which is the subject of this paper, is to understand the interplay and performance trade-offs between two-way communication (in both uplink and downlink) and computation during the offloading process.
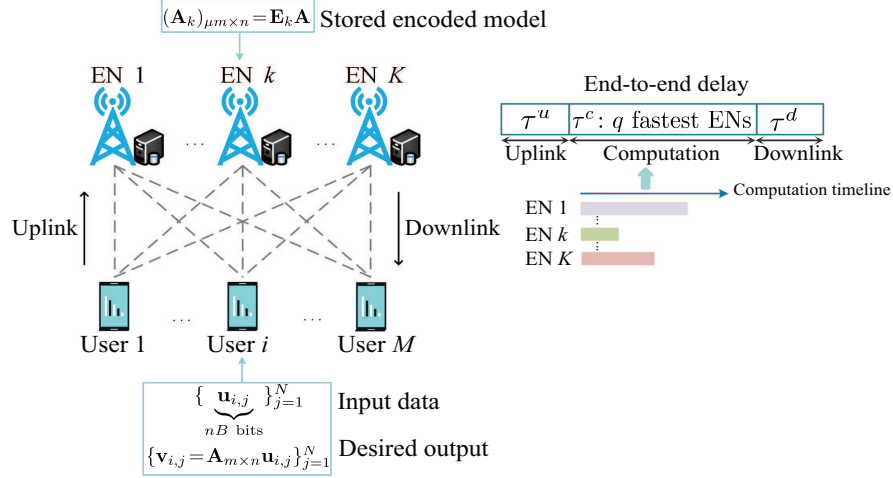
Fig. 1. A multi-cell MEC network carrying out distributed matrix multiplication via uplink communication, edge computing, and downlink communications.

To this end, this study focuses on the baseline problem of computing the product between user-generated data vectors $\{\mathbf{u}\}$ and a network-stored matrix $\mathbf{A}$. Matrix multiplication is a representative computation task that underlies many machine learning and data analytic problems. Examples of applications include recommendation systems based on collaborative filtering [6], in which the user-generated data $\{\mathbf{u}\}$ corresponds to user profile vectors, while the network-side matrix $\mathbf{A}$ collects the profile vectors of a certain class of items, e.g., movies. Matrix $\mathbf{A}$ is generally very large in practice, preventing a simple solution whereby users download and store the matrix for local computation.

Matrix multiplication, as many other more complex computations [7], can be decomposed into subtasks and distributedly computed across multiple servers. In MEC networks, the servers are embedded in distinct ENs, and hence distributed computing at the edge requires input data uploading via the uplink, computation at the ENs, and output data downloading via the downlink. A fundamental question that this work tackles is: *What is the minimum achievable upload-compute-download latency triplet for completing matrix multiplication in the presence of straggling servers and multi-cell interference?*

In the task offloading process discussed above, the overall latency is the sum of three components, namely the time needed for input uploading, server computing, and output downloading. This paper is devoted to studying the interplay and trade-offs among these three components from an information-theoretic standpoint. A key result that will be illustrated by our results is that investing more time in any one of the three steps may be instrumental in reducing the time needed for subsequent steps thanks to coded computing [8]–[19] and cooperative transmission [20]–[24]. As explained next, both coded computing and cooperative transmissions leverage forms of computation redundancy.

Background and related works: Coded computing was introduced in [9] for a master-slave system with ideal communication links and linear computations. The approach aims at reducing the average latency caused by distributed servers with random computation time, hence mitigating the problem of *straggling servers* [5], through linear coding of the rows of matrix $\mathbf{A}$. Linear coding assigns each server a flexible number of encoded rows of matrix $\mathbf{A}$. Thanks to maximum distance separable (MDS) coding, assigning more coded rows at the servers reduces the number of servers that need to complete their computations in order to recover the desired outputs [10]–[12]. Coded computing was introduced in [13] as a means to speed up the computation of

distributed matrix multiplication in a MEC system, providing a starting point for this work.

A simple way to ensure computation redundancy is to assign repeatedly the same rows of matrix $\mathbf{A}$ across multiple ENs. While this does not provide the same robustness against stragglers as MDS coding, it allows ENs to compute common outputs, i.e., computation replication, as proposed in [25]. This in turn makes it possible for the ENs to cooperate for transmission to the users in the downlink, which can reduce the download latency in an interference-limited system such as multi-user multi-server MEC systems shown in Fig. 1. This form of cooperative transmission enabled by computation redundancy has been explored by [13], [25], [26] for task offloading in multi-cell MEC systems and by [27] for data shuffling in wireless MapReduce systems, all with the goal of mitigating the multi-cell interference and hence boost the communication efficiency. Cooperative transmission has also been explored in the context of multi-cell caching systems in [20]–[24] to accelerate content delivery by caching overlapped contents at different ENs.

Overview and main contributions: In the MEC system of Fig. 1, investing more time for uplink communication allows the same user-generated input vectors to be received by more ENs, which enhances computation redundancy. The computation redundancy generally introduces a heavier computation load, which can in turn increase the robustness against straggling servers via coded computing and mitigate multi-cell downlink interference via cooperative transmission. Based on these observations, this paper aims to establish the optimal trade-off between computing and download latencies at any given upload latency. We focus on the high signal-to-noise ratio (SNR) regime in order to highlight the role of interference management as enabled by computation redundancy.

The most related prior works, as reviewed above, are [13] and [25]. The work [13] proposes a computing and downloading strategy by making the simplified assumption that the upload time is unconstrained so that the input vectors from all users are available at all ENs. The work [25] characterizes the trade-off between upload and download latencies by assuming that the computation time at each EN is deterministic (in contrast to random) so that coded computing is not needed. Moreover, reference [25] adopts a general task model, rather than matrix multiplication as studied in this work. In contrast to [13] and [25], in this paper, we study the joint design of task assignment, input upload, edge computing, and output download, and we analyze the performance trade-offs among upload, computing and download latencies.

In summary, this paper studies the communication (in both uplink and downlink) and computation tradeoff in multi-user multi-server MEC networks by enabling the use of coded computing and cooperative transmission. The main contributions are as follows:

- We propose a new task offloading strategy that integrates coded computing based on a cascade of MDS and repetition codes [11] with cooperative transmission at the ENs for interference management [22]. By uploading the same input vectors of a user to multiple ENs, the policy creates data redundancy at the ENs that is leveraged to reduce the computation time by coded computing and the download time via transmission cooperation. Moreover, by waiting for more non-straggling ENs to finish their tasks, the proposed policy enhances the downlink transmission cooperation opportunities, and hence it allows the computation time to be traded for download time.

- We derive achievable upload-compute-download latency triplets, as well as the end-to-end execution time, and characterize the trade-off region between computing and download latencies at any given upload latency. The analysis of upload and download latencies relies respectively on the analysis of degrees of freedom (DoF) for the effective X-multicast channel formed during uplink transmission and for the cooperative X channels obtained during downlink transmission [22].

- Furthermore, we provide a converse result that demonstrates the optimality of the achievable upload latency for fixed computation and download latencies, as well as constant multiplicative gaps to their respective lower bounds for computation and download latencies at a large upload latency. The proof is based on genie-aided arguments and on a generalization of the arguments in [21]. The end-to-end execution time is also proved to be order-optimal for a large upload latency.
- Through numerical examples, we show that, as compared to baseline schemes, the proposed policy can reduce the overall end-to-end execution time. We also show that, when the downlink transmission is the major bottleneck of the offloading process, the proposed cascaded MDS-repetition coding scheme reduces to repetition coding with no loss of optimality; while, when the bottleneck comes from the uplink transmission or edge computing, MDS coding is required to mitigate the effect of straggling ENs.

The rest of the paper is organized as follows. Section II presents the problem formulation and definitions. Main results including communication-computation latency trade-offs are presented in Section III. The proposed scheme is detailed in Section IV. Section V provides numerical examples. Conclusions are drawn in Section VI. The converse proof is available in Appendix.

Notations: $\mathcal{K}$ denotes the set of indexes $\{1, 2, \cdots, K\}$. $[a:b]$ denotes the set of integers $\{a+1, a+2, \ldots, b\}$. $[a]$ denotes the set of integers $[1:a]$. $(\cdot)^T$ denotes the transpose. $(x)^+$ denotes $\max\{x, 0\}$. $(X_i)_{i=a}^b$ denotes the vector $(X_a, X_{a+1}, \cdots, X_b)^T$. $\{x_i : i \in [a:b]\}$ or $\{x_i\}_{i=a}^b$ denotes the set $\{x_a, x_{a+1}, \cdots, x_b\}$. $\{x_k\}_{q:K}$ denotes the q-th smallest element of set $\{x_k : k \in [K]\}$. $\mathbb{F}_{2^B}^{m \times n}$ denotes the set of all matrices of dimension $m \times n$ with entries in the finite field $\mathbb{F}_{2^B}$ [28].

II. Problem Formulation

A. MEC Network Model

As shown in Fig. 1, we consider a multi-cell MEC network consisting of K single-antenna ENs communicating with M single-antenna users via a shared wireless channel. Denote by $\mathcal{K} = \{1, 2, \ldots, K\}$ the set of ENs and by $\mathcal{M} = \{1, 2, \ldots, M\}$ the set of users. Each EN is equipped with an edge server. The mobile users offload their computing tasks to the ENs through the uplink channel (from users to ENs) and then download the computation results back via the downlink channel (from ENs to users). Let h_{ki}^{u} denote the uplink channel fading from user $i \in \mathcal{M}$ to EN $k \in \mathcal{K}$, and h_{ik}^{d} denote the downlink channel fading from EN $k \in \mathcal{K}$ to user $i \in \mathcal{M}$, both of which are independent and identically distributed (i.i.d.) for all pairs (i, k) according to some continuous distribution. A central scheduling unit (CSU) is connected to all nodes via backhaul links to collect the uplink channel state information (CSI) $\mathbf{H}^{\mathrm{u}} \triangleq \{h_{ki}^{\mathrm{u}} : k \in \mathcal{K}, i \in \mathcal{M}\}$ and downlink CSI $\mathbf{H}^{\mathrm{d}} \triangleq \{h_{ik}^{\mathrm{d}} : i \in \mathcal{M}, k \in \mathcal{K}\}$ estimated at these nodes. It utilizes the collected global CSI to design the transmit or receive beamforming coefficients for the symbols transmitted or received at the nodes. We assume perfect CSI for uplink and downlink channels at the CSU, and we refer to [29]–[31] for analysis of the impact of imperfect CSI in the high-SNR regime.

We consider that each user has a matrix multiplication task to compute. Matrix multiplication is a building block of many machine learning and data analytic problems, e.g., linear inference tasks including collaborative filtering for recommendation systems [6]. Specifically, we assume that each user i has an input matrix with N input vectors[1] $\mathbf{u}_{i,j} \in \mathbb{F}_{2^B}^{n \times 1}$, $j \in [N]$, and it wishes to compute the output matrix with N output vectors $\mathbf{v}_{i,j} \in \mathbb{F}_{2^B}^{m \times 1}$, $j \in [N]$, where

$$\mathbf{v}_{i,j} = \mathbf{A}\mathbf{u}_{i,j}, \text{ for } j \in [N], \tag{1}$$

---

[1]The same number of input vectors for different users is assumed for analytical tractability. When this is not the case, a simple, generally suboptimal, solution is to add extra inputs (e.g., zero vectors) to each user to make their inputs equal in number.

and $\mathbf{A} \in \mathbb{F}_{2B}^{m \times n}$ is a data matrix available at the network end, and B is the size (in bits) of each element. The matrix $\mathbf{A}$ is partially stored across the ENs, which conduct the product operations in a distributed manner. To this end, each EN k has a storage capacity of $\mu m n B$ bits, and it can hence store a fraction $\mu \in [\frac{1}{K}, 1]$ of the rows of matrix $\mathbf{A}$. Specifically, during an offline storage phase, an encoding matrix $\mathbf{E}_k \in \mathbb{F}_{2B}^{\mu m \times m}$ is used to generate a coded matrix $\mathbf{A}_k = \mathbf{E}_k \mathbf{A}$, which is then stored at EN k, as in [10], [11].

B. Task Offloading Procedure

The task offloading procedure proceeds through task assignment, input uploading, edge computing, and output downloading.

1) Task Assignment: A task assignment scheme is defined through the following sets

$$\{\mathcal{U}_{i,\mathcal{K}'} : i \in \mathcal{M}, \mathcal{K}' \subseteq \mathcal{K}\}, \tag{2}$$

where $\mathcal{U}_{i,\mathcal{K}'} \subseteq \{\mathbf{u}_{i,j}\}_{j=1}^N$ denotes the subset of input vectors from user i that are assigned only to the subset of ENs $\mathcal{K}'$ for computation. We impose the condition $\bigcup_{\mathcal{K}' \subseteq \mathcal{K}} \mathcal{U}_{i,\mathcal{K}'} = \{\mathbf{u}_{i,j}\}_{j=1}^N$ for $i \in \mathcal{M}$ to guarantee that all input vectors are computed. Furthermore, by definition, we have the relation $\mathcal{U}_{i,\mathcal{K}'} \bigcap \mathcal{U}_{i,\mathcal{K}''} = \varnothing$ for $\mathcal{K}' \neq \mathcal{K}''$ so that these subsets are not overlapped. The subset of input vectors from all users assigned to each EN k is hence given as $\mathcal{U}_k = \bigcup_{i \in \mathcal{M}, \mathcal{K}' \subseteq \mathcal{K}: k \in \mathcal{K}'} \mathcal{U}_{i,\mathcal{K}'}$.

Definition 1. *(Repetition Order) For a given task assignment scheme $\{\mathcal{U}_{i,\mathcal{K}'}\}_{i \in \mathcal{M}, \mathcal{K}' \subseteq \mathcal{K}}$, the repetition order r, with $1 \leq r \leq K$, is defined as average input data redundancy, i.e., the total number of input vectors assigned to the K ENs (counting repetitions) divided by the total number of input vectors of the M users, i.e.,*

$$r \triangleq \frac{\sum_{k \in \mathcal{K}} |\mathcal{U}_k|}{MN}. \tag{3}$$

The repetition order indicates the average number of ENs that are assigned the same input vector, which has been adopted in [25] as a measure of the degrees of computation replication. The above task assignment $\{\mathcal{U}_{i,\mathcal{K}'}\}$ is realized through the following input uploading phase.

2) Input Uploading: At run time, each user i maps its input vectors $\{\mathbf{u}_{i,j}\}_{j=1}^N$ into a codeword $\mathbf{X}_i^{\mathrm{u}} \triangleq (X_i^{\mathrm{u}}(t))_{t=1}^{T^{\mathrm{u}}}$ of length T^{u} symbols under the power constraint $(T^{\mathrm{u}})^{-1} \mathbb{E}[||\mathbf{X}_i^{\mathrm{u}}||^2] \leq P^{\mathrm{u}}$. Note that $X_i^{\mathrm{u}}(t) \in \mathbb{C}$ is the symbol transmitted at time $t \in [T^{\mathrm{u}}]$. At each EN $k \in \mathcal{K}$, the received signal $Y_k^{\mathrm{u}}(t) \in \mathbb{C}$ at time $t \in [T^{\mathrm{u}}]$ can be expressed as

$$Y_k^{\mathrm{u}}(t) = \sum_{i \in \mathcal{M}} h_{ki}^{\mathrm{u}}(t) X_i^{\mathrm{u}}(t) + Z_k^{\mathrm{u}}(t), \tag{4}$$

where $Z_k^{\mathrm{u}}(t) \sim \mathcal{CN}(0,1)$ denotes the noise at EN k. Each EN k decodes the sequence $(Y_k^{\mathrm{u}}(t))_{t=1}^{T^{\mathrm{u}}}$ into an estimate $\{\widehat{\mathbf{u}}_{i,j}\}$ of the assigned input vectors $\{\mathbf{u}_{i,j} : \mathbf{u}_{i,j} \in \mathcal{U}_k\}$.

3) Edge Computing: After the uploading phase is completed, each EN k computes the products of the assigned estimated input vectors in set $\mathcal{U}_k$ with its stored coded model $\mathbf{A}_k$. The computation time for EN k to complete the computation of the corresponding $\mu m |\mathcal{U}_k|$ row-vector products[2] is modeled as

$$T_k^{\mathrm{c}} = \mu m |\mathcal{U}_k| \omega_k, \text{ for } k \in \mathcal{K}, \tag{5}$$

where the random variable ω_k represents the time needed by EN k to compute a row-vector product, and it is modelled as an exponential distribution with mean $1/\eta$ (see, e.g., [9], [10], [12], [14]). T_k^{c} is thus a scaled exponential distribution with mean $\mu m |\mathcal{U}_k|/\eta$. The MEC network

---

[2]The row-vector product indicates the product of a row vector of matrix $\mathbf{A}$ with a column vector $\mathbf{u}_{i,j}$.

waits until the fastest q ENs, denoted as subset $\mathcal{K}_q \subseteq \mathcal{K}$, have finished their tasks before returning the results back to users in the downlink. The cardinality $|\mathcal{K}_q| = q$ is referred to as *the recovery order*. The rest of $K - q$ ENs are known as *stragglers*. The resulting (random) duration of the edge computing phase is hence equal to the maximum computation time of the q fastest ENs, i.e., $T^{\mathrm{c}} = \max_{k \in \mathcal{K}_q} T_k^{\mathrm{c}}$.

4) Output Downloading: At the end of the edge computing phase, each EN $k \in \mathcal{K}_q$ obtains the coded outputs $\mathcal{V}_k \triangleq \{\mathbf{v}_{i,j,k} = \mathbf{A}_k \widehat{\mathbf{u}}_{i,j} : \mathbf{u}_{i,j} \in \mathcal{U}_k\}$. Every EN k in $\mathcal{K}_q$ then maps $\mathcal{V}_k$ into a length-T^{d} codeword $\mathbf{X}_k^{\mathrm{d}} \triangleq \left(X_k^{\mathrm{d}}(t)\right)_{t=1}^{T^{\mathrm{d}}}$ with an average power constraint $(T^{\mathrm{d}})^{-1} \mathbb{E}\left[||\mathbf{X}_k^{\mathrm{d}}||^2\right] \leq P^{\mathrm{d}}$. For each user $i \in \mathcal{M}$, its received signal $Y_i^{\mathrm{d}}(t) \in \mathbb{C}$ at time $t \in [T^{\mathrm{d}}]$ is given by

$$Y_i^{\mathrm{d}}(t) = \sum_{k \in \mathcal{K}_q} h_{ik}^{\mathrm{d}}(t) X_k^{\mathrm{d}}(t) + Z_i^{\mathrm{d}}(t), \tag{6}$$

where $Z_i^{\mathrm{d}}(t) \sim \mathcal{CN}(0,1)$ is the noise at user i. Each user i decodes the sequence $(Y_i^{\mathrm{d}}(t))_{t=1}^{T^{\mathrm{d}}}$ to obtain an estimate $\{\widehat{\mathbf{v}}_{i,j,k}\}_{j \in [N], k \in \mathcal{K}_q}$ of the coded outputs, from which it obtains an estimate $\{\widehat{\mathbf{v}}_{i,j}\}_{j \in [N]}$ of its desired outputs. This is possible if the estimated coded outputs $\{\widehat{\mathbf{v}}_{i,j,k}\}_{j \in [N], k \in \mathcal{K}_q}$ contain enough information to guarantee the condition $H(\{\mathbf{v}_{i,j}\}_{j \in [N]} | \{\widehat{\mathbf{v}}_{i,j,k}\}_{j \in [N], k \in \mathcal{K}_q}) = 0$. The overall error probability is given as $\mathrm{P}_{\mathrm{e}} \triangleq \mathbb{P}\left(\bigcup_{i=1,j=1}^{M \quad N} \{\widehat{\mathbf{v}}_{i,j} \neq \mathbf{v}_{i,j}\}\right)$. A task offloading policy is said to be feasible when the error probability $\mathrm{P}_{\mathrm{e}} \to 0$ as $B \to \infty$.

C. Performance Metric

The performance of the considered MEC network is characterized by the latency triplet accounting for task uploading, computing, and output downloading, which we measure in the high-SNR regime as defined below.

Definition 2. *The normalized uploading time (NULT), normalized computation time (NCT), and normalized downloading time (NDLT) achieved by a feasible policy with repetition order r and recovery order q are defined, respectively, as*

$$\tau^{\mathrm{u}}(r) \triangleq \lim_{P^{\mathrm{u}} \to \infty} \lim_{B \to \infty} \frac{\mathbb{E}_{\mathbf{H}^{\mathrm{u}}}[T^{\mathrm{u}}]}{NnB / \log P^{\mathrm{u}}}, \tag{7}$$

$$\tau^{\mathrm{c}}(r, q) \triangleq \lim_{m \to \infty} \frac{\mathbb{E}_{\boldsymbol{\omega}}[T^{\mathrm{c}}]}{Nm / \eta}, \tag{8}$$

$$\tau^{\mathrm{d}}(r, q) \triangleq \lim_{P^{\mathrm{d}} \to \infty} \lim_{m \to \infty} \lim_{B \to \infty} \frac{\mathbb{E}_{\mathbf{H}^{\mathrm{d}}}[T^{\mathrm{d}}]}{NmB / \log P^{\mathrm{d}}}. \tag{9}$$

The definitions (7) and (9) have been also adopted in [25], and follow the approach introduced in [21] by normalizing the delivery times to those of reference interference-free systems (with high-SNR rates $\log P^{\mathrm{u}}$ and $\log P^{\mathrm{d}}$, respectively). Similarly, the computation time in definition (8) is normalized by the average time needed to compute over all the input vectors of a user. To avoid rounding complications, in definition (8) and (9), we let the output dimension m grow to infinity.

Definition 3. *Given the definition of achievable NULT-NCT-NDLT triplet $\left(\tau^{\mathrm{u}}(r), \tau^{\mathrm{c}}(r, q), \tau^{\mathrm{d}}(r, q)\right)$ with repetition order r and recovery order q as in Definition 2, the optimal compute-download latency region for a given NULT τ^{u} is defined as the union of all NCT-NDLT pairs $(\tau^{\mathrm{c}}, \tau^{\mathrm{d}})$ that*
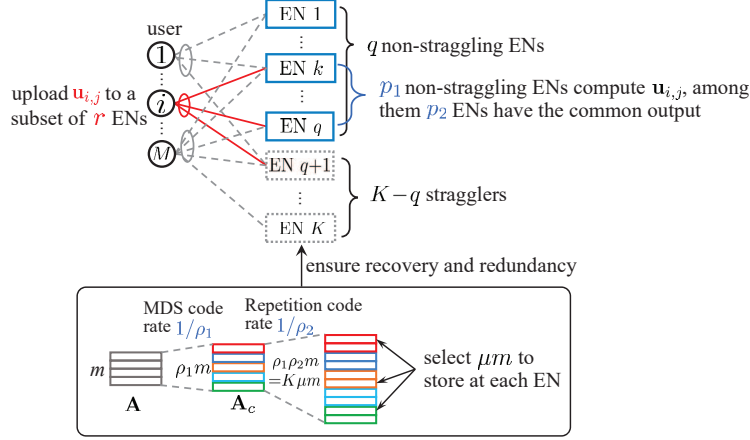
Fig. 2. (Bottom) Hybrid MDS-Repetition coding for matrix $\mathbf{A}$; (Top) Input uploading and edge computing.

satisfy $\tau^{\text{c}} \geq \tau^{\text{c}}(r, q)$ and $\tau^{\text{d}} \geq \tau^{\text{d}}(r, q)$ for some (r, q) while the corresponding NULT $\tau^{\text{u}}(r)$ is no larger than τ^{u}, i.e.,

$$\mathscr{T}^*(\tau^{\text{u}}) \triangleq \left\{ (\tau^{\text{c}}, \tau^{\text{d}}) : \left(\tau^{\text{u}}(r), \tau^{\text{c}}(r, q), \tau^{\text{d}}(r, q) \right) \text{ is achievable for some } (r, q) \text{ and } \tau^{\text{u}}(r) \leq \tau^{\text{u}}, \right.$$

$$\left. \tau^{\text{c}}(r, q) \leq \tau^{\text{c}}, \text{ and } \tau^{\text{d}}(r, q) \leq \tau^{\text{d}} \right\}. \quad (10)$$

Definition 4. *(End-to-end execution time) For a given pair (r, q), based on the defined communication and computation latency triplet, the end-to-end execution time is defined as the weighted sum of the NULT, NCT, and NDLT as*

$$\tau(r, q) = \tau^{\text{u}}(r) + \delta_c \tau^{\text{c}}(r, q) + \delta_d \tau^{\text{d}}(r, q). \quad (11)$$

In (11), $\delta_c = \frac{Nm/\eta}{NnB/\log P^{\text{u}}}$ represents the ratio between the reference time needed to compute over all the input vectors of a user and the reference time needed to upload all the input vectors of a user, while $\delta_d = \frac{NmB/\log P^{\text{d}}}{NnB/\log P^{\text{u}}}$ is ratio between the reference time needed to download all the output vectors of a user and the mentioned upload reference time.

Remark 1. *(Convexity of compute-download latency region.) For an input data assignment policy $\{\mathcal{U}_{i,\mathcal{K}'}\}_{i \in \mathcal{M}, \mathcal{K}' \subseteq \mathcal{K}}$ with repetition order r, fix an input uploading strategy achieving an NULT of τ^{u}. Consider now two policies π_1 and π_2 that differ may in their computing and download phases, and achieve two NCT-NDLT pairs $(\tau_1^{\text{c}}, \tau_1^{\text{d}})$ and $(\tau_2^{\text{c}}, \tau_2^{\text{d}})$, respectively. For any ratio $\lambda \in [0, 1]$, it can be seen that there exists a policy that achieves the NCT-NDLT pair $\lambda (\tau_1^{\text{c}}, \tau_1^{\text{d}}) + (1 - \lambda) (\tau_2^{\text{c}}, \tau_2^{\text{d}})$ for the same NULT τ^{u}. To this end, assuming m is sufficiently large, matrix $\mathbf{A}$, correspondingly, all output vectors in (1) are split horizontally so that $N\lambda m$ and $N(1 - \lambda)m$ outputs can be processed by using policies π_1 and π_2, respectively. By the linearity of the NCT in (8) and NDLT in (9) with respect to the output size, the claimed pair of NCT and NDLT is achieved. Thus, the region in (10) is convex. Similar arguments were also used in [21, Lemma 1].*

Based on the remark above, the region $\mathscr{T}^*(\tau^{\text{u}})$ in (10) is convex thanks to the time- and memory-sharing arguments, while it can be proved that the same is not true for the region of achievable triplets $(\tau^{\text{u}}, \tau^{\text{c}}, \tau^{\text{d}})$. Region $\mathscr{T}^*(\tau^{\text{u}})$ will be adopted to capture the trade-offs between computation and download latencies for a fixed upload latency. Our general goals are to characterize the minimum communication-computation latency triplet, the optimal tradeoff region between computing and download latencies, as well as the minimum end-to-end execution time.

III. MAIN RESULTS

In this section, we introduce a novel task offloading scheme based on the joint design of task assignment, two-way communication, and cascaded coded computing. Then, we study the communication-computation latency triplet and the end-to-end execution time achieved by this scheme. We derive inner and outer bounds on the compute-download latency region, and then discuss some consequences of the main results in terms of the tradeoffs among upload, computation, and download latencies. Furthermore, we specialize the main results to a number of simpler set-ups in order to illustrate the connections with existing works.

A. Key Ideas

We start by outlining the main ideas that underpin the proposed scheme. In task assignment, we choose a repetition-recovery order pair (r, q) from a feasible set $\mathcal{R}$ of values. We demonstrate that for any $(r, q) \in \mathcal{R}$, it is possible to recover all outputs through a suitable design of the system. For any such pair (r, q), as shown in Fig. 2-(bottom), matrix $\mathbf{A}$ is encoded by a cascade of an MDS code of rate $1/\rho_1$ and a repetition code of rate $1/\rho_2$. Of the encoded rows, μm different rows are stored at each EN, with each MDS encoded row replicated at ρ_2 distinct ENs. As we will prove, the MDS code can alleviate the impact of stragglers on the computation latency by decreasing the admissible values for the number q of non-straggling ENs (see also [9]–[11]); while repetition coding can reduce the download latency by enabling cooperative transmission among multiple ENs computing the same outputs [13], [25].

In the input upload phase, each user divides its N input vectors into $\binom{K}{r}$ subsets $\{\mathcal{U}_{i,\mathcal{K}'}\}$, with each subset uploaded to all the r ENs in subset $\mathcal{K}'$ for computation. By using interference alignment (IA), at each EN, a total of $M\binom{K-1}{r-1}$ desired subsets of inputs can be successfully decoded with the other $M\binom{K-1}{r}$ undesired subsets of interfering signals being aligned together. Then, as shown in Fig. 2-(top), in the computing phase, each input vector of any user is computed by a subset of p_1 non-straggling ENs with p_1 being at least $r - (K - q)$ and at most $\min\{r, q\}$. Therefore, since each encoded row of $\mathbf{A}$ is replicated at a subset of ρ_2 ENs, after computation, each MDS-encoded row-vector product result for a user will be replicated at a subset of p_2 non-straggling ENs, with p_2 being at least $\max\{\rho_2 - K + p_1, 1\}$ and at most $\min\{p_1, \rho_2\}$.

In the output download phase, each subset of p_2 ENs computing the same coded outputs can first use zero-forcing (ZF) precoding to null the interfering signal caused by common outputs at a subset of $p_2 - 1$ undesired users. When the number of undesired users does not exceed $p_2 - 1$, i.e., when $M - 1 \leq p_2 - 1$, by ZF precoding, each user only receives its desired outputs with all undesired outputs being cancelled out. When this condition is violated, i.e., when $M > p_2$, after ZF precoding, each output still causes interferences to $M - p_2$ undesired users. As detailed in [22], IA can be applied in cascade to the ZF precoders in order to mitigate the impact of these interfering signals.

B. Bounds

The scheme summarized above and detailed in Sec. IV achieves the following latency region.

Theorem 1. *(Inner bound). For the described MEC network with M users and K ENs, each with storage capacity $\mu \in [\frac{1}{K}, 1]$, the following communication-computation latency triplet $\big(\tau_a^u(r),$ $\tau_a^c(r, q), \tau_a^d(r, q)\big)$ is achievable*

$$\tau_a^u(r) = \frac{(M-1)r + K}{K}, \tag{12}$$

$$\tau_a^c(r, q) = \frac{Mr\mu(H_K - H_{K-q})}{K}, \tag{13}$$

$$\tau_a^d(r, q) = \sum_{p_1 = r-K+q}^{\min\{r,q\}} B_{p_1} \left(\sum_{p_2 = l_{p_1}}^{l_{\max}} \frac{B_{p_2}}{d_{p_1,M,p_2}^d} + \frac{B_{l_{p_1}-1}}{d_{p_1,M,l_{p_1}-1}^d} \right), \tag{14}$$

for any repetition order r and recovery order q in the set

$$\mathcal{R} \triangleq \big\{ (r, q) : r \in [K], q \in [K], \text{ and } (r-K+q)\mu \geq 1 \big\}, \tag{15}$$

where $H_K = \sum_{k=1}^{K} 1/k$, $H_0 = 0$, $B_{p_1} = \binom{q}{p_1}\binom{K-q}{r-p_1}/\binom{K}{r}$, $B_{p_2} = \binom{p_1}{p_2}\binom{K-p_1}{\rho_2-p_2}\rho_1/\binom{K}{\rho_2}$, $B_{l_{p_1}-1} = 1 - \sum_{p_2=l_{p_1}}^{l_{\max}} B_{p_2}$, and d_{p_1,M,p_2}^d is given by

$$d_{p_1,M,p_2}^d = \begin{cases} 1, & p_2 \geq M \\ \dfrac{\binom{p_1}{M-1}(M-1)}{\binom{p_1}{M-1}(M-1)+1}, & p_2 = M-1 \\ \max\left\{ d', \dfrac{p_2}{M} \right\}, & p_2 \leq M-2 \end{cases}, \tag{16}$$

with $d' \triangleq \max_{1 \leq t \leq p_2} \frac{p_1 - t + 1}{M + p_1 - 2t + 1}$;

$$\rho_2 = \inf\left\{ \rho : \binom{K}{\rho} - \binom{2K-r-q}{\rho} \geq \frac{1}{\rho_1}\binom{K}{\rho}, \rho_1\rho = K\mu, \rho_1 \in \left\{ 1, \frac{K\mu}{K\mu-1}, \frac{K\mu}{K\mu-2} \cdots, K\mu \right\} \right\}; \tag{17}$$

and

$$l_{p_1} = \inf\left\{ l : \sum_{p_2=l}^{l_{\max}} B_{p_2} m \leq m, l \in [l_{\min} : l_{\max}], l_{\max} = \min\{p_1, \rho_2\}, l_{\min} = \max\{\rho_2 - K + p_1, 1\} \right\}. \tag{18}$$

Therefore, for an NULT $\tau^u = \tau_a^u(r)$ given in (12) for some r, an inner bound $\mathcal{T}_{in}(\tau^u)$ on the compute-download latency region is given by the convex hull of the set $\big\{ \big(\tau_a^c(r, q), \tau_a^d(r, q)\big) : q \in [\lceil \frac{1}{\mu} \rceil + K - r : K] \big\}$.

Proof. The proof of Theorem 1 is given in Section IV. □

By Theorem 1, an achievable end-to-end execution time is given as follows.

Corollary 1. *An achievable end-to-end execution time for $(r, q) \in \mathcal{R}$ is given as $\tau_a(r, q) = \tau_a^u(r) + \delta_c \tau_a^c(r, q) + \delta_d \tau_a^d(r, q)$, where $\tau_a^u(r)$, $\tau_a^c(r, q)$, and $\tau_a^d(r, q)$ are given in (12), (13), and (14), respectively.*

We also have the following converse.

Theorem 2. *(Converse). For the same MEC network, the set of all admissible pairs (r, q) is included in the set $\mathcal{R}$ in (15). Furthermore, any feasible communication-computation latency*
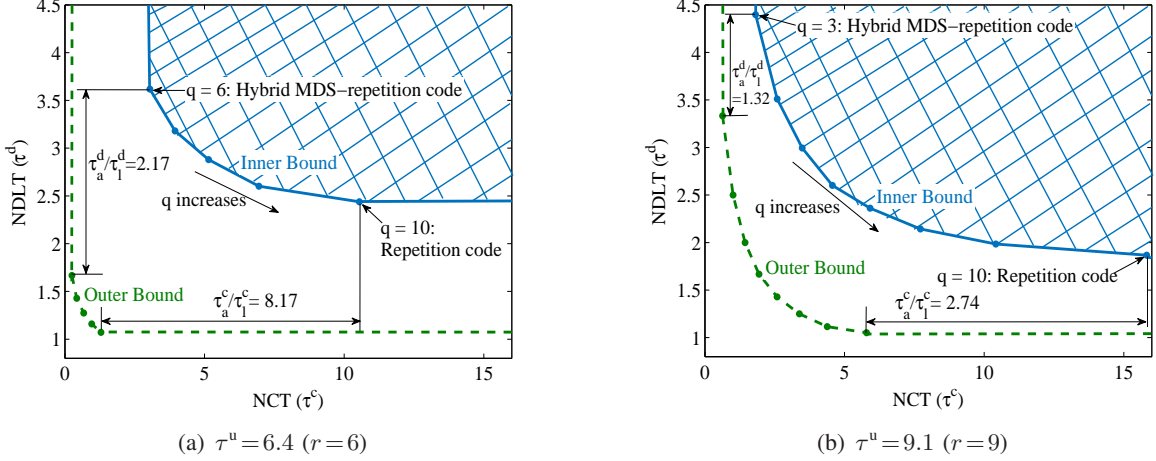
Fig. 3. Compute-download latency region bounds for $M=K=10$, $\mu=3/5$, $N=\binom{r}{10}$, and $m=\rho_2\binom{\rho_2}{10}$.

triplet $\big(\tau^{\mathrm{u}}(r), \tau^{\mathrm{c}}(r,q), \tau^{\mathrm{d}}(r,q)\big)$ *for pairs* (r,q) *in* $\mathcal{R}$ *is lower bounded as*

$$\tau^{\mathrm{u}}(r) \geq \tau_{\mathrm{a}}^{\mathrm{u}}(r), \tag{19}$$

$$\tau^{\mathrm{c}}(r,q) \geq \tau_l^{\mathrm{c}}(r,q) = \max_{t\in[q]} \frac{(H_K - H_{K-q+t-1})(r-K+t)^+ M\mu}{t}, \tag{20}$$

$$\tau^{\mathrm{d}}(r,q) \geq \tau_l^{\mathrm{d}}(r,q) = \max_{t\in\{1,\cdots,\min\{q,M\}\}} \frac{M - (M-t)(q-t)\frac{r}{K}\mu}{t}. \tag{21}$$

Therefore, for an NULT $\tau^{\mathrm{u}} = \tau_{\mathrm{a}}^{\mathrm{u}}(r)$ *in (12) for some* r, *an outer bound* $\mathcal{T}_{out}(\tau^{\mathrm{u}})$ *of the compute-download latency region is given by the convex hull of set* $\big\{\big(\tau_l^{\mathrm{c}}(r,q), \tau_l^{\mathrm{d}}(r,q)\big) : q \in \big[\lceil\frac{1}{\mu}\rceil + K - r : K\big]\big\}$.

Proof. The proof of Theorem 2 is available in Appendix. $\square$

Fig. 3 plots the derived inner and outer bounds on the compute-download latency region $\mathcal{T}^*(\tau^{\mathrm{u}})$ for the case with $M=K=10$ and two different values of τ^{u}. For instance, in Fig. 3-(a), for a small NULT $\tau^{\mathrm{u}} = 6.4$ at $r=6$, we have the achievable NCT-NDLT pairs $(\tau_{\mathrm{a}}^{\mathrm{c}}, \tau_{\mathrm{a}}^{\mathrm{d}}) = (3.04, 3.62)$ at $q=6$ and $(\tau_{\mathrm{a}}^{\mathrm{c}}, \tau_{\mathrm{a}}^{\mathrm{d}}) = (10.54, 2.44)$ at $q=10$; while, in Fig. 3-(b), for a large NULT $\tau^{\mathrm{u}} = 9.1$ at $r=9$, we have two smaller latency pairs $(\tau_{\mathrm{a}}^{\mathrm{c}}, \tau_{\mathrm{a}}^{\mathrm{d}}) = (2.59, 3.51)$ at $q=4$ and $(\tau_{\mathrm{a}}^{\mathrm{c}}, \tau_{\mathrm{a}}^{\mathrm{d}}) = (7.72, 2.14)$ at $q=8$. First, we observe that for both cases, *as* q *increases, the NDLT is reduced at the expense of an increasing NCT*: A larger q enables more opportunities for transmission cooperation at the ENs during output downloading, while increasing, on average, the time required for q ENs to complete their tasks. Furthermore, comparing Fig. 3-(a) with Fig. 3-(b), we also see that *allowing for a longer upload time* τ^{u} *increases the compute-download latency region*. This is because when more information is uploaded to ENs over a larger latency τ^{u}, on the one hand, users can wait for fewer ENs to finish their computing tasks, reducing the NCT; and, on the other hand, the increased duplication of outputs also increases opportunities for transmission cooperation to reduce the NDLT.

Corollary 2. *The minimum end-to-end execution time* $\tau(r,q)$ *for* $(r,q) \in \mathcal{R}$ *is lower bounded as* $\tau(r,q) \geq \tau_l(r,q) = \tau_{\mathrm{a}}^{\mathrm{u}}(r) + \delta_c \tau_l^{\mathrm{c}}(r,q) + \delta_d \tau_l^{\mathrm{d}}(r,q)$, *where* $\tau_{\mathrm{a}}^{\mathrm{u}}(r)$, $\tau_l^{\mathrm{c}}(r,q)$, *and* $\tau_l^{\mathrm{d}}(r,q)$ *are given in (12), (20), and (21), respectively.*

C. Optimality

The following lemma characterizes the optimality of the proposed scheme.

Lemma 1. *(Optimality). For any triplet $\big(\tau_a^u(r), \tau^c(r,q), \tau^d(r,q)\big)$ with NULT $\tau_a^u(r)$ in (12), it is not possible to reduce the achievable NULT τ^u while still guaranteeing the feasibility of a triplet $(\tau^u, \tau^c(r,q), \tau^d(r,q))$. Furthermore, for a sufficiently large NULT $\tau^u \geq \tau_a^u(K - n_1)$ and small recovery order $q \leq K(1-1/n_2)+1$, with integers $0 \leq n_1 < q/2$ and $n_2 \geq 1$, the multiplicative gap between the achievable NCT in (13) and its lower bound τ_l^c in (20) satisfies the inequality*

$$\tau_a^c/\tau_l^c \leq (1 + n_1)(1 + n_2). \tag{22}$$

Finally, for a sufficiently large NULT $\tau^u \geq \tau_a^u(K - n)$, with integer $n \geq 0$, the multiplicative gap between the achievable NDLT in (14) and its lower bound τ_l^d in (21) satisfies the inequality

$$\tau_a^d/\tau_l^d \leq 2(1+n\mu), \tag{23}$$

and hence, if $\tau^u \geq \tau_a^u(K)$, we have $\tau_a^d/\tau_l^d \leq 2$. In the special case $\mu = 1$, for a sufficiently large NULT $\tau^u \geq \tau_a^u(M+K-q)$, we have $\tau_a^d = \tau_l^d = 1$ that is optimal; for a smaller NULT $\tau_a^u(K-n) \leq \tau^u < \tau_a^u(M+K-q)$, with integer $q-M < n \leq q-1$, we have $\tau_a^d/\tau_l^d < n+1$.

Proof. The proof of Lemma 1 is given in Appendix. □

The multiplicative gaps in Fig. 3 are consistent with Lemma 1, since $\tau_a^c/\tau_l^c = 2.74 < 22$ at $(r,q) = (9,10)$ (i.e., $n_1 = 1$ and $n_2 = 10$) and $\tau_a^d/\tau_l^d = 1.32 < 3.2$ at $(r,q) = (9,3)$ (i.e., $n = 1$). Based on the inequality $\tau_a/\tau_l = (\tau_a^u + \delta_c\tau_a^c + \delta_d\tau_a^d)/(\tau_a^u + \delta_c\tau_l^c + \delta_d\tau_l^d) \leq \max\{\tau_a^c/\tau_l^c, \tau_a^d/\tau_l^d\}$, the order-optimality of the end-to-end execution time is obtained as below.

Corollary 3. *For a sufficiently large NULT $\tau^u \geq \tau_a^u(K - n_1)$ and small recovery order $q \leq K(1-1/n_2)+1$, with integers $0 \leq n_1 < q/2$ and $n_2 \geq 1$, the multiplicative gap between the achievable end-to-end execution time τ_a and its lower bound τ_l satisfies the inequality*

$$\tau_a/\tau_l \leq \max\{(1 + n_1)(1 + n_2), 2(1+n_1\mu)\}. \tag{24}$$

D. Special Cases

In the special case when $r = K$, hence ignoring limitations on the uplink transmission, the achievable NDLT (14) reduces to $\tau_a^d(K,q) = M \sum_{p_2=l_q}^{l_{max}} B_{p_2}/p_2 + B_{l_{p_1}-1}/(l_{p_1}-1)$ when using only ZF precoding in downlink, which is consistent with the normalized communication delay in [13, Eq. (13)]. Furthermore, when setting $q = K$, hence ignoring stragglers' effects, and $\mu = 1$, i.e., ignoring ENs' storage constraint, the achievable NDLT (14) reduces to $\tau_a^d = M/\min\{K, M\}$, which is optimal and recovers the communication load in [26, Remark 5], the NDT with cache-aided EN cooperation in [21, Eq. (25)], and the NDLT in [25, Eq. (50)].

IV. ACHIEVABLE SCHEME

In this section, we present the achievable scheme for any $\mu \in \{1/K, 2/K, \cdots, 1\}^3$ and any repetition and recovery order pair (r,q) in the feasible set $\mathcal{R}$ in (15). Note that each input is computed by at least $r-(K-q)$ non-stragglers, so set $\mathcal{R}$ ensures that any subset of $r-K+q$ ENs can store at least m rows of $\mathbf{A}$ to multiply each input. The entire scheme including task assignment, input uploading, edge computing and output downloading are detailed below.

---

[3]For general $u \in [\frac{1}{K}, 1]$ satisfying $K\mu = \beta\lceil K\mu \rceil + (1-\beta)\lfloor K\mu \rfloor$, we can use memory- and time-sharing methods to achieve the linear combinations of the latency triplets achieved at integers $\lceil K\mu \rceil$ and $\lfloor K\mu \rfloor$.
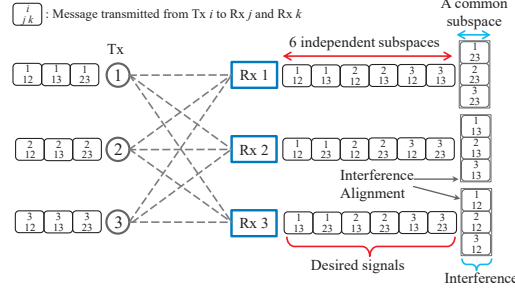
Fig. 4. Interference alignment on the 3-Tx 3-Rx X-multicast channel with size-2 multicast group.

1) Task Assignment: In this paper, we treat tasks from all users equally without considering user priority. So, without loss of generality, we consider the task assignment of $\{\mathbf{u}_{i,j}\}_{j=1}^{N}$ for user $i \in \mathcal{M}$. As discussed in Section III-A, for a repetition order r, we partition the N input vectors $\{\mathbf{u}_{i,j}\}_{j=1}^{N}$ of each user $i \in \mathcal{M}$ into $\binom{K}{r}$ equal-sized subsets, each denoted as $\mathcal{U}_{i,\mathcal{K}'}$ and assigned to all the r ENs in subset $\mathcal{K}' \subseteq \mathcal{K}$ for computation. Each EN k is thus assigned $M\binom{K-1}{r-1}N/\binom{K}{r} = MNr/K$ inputs corresponding to subsets $\{\mathcal{U}_{i,\mathcal{K}'} : i \in \mathcal{M}, \mathcal{K}' \subseteq \mathcal{K}, |\mathcal{K}'| = r, k \in \mathcal{K}'\}$. By Definition 1, the *repetition order* is calculated as $K(MNr/K)/MN = r$, which equals the cardinality $|\mathcal{K}'|$.

2) Input Uploading: Based on the task assignment $\{\mathcal{U}_{i,\mathcal{K}'}\}$, each user $i \in \mathcal{M}$ uploads the subset $\mathcal{U}_{i,\mathcal{K}'}$ of inputs to the subset $\mathcal{K}'$ of ENs via the uplink channel for $\mathcal{K}' \subseteq \mathcal{K}$ and $|\mathcal{K}'| = r$. In other words, each user communicates with all $\binom{K}{r}$ distinct subsets of ENs of cardinality r, and any subset of r ENs can form a receiver multicast group. Hence, the resulting uplink channel can be treated as an X-multicast channel with M transmitters, K receivers, and size-r multicast group, the same as that defined in [32] (see Fig. 4 for the case with $M = K = 3$ and $r = 2$). Enabled by asymptotic interference alignment with infinite symbol extensions, each group of M interfering signals from M transmitters can be aligned along the same direction at each receiver [32]. As a result, each receiver can successfully decode a total of $M\binom{K-1}{r-1}$ desired messages from M transmitters over the symbol-extended channel, with the other $M\binom{K-1}{r}$ undesired messages being aligned into $\binom{K-1}{r}$ common subspaces, each subspace containing M undesired messages from M transmitters. For instance, in Fig. 4, each receiver can decode the desired $3\binom{2}{1} = 6$ messages occupying independent subspaces with the undesired 3 signals sent by 3 transmitters being aligned into a common subspace. A per-receiver DoF of $6/7$ can be achieved asymptotically. In general, as proved in [22, Lemma 1] and [32, Theorem 2], the optimal per-receiver DoF of this channel is given by $d_r^{\mathrm{u}} = Mr/(Mr + K - r)$. The per-receiver rate of this channel in the high SNR regime can be approximated as $R_r^{\mathrm{u}} = d_r^{\mathrm{u}} \times \log P^{\mathrm{u}} + o(\log P^{\mathrm{u}})$, where only the first term is relevant in computing (7), so the uploading time can be approximately expressed as $T^{\mathrm{u}} = \frac{MNr}{K}nB/(d_r^{\mathrm{u}}\log P^{\mathrm{u}} + o(\log P^{\mathrm{u}}))$. Let $P^{\mathrm{u}} \to \infty$ and $B \to \infty$, by Definition 2, the NULT $\tau_{\mathrm{a}}^{\mathrm{u}}$ at repetition order r is given as below,

$$\tau_{\mathrm{a}}^{\mathrm{u}}(r) = \frac{Mr/K}{d_r^{\mathrm{u}}} = \frac{(M-1)r + K}{K}. \tag{25}$$

3) Edge Computing: After the input uploading phase is finished, each EN computes the products of the assigned input vectors and the stored coded matrix. Following Section III-A, a cascade of an MDS code with rate $1/\rho_1$ and a repetition code with rate $1/\rho_2$ is applied to encode matrix $\mathbf{A}$ into $\mathbf{A}_c$. Under the constraint of the total storage size $K\mu$, the code rates satisfy $\rho_1\rho_2 = K\mu$, $\rho_1 \in \{1, K\mu/(K\mu-1), K\mu/(K\mu-2), \cdots, K\mu\}$ and $\rho_2 \in [K\mu]$. Then, we split the coded matrix $\mathbf{A}_c$ into $\binom{K}{\rho_2}$ submatrices $\{\mathbf{A}_{c,\mathcal{K}''}\}$, each stored at a distinct subset $\mathcal{K}''$ of ρ_2 ENs. As shown in Fig. 2, when there are $K - q$ stragglers randomly occurring, any subset of $r - K + q$

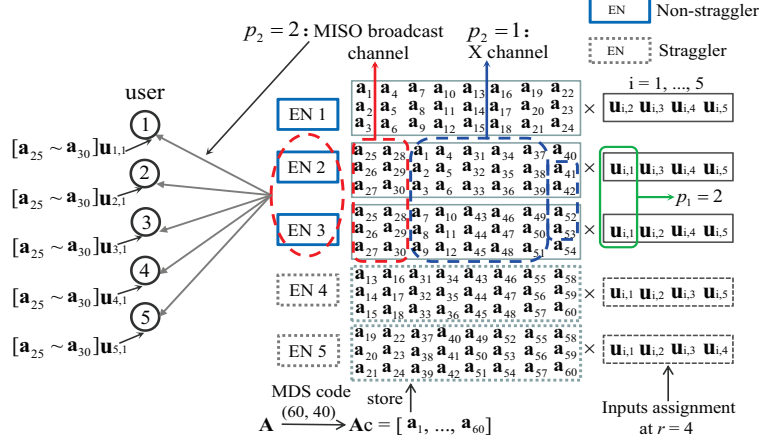Fig. 5. Illustration of downlink transmission for $K=M=5$, $\mu=3/5$, $m=40$, $N=5$, $q=3$, $r=4$, and $(\rho_1,\rho_2)=(3/2,2)$. The MISO broadcast channel and X-channel are formed to transmit outputs of $\{\mathbf{u}_{i,1}\}_{i=1}^5$ back to the users sequentially. This figure only shows the pattern of MISO broadcast channels for transmitting $\{\mathbf{a}_{25}\mathbf{u}_{i,1},\cdots,\mathbf{a}_{30}\mathbf{u}_{i,1}\}_{i=1}^5$. Outputs of $\{\mathbf{u}_{i,2}\}_{i=1}^5$, $\{\mathbf{u}_{i,3}\}_{i=1}^5$, $\cdots$, $\{\mathbf{u}_{i,5}\}_{i=1}^5$ are transmitted in a similar way.

non-straggling ENs must store at least m encoded rows to compute all outputs. This can be ensured by condition $\rho_1 m - \binom{K-(r-K+q)}{\rho_2}\rho_1 m/\binom{K}{\rho_2} \geq m$. Further, under this recovery condition, in order to create more data redundancy, the parameter $\rho_2 \in [K\mu]$ is maximized as

$$\rho_2 = \inf\left\{\rho: \binom{K}{\rho} - \binom{2K-r-q}{\rho} \geq \frac{1}{\rho_1}\binom{K}{\rho}, \rho_1\rho = K\mu, \rho_1 \in \left\{1, \frac{K\mu}{K\mu-1}, \frac{K\mu}{K\mu-2}\cdots, K\mu\right\}\right\};$$ (26)

As an example, in Fig. 5, for $K=M=5$, $m=40$, $\mu=3/5$, $q=3$, and $r=4$, by (26), we have $(\rho_1,\rho_2)=(3/2,2)$ such that $\mathbf{A}$ is encoded into 60 rows and then split into $\binom{5}{2}=10$ submatrices, each with 6 rows replicated at 2 ENs. By the given task input assignment $\{\mathcal{U}_{i,\mathcal{K}'}\}$, each EN k computes $MNr\mu m/K$ row-vector products. Let $\omega_{1:K} \leq \omega_{2:K} \leq \cdots \leq \omega_{K:K}$ denote the order statistics in a sample of size K from an exponential distribution with mean $1/\eta$ [33], by Definition 2, the NCT is given by

$$\tau_a^c(r,q) = \lim_{m\to\infty} \frac{\mathbb{E}\left[\frac{MNr\mu m}{K}\omega_{q:K}\right]}{Nm/\eta} = \frac{Mr\mu(H_K - H_{K-q})}{K},$$ (27)

which follows $\mathbb{E}[\omega_{q:K}] = (H_K - H_{K-q})/\eta$ [33, Eq. (4.6.6)].

4) Output Downloading: At the end of edge computing phase, the $K-q$ non-straggling ENs return the computed outputs back to users via the downlink channel. Following Section III-A, for each user, the number of input vectors computed by p_1 non-straggling ENs equals $\binom{K-q}{r-p_1}N/\binom{K}{r} = B_{p_1}N/\binom{q}{p_1}$, where $r-(K-q) \leq p_1 \leq \min\{r,q\}$. Furthermore, the number of encoded rows of $\mathbf{A}$ replicated at p_2 non-straggling ENs is $\binom{K-p_1}{\rho_2-p_2}\rho_1 m/\binom{K}{\rho_2} = B_{p_2}m/\binom{p_1}{p_2}$, where $\max\{\rho_2-K+p_1,1\} \leq p_2 \leq \min\{p_1,\rho_2\}$. Hence, among the p_1 non-straggling ENs, any subset of p_2 ENs computing the same $MB_{p_1}NB_{p_2}m/\left(\binom{q}{p_1}\binom{p_1}{p_2}\right)$ outputs can form a transmitter cooperation group, resulting in $\binom{p_1}{p_2}$ groups in total, and each EN cooperation group has outputs to send to all users. The resulting downlink is a cooperative X channel with p_1 transmitters, M receivers, and size-p_2 cooperation group, as defined in [22] (see Fig. 6 for the case with $p_1=M=3$ and $p_2=2$). As discussed in Sec. III-A, when $p_2 \geq M$, each subset of p_2 ENs can cooperatively transmit common outputs to M users via ZF precoding. In contrast, when $p_2 < M$, each subset of p_2 ENs partitions each common output into $\binom{M-1}{p_2-1}$ submessages, and first use ZF precoding to null the interference caused by each submessage at a distinct subset of p_2-1 undesired users. Then, by
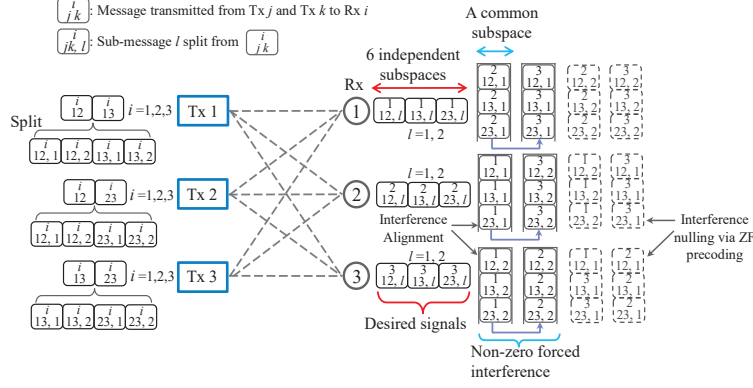
Fig. 6. Interference alignment and ZF precoding on the 3-Tx 3-Rx cooperative X channel with size-2 cooperation group.

cascading ZF precoding with asymptotic IA, the rest of interferences from each subset of $t-1$ ENs can be aligned into a distinct subspace at each user [22]. Particularly, when $p_2 = M-1$, each submessage only causes interference to one user, so all interfering signals at each user can be aligned into a common subspace. For example, in Fig. 6, each common message is split into 2 submessages with each being cancelled at a undesired receiver and causing interference only to another undesired receiver. Then, each receiver can decode the $2\binom{3}{2}=6$ desired submessages with the rest $2\binom{3}{2}=6$ interferences being aligned into a common subspace, which achieves a per-receiver DoF of $6/7$.

In general, by [22, Lemma 1], an achievable per-receiver DoF $d^{\mathrm{d}}_{p_1,M,p_2}$ of this downlink channel is given as (16), which is within a multiplicative gap of 2 to the optimal DoF. The per-receiver channel rate for high SNR regime can be approximated as $d^{\mathrm{d}}_{p_1,M,p_2} \times \log P^{\mathrm{d}} + o(\log P^{\mathrm{d}})$, where only the first term is relevant in computing (9). The traffic load for each user to download its desired outputs is $B_{p_1} N B_{p_2} m B / \binom{q}{p_1}$ bits, so the downloading time can be approximately given by $T^{\mathrm{d}} = \frac{B_{p_1} N B_{p_2} m B / \binom{q}{p_1}}{d^{\mathrm{d}}_{p_1,M,p_2} \log P^{\mathrm{d}} + o(\log P^{\mathrm{d}})}$. Let $P^{\mathrm{d}} \to \infty$ and $mB \to \infty$, by Definition 2, the NDLT for each user to download the outputs replicated at p_2 non-stragglers is given by

$$\tau^{\mathrm{d}}_{p1,p2} = \frac{B_{p_1} B_{p_2} / \binom{q}{p_1}}{d^{\mathrm{d}}_{p_1,M,p_2}}. \tag{28}$$

Due to the MDS coding, the total number of coded outputs available on the p_1 ENs may exceed the number m needed to recover the outputs of each input vector. Denote by $l_{p_1}-1$ the minimum degrees of replication of needed coded outputs on the p_1 ENs, l_{p_1} is determined by $l_{p_1} = \inf\{l : \sum_{p_2=l}^{\min\{p_1,p_2\}} B_{p_2} m \le m, l \ge \max\{\rho_2 - K + p_1, 1\}\}$, so the number of needed coded outputs replicated at $l_{p_1}-1$ ENs equals $M B_{p_1} N B_{l_{p_1}-1} m / \binom{q}{p_1}$, where $B_{l_{p_1}-1} = 1 - \sum_{p_2=l_{p_1}}^{l_{\max}} B_{p_2}$. Note that $B_{l_{p_1}-1} m / \binom{p_1}{l_{p_1}-1}$ can be seen as an integer for infinitely large m since $\left(B_{l_{p_1}-1} m \bmod \binom{p_1}{l_{p_1}-1}\right)/m < \binom{p_1}{l_{p_1}-1}/m \to 0$ as $m \to \infty$. So it enables any subset of $l_{p_1}-1$ ENs among p_1 ENs to cooperatively transmit $B_{p_1} N B_{l_{p_1}-1} m / \left(\binom{q}{p_1}\binom{p_1}{l_{p_1}-1}\right)$ common outputs to each user, the downlink channel is also a cooperative X channel with p_1 transmitters, M receivers, and size-$(l_{p_1}-1)$ cooperation group. Similar to (28), the NDLT for each user to download the outputs replicated at $l_{p_1}-1$ non-stragglers is given by

$$\tau^{\mathrm{d}}_{p1,l_{p_1}-1} = \frac{B_{p_1} B_{l_{p_1}-1} / \binom{q}{p_1}}{d^{\mathrm{d}}_{p_1,M,l_{p_1}-1}}. \tag{29}$$

Furthermore, by considering all the inputs computed by p_1 ENs, with p_1 from $r-(K-q)$ to

$\min\{r, q\}$, and all the outputs replicated at p_2 ENs, with p_2 from l_{p_1-1} to $\min\{p_1, \rho_2\}$, and by summing all download time given in (28) and (29), the NDLT $\tau_a^d(r, q)$ is obtained as below,

$$\tau_a^d(r, q) = \sum_{p_1=r-K+q}^{\min\{r,q\}} B_{p_1} \left(\sum_{p_2=l_{p_1}}^{l_{\max}} \frac{B_{p_2}}{d_{p_1,M,p_2}^d} + \frac{B_{l_{p_1-1}}}{d_{p_1,M,l_{p_1}-1}^d} \right) \tag{30}$$

We now illustrate the output downloading latency by the example in Fig. 5. First, for inputs $\{\mathbf{u}_{i,1}\}_{i=1}^5$ computed by $p_1=2$ ENs, there are 30 outputs $\{\mathbf{a}_{25}\mathbf{u}_{i,1}, \dots, \mathbf{a}_{30}\mathbf{u}_{i,1}\}_{i=1}^5$ replicated at $p_2=2$ ENs. These 30 outputs can be cooperatively transmitted back to the users via ZF precoding, resulting in a 2-transmitter 5-receiver MISO broadcast channel that is a special case of cooperative X channels under full transmitter cooperation. As a result, an NDLT of $3/40$ is achieved. After this round of transmission, users still need $34 \times 5 = 170$ outputs inside the blue dashed rectangle in Fig. 5, which can be transmitted by the 2 ENs via interference alignment. The downlink is a 2-transmitter 5-receiver X channel that is a special case of cooperative X channels with size-1 cooperation group, yielding the NDLT of $51/100$. Thus, the NDLT for outputs of $\{\mathbf{u}_{i,1}\}_{i=1}^5$ is $3/40 + 51/100 = 117/200$. Then, the input vectors $\{\mathbf{u}_{i,2}\}_{i=1}^5, \{\mathbf{u}_{i,3}\}_{i=1}^5$ are also computed by $p_1=2$ ENs, their outputs can be transmitted in a similar way, which achieves an NDLT of $(117/200) \times 2 = 117/100$. Likewise, for the inputs $\{\mathbf{u}_{i,4}\}_{i=1}^5, \{\mathbf{u}_{i,5}\}_{i=1}^5$ computed by $p_1=3$ ENs, the 3-transmitter 5-receiver cooperative X-channel with size-2 cooperation group, and 3-transmitter 5-receiver X-channel are formed to transmit the total 400 outputs, yielding an NDLT of $(21/100 + 77/300) \times 2 = 14/15$. Thus, in this example, the total NDLT at $(r, q) = (4, 3)$ is $14/15 + (117/200) \times 3 = 1613/600$.

5) Inner Bound of Compute-Download Latency Region: For an NULT $\tau^u = \tau_a^u(r)$ given in (25) for some $r \in [K]$, by the region $\mathcal{R}$ given in (15), the feasible recovery order q satisfies $\lceil 1/\mu \rceil + K - r \leq q \leq K$. By Remark 1, for any two integer-valued q_1 and q_2 in $\left[\lceil 1/\mu \rceil + K - r : K\right]$, any convex combination of achievable pairs $(\tau_a^c(r, q_1), \tau_a^d(r, q_1))$ and $(\tau_a^c(r, q_2), \tau_a^d(r, q_2))$ can also be achieved. So an inner bound $\mathcal{T}_{in}(\tau^u)$ of the compute-download latency region is given as the convex hull of set $\left\{ \left(\tau_a^c(r, q), \tau_a^d(r, q) \right) : q \in \left[\lceil \frac{1}{\mu} \rceil + K - r : K\right] \right\}$.

V. NUMERICAL EXAMPLES AND DISCUSSION

In this section, we first evaluate the system performance in terms of the asymptotic end-to-end execution time τ in Eq. (11). Then, we use numerical examples to show the average uploading, computing, downloading, and end-to-end execution times in the non-asymptotic regime.

A. Asymptotic Results

The analysis in the previous section has shown that, by choosing the repetition order r and the recovery order q, one can obtain different triplets of the upload latency τ^u, computation latency τ^d, and download latency τ^c. As a result, parameters (r, q) can be optimized to minimize the end-to-end execution time, yielding the minimum end-to-end execution time $\tau^* = \min_{(r,q) \in \mathcal{R}} \tau(r, q)$. To illustrate the minimum end-to-end execution time τ^*, we consider a MEC network with $M = 8$ users and $K = 10$ servers. Each EN has a fractional storage size of $\mu = 3/5$. We compare the proposed scheme with the following baseline strategies, for which parameters r and q are also optimized: a) MDS coding: Only an MDS code is applied to encode $\mathbf{A}$, i.e., we have the special case $(1/\rho_1, 1/\rho_2) = (1/K\mu, 1)$; b) Repetition coding: Only the repetition code is applied to encode $\mathbf{A}$, i.e., we have $(1/\rho_1, 1/\rho_2) = (1, 1/K\mu)$; c) Achievable schemes in [13]: Each EN has the inputs of all users and transmits the outputs back by using one-shot linear precoding. A cascade of MDS and repetition codes is applied to encode $\mathbf{A}$ as in the proposed scheme.
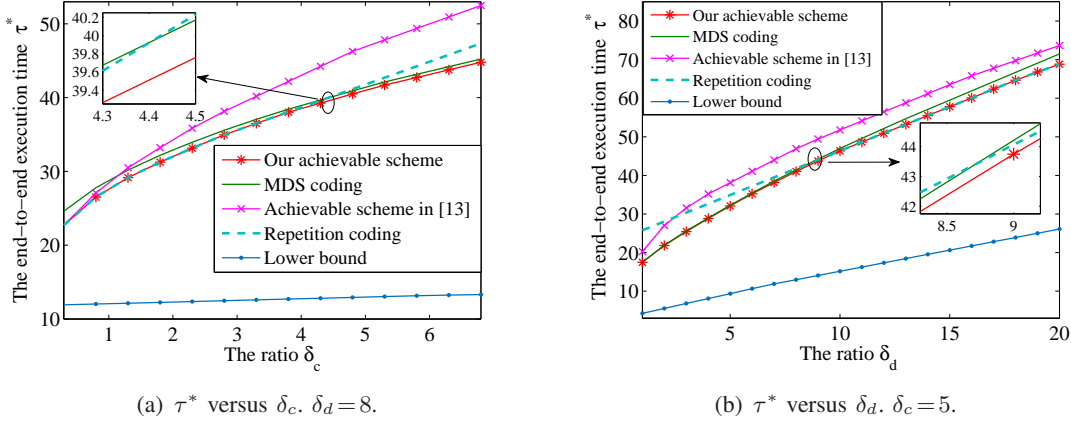
(a) τ^* versus δ_c. $\delta_d = 8$.

(b) τ^* versus δ_d. $\delta_c = 5$.

Fig. 7. The impacts of the ratios δ_c and δ_d on the end-to-end execution time τ^*.

Fig. 7 shows the impacts of the ratio δ_c between computation and upload reference latencies and the ratio δ_d between download and upload reference latencies on the end-to-end execution time τ^*. By providing more flexible choices for task uploading, the proposed scheme increases the achievable region in terms of the latencies for task uploading, computing, and output downloading, leading to a reduction in the end-to-end execution time. In Fig. 7-(a), it is also observed that, when the computing speed is slow, i.e., δ_c is large, it is suboptimal to upload the input data of each user to all ENs, and, as a result, the proposed scheme obtains gains with respect to [13] that increases with δ_c. In contrast, when δ_c is small, the proposed scheme reduces to that of [13], since, for fast server computing speeds, the optimal task assignment policy prefers to replicate tasks at all ENs.

Fig. 7 also demonstrates the advantages of using both MDS and repetition coding. In Fig. 7-(b), when δ_d is sufficiently small, i.e., $\delta_d \leq 8.5$, the proposed cascaded MDS-repetition coding scheme approaches MDS coding. This is because in this regime, repetition coding only brings limited transmission cooperation gain in the downlink, but it requires large upload and computation latencies for output recovery. Therefore, when uplink transmission and edge computing are the major bottlenecks of the offloading process, MDS coding can reduce the input uploading constraint required for output recovery and mitigate random straggling effects at computing phases. In contrast, when δ_d is larger, downlink latency becomes the bottleneck, and repetition coding is preferable since it can fully exploit transmission cooperation gains in data downloading to reduce the download time.

B. Non-asymptotic Results

We consider a 2-user 3-server MEC network, and set the network parameters $N = 3$, $\mu = 2/3$, $n = 100$, $m = 900$ rows, $B = 8$ bits, $1/\eta = 10^{-4}$ seconds. The uplink and downlink channel bandwidths are $B_u = B_d = 100$ KHz. We consider the normalized Rayleigh channel fading and normalized noise power. By set $\mathcal{R}$ in (15), we have feasible pairs $(r, q) \in \{(2, 3), (3, 3), (3, 2)\}$. Interference alignment on 2×3 or 3×2 networks only needs finite symbol extensions over these channels. The simulations are averaged over 50000 independent channel realizations.

Table I shows the actual average input uploading time T^u, edge computing time T^c, output downloading time T^d, and end-to-end execution time T, at different transmission powers. It is seen that the optimal policy is to upload user inputs to all 3 ENs and then wait for the fastest 2 ENs to finish their tasks, which significantly reduces the total time T. For example, when $P^u = P^d = 20$ dB, the total time at $(r, q) = (2, 3)$ is decreased by 23% compared to the

Table I. Actual average uploading, computing, and downloading times

P^{u} (or P^{d})	10 dB			20 dB			30 dB		
(r,q)	$(2,3)$	$(3,3)$	$(3,2)$	$(2,3)$	$(3,3)$	$(3,2)$	$(2,3)$	$(3,3)$	$(3,2)$
T^{u} (sec.)	0.112	0.158	0.158	0.027	0.036	0.036	0.011	0.014	0.014
T^{c} (sec.)	0.44	0.66	0.3	0.44	0.66	0.3	0.44	0.66	0.3
T^{d} (sec.)	0.315	0.106	0.313	0.105	0.048	0.105	0.053	0.029	0.053
Total time T (sec.)	0.867	0.924	0.771	0.572	0.744	0.441	0.504	0.703	0.367

optimal time at $(r,q)=(3,2)$. In this policy, the higher repetition order enables the transmission cooperation to reduce downloading times, and the lower recovery order mitigates the effect of 1 straggling node and thus reduces computing times. Comparing the times at $(r,q)=(2,3)$ and $(r,q)=(3,3)$, we also see that replicating inputs at all 3 ENs may cause more total times since the gain on reducing downloading times is limited compared to the increased computing time.

VI. CONCLUSIONS

This paper studies the communication-computation tradeoff for distributed matrix multiplication in multi-user multi-server MEC networks with straggling ENs. We propose a new task offloading policy that leverages cascaded coded computing and cooperative transmission to alleviate the impact of straggling ENs and speed up the communication phase. We derive achievable upload-compute-download latency triplets and the end-to-end execution time, as well as the lower bounds. We prove that the obtained upload latency is optimal for fixed computation and download latencies, and that the computation latency and download latency are within constant multiplicative gaps to their respective lower bounds for a sufficiently large upload latency. Our results reveal that for a fixed upload latency, the download latency can be traded for computation latency; and that increasing the upload latency can reduce both the computation latency and download latency. Through numerical results, we show that the proposed policy is able to obtain a more flexible trade-off among upload, computation, and download latencies than baseline schemes, and that this leads to a significant reduction in the end-to-end execution time.

APPENDIX: PROOFS OF CONVERSE

In this appendix, we prove Theorem 2 and Lemma 1. Note that in each subsection, we first derive the lower bound in Theorem 2, and then prove the multiplicative gap in Lemma 1.

For a repetition-recovery order pair (r,q), as discussed, each input is computed by at least $r-(K-q)$ non-stragglers. The condition $(r-K+q)\mu \geq 1$ must be satisfied such that any subset of $r-K+q$ non-stragglers are able to provide sufficient information to compute the outputs of all users. This proves that no pair (r,q) is feasible outside the feasible set $\mathcal{R}$ in (15). Then, we consider an arbitrary user input assignment policy $\{\mathcal{U}_{i,\mathcal{K}'} : i\in\mathcal{M}, \mathcal{K}'\subseteq\mathcal{K}, |\mathcal{K}'|=r\}$ with $(r,q)\in \mathcal{R}$. The input vectors from user i assigned to EN k are denoted as set $\mathcal{I}_{i,k} \triangleq \{\mathcal{U}_{i,\mathcal{K}'}\}_{\mathcal{K}'\subset\mathcal{K}:k\in\mathcal{K}'}$ for $i\in\mathcal{M}$ and $k\in\mathcal{K}$. The size of $\mathcal{I}_{i,k}$ is denoted as $\gamma_{i,k}NnB$ bits, where the ratio $\gamma_{i,k}$ satisfies

$$\sum_{k\in\mathcal{K}} \gamma_{i,k} = r, \ i \in \mathcal{M} \tag{31}$$

$$0 \leq \gamma_{i,k} \leq 1, \ i \in \mathcal{M}, \text{ and } k \in \mathcal{K}. \tag{32}$$

In the following, we first derive the lower bounds on the NULT, NCT, and NDLT for a particular task assignment policy $\{\mathcal{U}_{i,\mathcal{K}'}\}$ with repetition-recovery order pair $(r,q)\in\mathcal{R}$. Then, by considering all possible task assignment policies and the effect of random stragglers, we obtain the minimum lower bounds for the NULT τ_l^{u}, NCT τ_l^{c}, and NDLT τ_l^{d}. For a fixed NULT at

$r \in \left[\lceil \frac{1}{\mu} \rceil, K\right]$, by convexity of the compute-download latency region, an outer bound of this region is given by the convex hull of all pairs $\{(\tau_l^{\mathrm{c}}, \tau_l^{\mathrm{d}})\}$, as described in Theorem 2.

A. Lower Bound and Optimality of NULT

1) Lower bound: For a particular task assignment policy $\{\mathcal{U}_{i,\mathcal{K}'}\}$, we use genie-aided arguments to derive a lower bound on the NULT. Specifically, for any EN k and user i_o, consider the following three disjoint subsets of task input vectors (or messages):

$$\mathcal{W}_r = \{\mathcal{U}_{i,\mathcal{K}'} : i \in \mathcal{M}, k \in \mathcal{K}'\}, \tag{33}$$

$$\mathcal{W}_t = \{\mathcal{U}_{i,\mathcal{K}'} : i = i_o, k \notin \mathcal{K}'\}, \tag{34}$$

$$\overline{\mathcal{W}} = \{\mathcal{U}_{i,\mathcal{K}'} : i \neq i_o \text{ and } k \notin \mathcal{K}'\}. \tag{35}$$

The set $\mathcal{W}_r$ indicates the input messages from all users assigned to EN k or all input messages that EN k needs to decode, which satisfies $|\mathcal{W}_r| = \sum_{i \in \mathcal{M}} \gamma_{i,k} NnB$. The set $\mathcal{W}_t$ indicates the input messages from user i_o assigned to all ENs in $\mathcal{K}$ excluding EN k, which satisfies $|\mathcal{W}_t| = (1 - \gamma_{i_o,k}) NnB$. The last set $\overline{\mathcal{W}}$ indicates all input messages from users in $\mathcal{M}$ excluding user i assigned to ENs in $\mathcal{K}$ excluding EN k.

Let a genie provide the messages $\overline{\mathcal{W}}$ to all ENs, and additionally provide messages $\mathcal{W}_r$ to ENs in $\mathcal{M}/\{k\}$. The received signal of EN j can be represented as

$$\mathbf{y}_j = \sum_{i=1, i \neq i_o}^{M} \mathbf{H}_{ji}^{\mathrm{u}} \mathbf{X}_i + \mathbf{H}_{ji_o}^{\mathrm{u}} \mathbf{X}_{i_o} + \mathbf{Z}_j^{\mathrm{u}}, \tag{36}$$

where the diagonal matrices $\mathbf{H}_{ji}^{\mathrm{u}}$, $\mathbf{X}_i$, and $\mathbf{Z}_j^{\mathrm{u}}$ denotes the channel coefficients from user i to EN j, signal transmitted by user i, and noise received at EN j, respectively, over the block length T^{u}. The ENs in $\mathcal{M}/\{k\}$ have messages $\overline{\mathcal{W}} \bigcup \mathcal{W}_r$, which include the input messages that EN k should decode and input messages transmitted by all users excluding user i_o. By this genie-aided information, each EN $j \in \mathcal{M}/\{k\}$ can construct the transmitted symbols $\{\mathbf{X}_i : i \neq i_o\}$ and subtract them from the received signal. So we can rewrite the signal received at EN $j \neq k$ as

$$\bar{\mathbf{y}}_j = \mathbf{y}_j - \sum_{i \in \mathcal{M}/\{i_o\}} \mathbf{H}_{ji}^{\mathrm{u}} \mathbf{X}_i = \mathbf{H}_{ji_o}^{\mathrm{u}} \mathbf{X}_{i_o} + \mathbf{Z}_j^{\mathrm{u}}. \tag{37}$$

Each EN $j \in \mathcal{M}/\{k\}$ needs to decode the input messages in subset $\mathcal{W}_t$ assigned to it, denoted as $\mathcal{W}_t^j$. By Fano's inequality and (37), we have

$$H(\mathcal{W}_t^j | \mathbf{y}_j, \overline{\mathcal{W}}, \mathcal{W}_r) \leq T^{\mathrm{u}} \epsilon, \quad j \in \mathcal{M}/\{i\}. \tag{38}$$

Since EN k can decode input messages $\mathcal{W}_r$ assigned to it, by Fano's inequality, we also obtain

$$H(\mathcal{W}_r | \hat{\mathbf{y}}_k, \overline{\mathcal{W}}) \leq T^{\mathrm{u}} \epsilon. \tag{39}$$

Then, EN k can construct the transmitted symbols $\{\mathbf{X}_i : i \neq i_o\}$ based on genie-aided messages $\overline{\mathcal{W}}$ and its decoded messages $\mathcal{W}_r$, and subtract them from its received signal, obtaining

$$\bar{\mathbf{y}}_k = \mathbf{y}_k - \sum_{i \in \mathcal{M}/\{i_o\}} \mathbf{H}_{ki}^{\mathrm{u}} \mathbf{X}_i = \mathbf{H}_{ki_o}^{\mathrm{u}} \mathbf{X}_{i_o} + \mathbf{Z}_k^{\mathrm{u}}. \tag{40}$$

Reducing the noise in the constructed signal $\bar{\mathbf{y}}_k$ and multiplying it by $\mathbf{H}_{ji_o}^{\mathrm{u}} \left(\mathbf{H}_{ki_o}^{\mathrm{u}}\right)^{-1}$, we obtain

$$\bar{\mathbf{y}}_k^j = \mathbf{H}_{ji_o}^{\mathrm{u}} \left(\mathbf{H}_{ki_o}^{\mathrm{u}}\right)^{-1} \bar{\mathbf{y}}_k = \mathbf{H}_{ji_o}^{\mathrm{u}} \mathbf{X}_{i_o} + \hat{\mathbf{Z}}_j^{\mathrm{u}}, \tag{41}$$

where $\hat{\mathbf{Z}}_j^{\mathrm{u}}$ is the reduced noise. By (37), we see that $\bar{\mathbf{y}}_k^j$ is a degraded version of $\bar{\mathbf{y}}_j$ for EN $j \in \mathcal{M}/\{i\}$. Hence, for the messages that ENs in $\mathcal{M}/\{i\}$ can decode, EN k must also be able to decode them, and we have

$$H(\mathcal{W}_t^j | \hat{\mathbf{y}}_k, \overline{\mathcal{W}}, \mathcal{W}_r) \leq H(\mathcal{W}_t^j | \mathbf{y}_j, \overline{\mathcal{W}}, \mathcal{W}_r) \leq T^{\mathrm{u}} \epsilon, j \in \mathcal{M}/\{i\}. \tag{42}$$

Using genie-aided information, receiver cooperation, and noise reducing as discussed above can only improve channel capacity. Thus, we obtain the following chain of inequalities,

$$|\mathcal{W}_r| + |\mathcal{W}_t| = H(\mathcal{W}_r, \mathcal{W}_t)$$

$$\overset{(a)}{=} H(\mathcal{W}_r, \mathcal{W}_t | \overline{\mathcal{W}})$$

$$\overset{(b)}{=} I(\mathcal{W}_r, \mathcal{W}_t; \hat{\mathbf{y}}_k | \overline{\mathcal{W}}) + H(\mathcal{W}_r, \mathcal{W}_t | \hat{\mathbf{y}}_k, \overline{\mathcal{W}})$$

$$\overset{(c)}{=} I(\mathcal{W}_r, \mathcal{W}_t; \hat{\mathbf{y}}_k | \overline{\mathcal{W}}) + H(\mathcal{W}_r | \hat{\mathbf{y}}_k, \overline{\mathcal{W}}) + H(\mathcal{W}_t | \hat{\mathbf{y}}_k, \mathcal{W}_r, \overline{\mathcal{W}})$$

$$\leq I(\mathcal{W}_r, \mathcal{W}_t; \hat{\mathbf{y}}_k | \overline{\mathcal{W}}) + H(\mathcal{W}_r | \hat{\mathbf{y}}_k, \overline{\mathcal{W}}) + \sum_{j \in \mathcal{M}/\{k\}} H(\mathcal{W}_t^j | \hat{\mathbf{y}}_k, \mathcal{W}_r, \overline{\mathcal{W}})$$

$$\overset{(d)}{\leq} I(\mathcal{W}_r, \mathcal{W}_t; \hat{\mathbf{y}}_k | \overline{\mathcal{W}}) + T^{\mathrm{u}} \epsilon + \sum_{j \in \mathcal{M}/\{k\}} T^{\mathrm{u}} \epsilon$$

$$\overset{(e)}{\leq} I(\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_{a_i}, \mathbf{x}_{i_o}; \hat{\mathbf{y}}_i | \overline{\mathcal{W}}) + M T^{\mathrm{u}} \epsilon$$

$$\overset{(f)}{\leq} T^{\mathrm{u}} \log P^{\mathrm{u}} + M T^{\mathrm{u}} \epsilon, \tag{43}$$

where (a) is due to the independence of messages; (b) and (c) are based on the chain rule; (d) follows Fano's inequalities (39) and (42); (e) uses the data processing inequality; and (f) follows the DoF bound of MAC channel. Dividing (43) by $NnB/\log P^{\mathrm{u}}$, and let $P^{\mathrm{u}} \to \infty$ and $\epsilon \to 0$ as $B \to \infty$, we have

$$\tau^{\mathrm{u}} \geq \frac{|\mathcal{W}_r| + |\mathcal{W}_t|}{NnB} = \sum_{i \in \mathcal{M}} \gamma_{i,k} + 1 - \gamma_{i_o,k} = \sum_{i \in \mathcal{M}/\{i_o\}} \gamma_{i,k} + 1. \tag{44}$$

Hence, the NULT for a particular task assignment $\boldsymbol{\gamma} \triangleq [\gamma_{i,k}]_{i \in \mathcal{M}, k \in \mathcal{K}}$ satisfies $\tau^{\mathrm{u}} \geq \sum_{i \in \mathcal{M}/\{i_o\}} \gamma_{i,k} + 1$ for $k \in \mathcal{K}, i_o \in \mathcal{M}$, i.e., the minimum NULT for task assignment policy $\boldsymbol{\gamma}$ is lower bounded by

$$\tau^{\mathrm{u}^*}(r, \boldsymbol{\gamma}) \geq \max_{k \in \mathcal{K}, i_o \in \mathcal{M}} \sum_{i \in \mathcal{M}/\{i_o\}} \gamma_{i,k} + 1. \tag{45}$$

Further, the minimum NULT over all feasible task assignment is given as $\tau^{\mathrm{u}^*}(r) = \min_{\boldsymbol{\gamma}} \tau^{\mathrm{u}^*}(r, \boldsymbol{\gamma})$, i.e., it can be lower bounded by the optimal solution of the optimization problem

$$\mathcal{P}_1 : \quad \min_{\boldsymbol{\gamma}} \quad \max_{k \in \mathcal{K}, i_o \in \mathcal{M}} \sum_{i \in \mathcal{M}/\{i_o\}} \gamma_{i,k} + 1$$

$$s.t. \quad (31), (32).$$

Note that (31) and (32) are the task assignment constraints for recovery order r. By defining a new variable $\lambda_{k, \bar{i}_o} = \sum_{i \in \mathcal{M}/\{i_o\}} \gamma_{i,k}$, Problem $\mathcal{P}_1$ can be transformed into

$$\mathcal{P}_2 : \quad \min_{\boldsymbol{\lambda}} \quad \max_{k \in \mathcal{K}, i_o \in \mathcal{M}} \lambda_{k, \bar{i}_o} + 1$$

$$s.t. \quad \sum_{k \in \mathcal{K}} \lambda_{k, \bar{i}_o} = r(M - 1), \ i_o \in \mathcal{M}, \tag{46}$$

$$0 \leq \lambda_{k, \bar{i}_o} \leq M - 1, \ k \in \mathcal{K}, \ i_o \in \mathcal{M}. \tag{47}$$

Lemma 2. *The unique optimal solution to $\mathcal{P}_2$ is given by $\lambda_{k, \bar{i}_o}^* = r(M-1)/K$, $k \in \mathcal{K}$, $i_o \in \mathcal{M}$.*

Proof: By contradiction, assuming that there exists an optimal solution $\{\lambda_{k, \bar{i}_o}'\}$ to $\mathcal{P}_2$ which does not satisfy $\lambda_{k, \bar{i}_o}' = r(M-1)/K$ for $k \in \mathcal{K}$ and $i_o \in \mathcal{M}$. By (46), there must exist index $j \in \mathcal{K}$

such that $\lambda'_{j,\bar{i}_o} > r(M-1)/K$ for $i_o \in \mathcal{M}$; otherwise, we have $\sum_{k \in \mathcal{K}} \lambda'_{k,\bar{i}_o} < r(M-1)$ for $i_o \in \mathcal{M}$. The optimal objective satisfies $\max_{k \in \mathcal{K}, i_o \in \mathcal{M}} \lambda'_{k,\bar{i}_o} + 1 \geq \lambda'_{j,\bar{i}_o} + 1 > r(M-1)/K+1$, and $r(M-1)/K+1$ is the objective value at $\lambda_{k,\bar{i}_o} = r(M-1)/K$, $k \in \mathcal{K}$, $i_o \in \mathcal{M}$. So the initial assumption does not hold. The optimal solution to $\mathcal{P}_2$ is $\lambda^*_{k,\bar{i}_o} = r(M-1)/K$ for $k \in \mathcal{K}$ and $i_o \in \mathcal{M}$, which is unique.

In turn, we use $\{\lambda^*_{k,\bar{i}_o}\}$ in Lemma 2 to construct a feasible solution to $\mathcal{P}_1$ by letting $\gamma^*_{i,k} = \lambda^*_{k,\bar{i}_o}/(M-1)$ for $i \in \mathcal{M}$ and $k \in \mathcal{K}$, and hence obtain the optimal solution to $\mathcal{P}_1$ as $\gamma^*_{i,k} = r/K$. Therefore, at repetition order r, the minimum NULT $\tau^{\mathrm{u}^*}(r)$ is lower bounded by

$$\tau^{\mathrm{u}^*}(r) \geq \tau^{\mathrm{u}}_l(r) = \frac{r(M-1)+K}{K}. \tag{48}$$

The lower bound of NULT in Theorem 2 is thus proved.

2) Optimality: Since (48) is the same as achievable bound (12), the NULT in (12) is optimal for any given r, or more sufficiently, for any fixed $\left(\tau^{\mathrm{c}}(r,q), \tau^{\mathrm{d}}(r,q)\right)$, as stated in Lemma 1.

B. Lower Bound and Multiplicative Gap Analysis of NCT

1) Lower bound: Let $\{X_k\}_{q:K}$ denote the q-th smallest value of K variables $\{X_k\}_{k=1}^K$ and $q:K$ denote the index of q-th smallest variable. For a particular task assignment policy $\{\mathcal{U}_{i,\mathcal{K}'}\}$ with repetition order r and recovery order q and satisfying (31) and (32), the computation time when the q-th fastest EN finishes its assigned tasks is lower bounded by

$$T^{\mathrm{c}}_{q:K} = \left\{\mu m \sum_{i \in \mathcal{M}} \gamma_{i,k} N \omega_k\right\}_{q:K}$$

$$\overset{(g)}{\geq} \max_{t \in [q]} \left\{\mu m \left\{\sum_{i \in \mathcal{M}} \gamma_{i,k} N\right\}_{t:K} \cdot \omega_{q-t+1:K}\right\}, \tag{49}$$

where (g) follows the fact that for 2 sequences $\{x_k\}_{k=1}^K$ and $\{y_k\}_{k=1}^K$, given $\{x_k\}_{t:K}\{y_k\}_{q-t+1:K}$ for $t \in [q]$, there are at most $q-1$ product values among $\{x_k y_k\}_{k=1}^K$ less than $\{x_k\}_{t:K}\{y_k\}_{q-t+1:K}$ for $t \in [q]$. So the q-th smallest product satisfies $\{x_k y_k\}_{q:K} \geq \{x_k\}_{t:K}\{y_k\}_{q-t+1:K}$ for $t \in [q]$. Taking the expectation on $T^{\mathrm{c}}_{q:K}$, we have

$$\mathbb{E}\left[T^{\mathrm{c}}_{q:K}\right] \geq \mathbb{E}\left[\max_{t \in [q]} \left\{\mu m \left\{\sum_{i \in \mathcal{M}} \gamma_{i,k} N\right\}_{t:K} \cdot \omega_{q-t+1:K}\right\}\right]$$

$$\overset{(h)}{\geq} \max_{t \in [q]} \left\{\mu m \left\{\sum_{i \in \mathcal{M}} \gamma_{i,k} N\right\}_{t:K} \cdot \mathbb{E}\left[\omega_{q-t+1:K}\right]\right\}$$

$$\overset{(i)}{=} \max_{t \in [q]} \frac{(H_K - H_{K-q+t-1})\mu m}{\eta} \left\{\sum_{i \in \mathcal{M}} \gamma_{i,k} N\right\}_{t:K}, \tag{50}$$

where (h) follows $\mathbb{E}\left[\max_t x_t\right] \geq \max_t \mathbb{E}\left[x_t\right]$, (i) uses the $(q-t+1)$-th order statistic of K i.i.d exponential random variables. The second term denotes the t-th smallest value among K EN workload sizes. By (31) and (32), for $\forall i \in \mathcal{M}$, we let $\gamma_{i,k} = 1$, $k = t+1:K, t+2:K, \cdots, K:K$, where $k = t:K$ denotes the index of the t-th smallest value in $\{\gamma_{i,k}\}_{k \in [K]}$. So the sum of the t smallest values ($k = 1:K, \cdots, t:K$) is lower bounded by $(r-K+t)^+ NM$. Since the second term also represents the largest value among those t smallest EN workload sizes, so this term can be further lower bounded by the average value $(r-K+t)^+ NM/t$. So the average time for the q fastest ENs to finish their tasks is lower bounded by $T^{\mathrm{c}}(r,q) \geq \max_{t \in [q]} \left((H_K - H_{K-q+t-1})\mu m/\eta\right)\left((r-\right.$
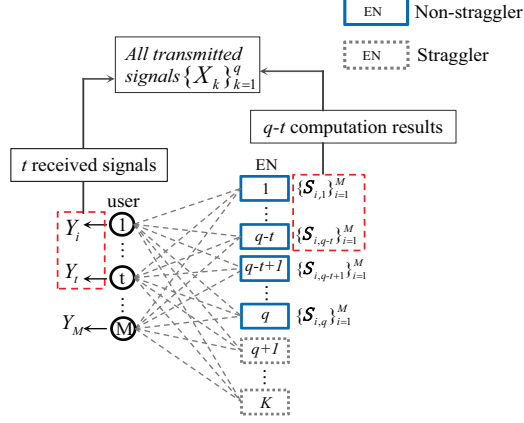
Fig. 8. Illustration of the converse proof for NDLT.

$K+t)^+NM/t$). Normalizing it by Nm/η, the lower bound of the minimum NCT is given by

$$\tau^{c*}(r,q) \geq \tau_l^c(r,q) = \max_{t\in[q]} \frac{(H_K - H_{K-q+t-1})(r-K+t)^+M\mu}{t}. \tag{51}$$

2) Multiplicative gap: The multiplicative gap between the achievable NCT in Theorem 1 and the lower bound (51) satisfies

$$\frac{\tau_a^c(r,q)}{\tau_l^c(r,q)} \leq \min_{t\in[q]} \frac{Mr\mu(H_k - H_{K-q})t}{K(H_K - H_{K-q+t-1})(r-K+t)^+M\mu}$$

$$\leq \min_{t\in[q]} \frac{t}{(r-K+t)^+} \cdot \left(1 + \frac{H_{K-q+t-1} - H_{K-q}}{H_K - H_{K-q+t-1}}\right)$$

$$\leq \frac{q/2}{(r-K+q/2)^+} \cdot \left(1 + \frac{\frac{q}{2}\frac{1}{K-q+1}}{\frac{q}{2}\frac{1}{K}}\right)$$

$$= \frac{q/2}{(r-K+q/2)^+} \cdot \left(1 + \frac{K}{K-q+1}\right). \tag{52}$$

When $r \geq K-n_1$ and $q \leq K(1-1/n_2)+1$ with integers $0 \leq n_1 < q/2$ and $n_2 \geq 1$, we have $\frac{q/2}{(r-K+q/2)^+} \leq \frac{q/2}{q/2-n_1} \leq n_1+1$ and $K/(K-q+1) \leq n_2$, respectively, and consequently, we have $\tau_a^c/\tau_l^c \leq (1+n_1)(1+n_2)$. Since the NULT is optimal and increases strictly with r, the repetition order satisfies $r \geq K-n_1$ when the NULT $\tau^u \geq \tau_a^u(K-n_1)$.

We thus prove the lower bound of NCT in (20) and the order-optimality of NCT in (22).

C. Lower Bound and Multiplicative Gap Analysis of NDLT

1) Lower bound: For a particular task assignment policy $\{\mathcal{U}_{i,\mathcal{K}'}\}$ satisfying (31) and (32), and a particular subset of q ENs denoted as $\mathcal{K}_q \subseteq \mathcal{K}$ whose outputs are available, each EN $k \in \mathcal{K}_q$ is assigned $r_{i,k}N$ input vectors from each user $i \in \mathcal{M}$ and can store μm rows of $\mathbf{A}$. Since each user i wants mN row-vector product results $\{\mathbf{v}_{i,j} = \mathbf{A}_{m\times n}\mathbf{u}_{i,j}\}_{j\in[N]}$, it is equivalent to state that each EN k can store $r_{i,k}\mu$ fractional outputs desired by each user i, denoted as $\mathcal{S}_{i,k} \triangleq \{\mathbf{A}_k\mathbf{u}_{i,j} : \mathbf{u}_{i,j} \in \mathcal{U}_{i,\mathcal{K}'}, k \in \mathcal{K}'\}$ and with size $|\mathcal{S}_{i,k}| = \gamma_{i,k}\mu NmB$ bits, where $\gamma_{i,k}$ satisfies (31) and (32). Thus, the policy $\{\mathcal{U}_{i,\mathcal{K}'}\}_{i\in\mathcal{M},k\in\mathcal{K}}$ with an available EN set $\mathcal{K}_q$ is equivalent to a particular computation results distribution $\{\mathcal{S}_{i,k}\}_{i\in\mathcal{M},k\in\mathcal{K}_q}$.

Let $\mathcal{M}_t \subseteq \mathcal{M}$ denote an arbitrary subset of t users and $\mathcal{Q}_{q-t} \subseteq \mathcal{K}_q$ denote an arbitrary subset of $q-t$ ENs. Also, we have $\mathcal{M}_{M-t} = \mathcal{M}/\mathcal{M}_t$ and $\mathcal{Q}_t = \mathcal{K}_q/\mathcal{Q}_{q-t}$. For a particular computation results distribution $\{\mathcal{S}_{i,k}\}_{i\in\mathcal{M},k\in\mathcal{K}_q}$, we adopt the arguments proved in [21, Lemma 6] to derive

the lower bound of the NDLT, i.e., intuitively, as shown in Fig. 8, *given any subset of t signals received at $t \leq \min\{q, M\}$ users, denoted as $\{Y_i\}_{i \in \mathcal{M}_t}$, and the stored computation results information of $q - t$ ENs, denoted as $\{\mathcal{S}_{i,k}\}_{i \in \mathcal{M}, k \in \mathcal{Q}_{q-t}}$, all transmitted signals $\{X_k\}_{k \in \mathcal{K}_q}$ and all the desired outputs $\{\mathbf{v}_{i,j}\}_{i \in \mathcal{M}, j \in [N]}$ can be resolved in the high-SNR regime.* First, we have the following equality,

$$MNmB = H\big(\{\mathbf{v}_{i,j}\}_{i \in \mathcal{M}, j \in [N]}\big)$$
$$= I\big(\{\mathbf{v}_{i,j}\}_{i \in \mathcal{M}, j \in [N]}; \{Y_i\}_{i \in \mathcal{M}_t}, \{\mathcal{S}_{i,k}\}_{i \in \mathcal{M}, k \in \mathcal{Q}_{q-t}}\big) + H\big(\{\mathbf{v}_{i,j}\}_{i \in \mathcal{M}, j \in [N]} | \{Y_i\}_{i \in \mathcal{M}_t}, \{\mathcal{S}_{i,k}\}_{i \in \mathcal{M}, k \in \mathcal{Q}_{q-t}}\big).$$
(53)

For the first term, following steps in [21, Eq. (64)], we have

$$I\big(\{\mathbf{v}_{i,j}\}_{i \in \mathcal{M}, j \in [N]}; \{Y_i\}_{i \in \mathcal{M}_t}, \{\mathcal{S}_{i,k}\}_{i \in \mathcal{M}, k \in \mathcal{Q}_{q-t}}\big)$$

$$= I\big(\{\mathbf{v}_{i,j}\}_{i \in \mathcal{M}, j \in [N]}; \{Y_i\}_{i \in \mathcal{M}_t}\big) + I\big(\{\mathbf{v}_{i,j}\}_{i \in \mathcal{M}, j \in [N]}; \{\mathcal{S}_{i,k}\}_{i \in \mathcal{M}, k \in \mathcal{Q}_{q-t}} | \{Y_i\}_{i \in \mathcal{M}_t}\big)$$

$$\leq I\big(\{\mathbf{v}_{i,j}\}_{i \in \mathcal{M}, j \in [N]}; \{Y_i\}_{i \in \mathcal{M}_t}\big) + I\big(\{\mathbf{v}_{i,j}\}_{i \in \mathcal{M}, j \in [N]}; \{\mathcal{S}_{i,k}\}_{i \in \mathcal{M}, k \in \mathcal{Q}_{q-t}}, \{\mathbf{v}_{i,j}\}_{i \in \mathcal{M}_t, j \in [N]} | \{Y_i\}_{i \in \mathcal{M}_t}\big)$$

$$= I\big(\{\mathbf{v}_{i,j}\}_{i \in \mathcal{M}, j \in [N]}; \{Y_i\}_{i \in \mathcal{M}_t}\big) + I\big(\{\mathbf{v}_{i,j}\}_{i \in \mathcal{M}, j \in [N]}; \{\mathbf{v}_{i,j}\}_{i \in \mathcal{M}_t, j \in [N]} | \{Y_i\}_{i \in \mathcal{M}_t}\big)$$

$$\qquad\qquad + I\big(\{\mathbf{v}_{i,j}\}_{i \in \mathcal{M}, j \in [N]}; \{\mathcal{S}_{i,k}\}_{i \in \mathcal{M}, k \in \mathcal{Q}_{q-t}}, | \{\mathbf{v}_{i,j}\}_{i \in \mathcal{M}_t, j \in [N]}, \{Y_i\}_{i \in \mathcal{M}_t}\big)$$

$$\leq I\big(\{\mathbf{v}_{i,j}\}_{i \in \mathcal{M}, j \in [N]}; \{Y_i\}_{i \in \mathcal{M}_t}\big) + H\big(\{\mathbf{v}_{i,j}\}_{i \in \mathcal{M}_t, j \in [N]} | \{Y_i\}_{i \in \mathcal{M}_t}\big)$$

$$+ H\big(\{\mathcal{S}_{i,k}\}_{i \in \mathcal{M}, k \in \mathcal{Q}_{q-t}} | \{\mathbf{v}_{i,j}\}_{i \in \mathcal{M}_t, j \in [N]}, \{Y_i\}_{i \in \mathcal{M}_t}\big) - H\big(\{\mathcal{S}_{i,k}\}_{i \in \mathcal{M}, k \in \mathcal{Q}_{q-t}} | \{\mathbf{v}_{i,j}\}_{i \in \mathcal{M}, j \in [N]}, \{Y_i\}_{i \in \mathcal{M}_t}\big)$$

$$\overset{(j)}{\leq} h\big(\{Y_i\}_{i \in \mathcal{M}_t}\big) - h\big(\{Y_i\}_{i \in \mathcal{M}_t} | \{\mathbf{v}_{i,j}\}_{i \in \mathcal{M}, j \in [N]}\big) + tNmB\epsilon + H\big(\{\mathcal{S}_{i,k}\}_{i \in \mathcal{M}, k \in \mathcal{Q}_{q-t}} | \{\mathbf{v}_{i,j}\}_{i \in \mathcal{M}_t, j \in [N]}\big)$$

$$\overset{(k)}{\leq} tT \log\big(2\pi e(\Lambda P^{\mathsf{d}} + 1)\big) - h\big(\{n_i\}_{i \in \mathcal{M}_t} | \{\mathbf{v}_{i,j}\}_{i \in \mathcal{M}, j \in [N]}\big) + tNmB\epsilon + \sum_{k \in \mathcal{Q}_{q-t}} H\big(\{\mathcal{S}_{i,k}\}_{i \in \mathcal{M}} | \{\mathbf{v}_{i,j}\}_{i \in \mathcal{M}_t, j \in [N]}\big)$$

$$\overset{(l)}{\leq} tT \log\big(2\pi e(\Lambda P^{\mathsf{d}} + 1)\big) - tT \log(2\pi e) + tNmB\epsilon + \sum_{k \in \mathcal{Q}_{q-t}} \sum_{i \in \mathcal{M}_{M-t}} H\big(\mathcal{S}_{i,k}\big)$$

$$\leq tT \log\big(\Lambda P^{\mathsf{d}} + 1\big) + tNmB\epsilon + \sum_{k \in \mathcal{Q}_{q-t}} \sum_{i \in \mathcal{M}_{M-t}} \gamma_{i,k} \mu NmB,$$
(54)

where, in step (j), $\{Y_i\}$ are continuous random variables, the third term uses Fano's inequality, the fourth term is because dropping the condition increases the entropy, the last term is 0 since the storage information $\{\mathcal{S}_{i,k}\}$ are the functions of $\{\mathbf{v}_{i,j}\}_{i \in \mathcal{M}, j \in [N]}$; In step (k), the first term uses [21, Lemma 5], and note that Λ defined in [21, Lemma 5] is a constant only depending on downlink channel coefficients in $\mathbf{H}^{\mathsf{d}}$. For the second term, by [21, Lemma 6] that proves the adopted argument, we have

$$H\big(\{\mathbf{v}_{i,j}\}_{i \in \mathcal{M}, j \in [N]} | \{Y_i\}_{i \in \mathcal{M}_t}, \{\mathcal{S}_{i,k}\}_{i \in \mathcal{M}, k \in \mathcal{Q}_{q-t}}\big) \leq tNmB\epsilon + T \log \det\big(\mathbf{I}_{M-t} + \tilde{\mathbf{H}}^{\mathsf{d}}(\tilde{\mathbf{H}}^{\mathsf{d}})^H\big),$$
(55)

where the $(M-t) \times (M-t)$ matrix $\tilde{\mathbf{H}}^{\mathsf{d}}$ defined in [21, Lemma 6] only depends on the channel matrix $\mathbf{H}^{\mathsf{d}}$, and $\mathbf{I}_{M-t}$ is a $(M-t) \times (M-t)$ identity matrix. The expressions of $\tilde{\mathbf{G}}$ and Λ are omitted here since they can be treated as constants.

Substituting (54) and (55) into (53), we have

$$MNmB \leq tT \log\big(\Lambda P^{\mathsf{d}} + 1\big) + 2tNmB\epsilon + \sum_{k \in \mathcal{Q}_{q-t}} \sum_{i \in \mathcal{M}_{M-t}} \gamma_{i,k} \mu NmB + T \log \det\big(\mathbf{I}_{M-t} + \tilde{\mathbf{H}}^{\mathsf{d}}(\tilde{\mathbf{H}}^{\mathsf{d}})^H\big),$$
(56)

Moving T to the left side and dividing by $\frac{NmB}{\log P^{\mathrm{d}}}$, we have

$$\frac{T}{NmB/\log P^{\mathrm{d}}} \geq \frac{M - \sum\limits_{k \in \mathcal{Q}_{q-t}} \sum\limits_{i \in \mathcal{M}_{M-t}} \gamma_{i,k}\mu - 2t\epsilon}{t} \cdot \frac{t \log P^{\mathrm{d}}}{t \log\left(\Lambda P^{\mathrm{d}}+1\right) + \log \det\left(\mathbf{I}_{M-t} + \tilde{\mathbf{H}}^{\mathrm{d}}(\tilde{\mathbf{H}}^{\mathrm{d}})^{H}\right)}. \tag{57}$$

Taking $P^{\mathrm{d}} \to \infty$ and $\epsilon \to 0$ as $B \to \infty$, the minimum NDLT under the output distribution $\{\mathcal{S}_{i,k}\}_{i \in \mathcal{M}, k \in \mathcal{Q}_{q-t}}$ is lower bounded by

$$\tau^{\mathrm{d}*}(r, \mathcal{K}_q, \mathcal{Q}_{q-t}) \geq \frac{M - \sum\limits_{k \in \mathcal{Q}_{q-t}} \sum\limits_{i \in \mathcal{M}_{M-t}} \gamma_{i,k}\mu}{t}, \quad \forall \mathcal{Q}_{q-t} \subseteq \mathcal{K}_q. \tag{58}$$

Note that the adopted argument holds for any subset of $q-t$ ENs (see Fig. 8). Thus, by tasking the sum over all possible subset $\mathcal{Q}_{q-t} \subseteq \mathcal{K}_q$, we have

$$\begin{aligned}
\binom{q}{q-t} \tau^{\mathrm{d}*}(r, \mathcal{K}_q, q-t) &\geq \sum_{\mathcal{Q}_{q-t} \subseteq \mathcal{K}_q} \frac{M - \sum\limits_{k \in \mathcal{Q}_{q-t}} \sum\limits_{i \in \mathcal{M}_{M-t}} \gamma_{i,k}\mu}{t} \\
&= \frac{\binom{q}{q-t}M - \sum\limits_{i \in \mathcal{M}_{M-t}} \sum\limits_{\mathcal{Q}_{q-t} \subseteq \mathcal{K}_q} \sum\limits_{k \in \mathcal{Q}_{q-t}} \gamma_{i,k}\mu}{t} \\
&= \frac{\binom{q}{q-t}M - \sum\limits_{i \in \mathcal{M}_{M-t}} \binom{q-1}{q-t-1} \sum\limits_{k \in \mathcal{K}_q} \gamma_{i,k}\mu}{t}.
\end{aligned} \tag{59}$$

For the particular policy $\{\mathcal{U}_{i,\mathcal{K}'}\}$ with repetition order r and satisfying (31) and (32), this lower bound also holds for any subset $\mathcal{K}_q$ since $K-q$ stragglers occur randomly (see Fig. 8), by taking the sum over all possible subsets $\mathcal{K}_q \subseteq \mathcal{K}$, we have

$$\begin{aligned}
\binom{K}{q}\binom{q}{q-t} \tau^{\mathrm{d}*}(r, q, q-t) &\geq \sum_{\mathcal{K}_q \subseteq \mathcal{K}} \frac{\binom{q}{q-t}M - \sum\limits_{i \in \mathcal{M}_{M-t}} \binom{q-1}{q-t-1} \sum\limits_{k \in \mathcal{K}_q} \gamma_{i,k}\mu}{t} \\
&= \frac{\binom{K}{q}\binom{q}{q-t}M - \sum\limits_{i \in \mathcal{M}_{M-t}} \binom{q-1}{q-t-1} \sum\limits_{\mathcal{K}_q \subseteq \mathcal{K}} \sum\limits_{k \in \mathcal{K}_q} \gamma_{i,k}\mu}{t} \\
&= \frac{\binom{K}{q}\binom{q}{q-t}M - \sum\limits_{i \in \mathcal{M}_{M-t}} \binom{q-1}{q-t-1}\binom{K-1}{q-1} \sum\limits_{k \in \mathcal{K}} \gamma_{i,k}\mu}{t} \\
&\stackrel{(m)}{=} \frac{\binom{K}{q}\binom{q}{q-t}M - (M-t)\binom{q-1}{q-t-1}\binom{K-1}{q-1}r\mu}{t},
\end{aligned} \tag{60}$$

where (m) is due to (31). Remanaging (60), the lower bound of NDLT at pair (r,q) is given by

$$\tau^{\mathrm{d}*}(r, q, q-t) \geq \frac{M - (M-t)(q-t)\frac{r}{K}\mu}{t}, \tag{61}$$

Since the argument we adopt to derive (61) holds for $1 \leq t \leq \min\{q, M\}$, the lower bound of the minimum NDLT at pair (r,q) can be optimized as

$$\tau^{\mathrm{d}*}(r, q) \geq \tau^{\mathrm{d}}_l(r, q) = \max_{t \in \{1, \cdots, \min\{q, M\}\}} \frac{M - (M-t)(q-t)\frac{r}{K}\mu}{t}. \tag{62}$$

2) Multiplicative gap: By (14), the achievable NDLT is upper bounded by

$$\tau_a^d = \sum_{p_1=r-K+q}^{\min\{r,q\}} B_{p_1} \left(\sum_{p_2=l_{\min}}^{l_{\max}} \frac{B_{p_2}}{d_{p_1,M,p_2}^d} + \frac{B_{l_{p_1}-1}}{d_{p_1,M,l_{p_1}-1}^d} \right)$$

$$\overset{(m)}{\leq} \sum_{p_1=r-K+q}^{\min\{r,q\}} B_{p_1} \frac{\sum_{p_2=l_{\min}}^{l_{\max}} B_{p_2} + B_{l_{p_1}-1}}{d_{p_1,M,1}^d}$$

$$\overset{(n)}{\leq} \frac{1}{d_{r-K+q,M,1}^d}, \tag{63}$$

where (m) is because d_{p_1,M,p_2}^d increases with p_2 [34, Lemma 1] and (n) is because $d_{p_1,M,1}^d = p_1/(p_1+M-1)$ increases with p_1. By (62), we have $\tau_l^d(r,q) \geq M/\min\{q,M\}$, so the multiplicative gap satisfies

$$\frac{\tau_a^d}{\tau_l^d} \leq \frac{\min\{q,M\}}{d_{r-K+q,M,1}^d} = \frac{\min\{q,M\}(r-K+q+M-1)}{(r-K+q)M}. \tag{64}$$

If $q \leq M$, we have $\tau_a^d/\tau_l^d \leq \frac{q}{r-K+q}(\frac{q}{M} + \frac{M-1}{M} - \frac{K-r}{M}) \leq \frac{2q}{r-K+q} \leq \frac{2q}{q-n} \leq 2(n\mu+1)$ for $r \geq K-n$; otherwise, we have $\tau_a^d/\tau_l^d \leq 1 + \frac{M-1}{r-K+q} \leq 1 + \frac{q-1}{q-n} \leq 2+(n-1)\mu$ for $r \geq K-n$. Here, integer n satisfies $n \leq q - 1/\mu$ due to $(r-K+q)\mu \geq 1$. In summary, since $2(n\mu+1) > 2+(n-1)\mu$, we have $\tau_a^d/\tau_l^d \leq 2(n\mu+1)$ for $r \geq K-n$. Furthermore, when the NULT $\tau^u \geq \tau_a^u(K-n)$, the repetition order satisfies $r \geq K-n$. Thus, when $r = K$, or equivalently, $\tau^u \geq \tau_a^u(K)$, we have $\tau_a^d/\tau_l^d \leq 2$.

Next, consider the special case $\mu = 1$. For any input vectors with degrees of replication of p_1, the degrees of replication of their associated outputs is $p_2 = l_{\max} = l_{\min} = p_1$, and we also have $B_{p_2} = 1$, so the achievable NDLT in (14) can be simplified as

$$\tau_a^d = \sum_{p_1=r-K+q}^{\min\{r,q\}} B_{p_1} \frac{1}{d_{p_1,M,p_1}^d} \leq M \frac{\sum_{p_1=r-K+q}^{\min\{r,q\}} B_{p_1}}{\min\{r-K+q,M\}} = \frac{M}{\min\{r-K+q,M\}}. \tag{65}$$

Due to $\tau_l^d(r,q) \geq M/\min\{q,M\}$, we have $\tau_a^d/\tau_l^d \leq \min\{q,M\}/\min\{r-K+q,M\}$. It is seen that when $\tau^u(r) \geq \tau_a^u(M+K-q)$, we have $\tau_a^d = \tau_l^d = 1$ that is optimal; when $\tau_a^u(K-n) \leq \tau^u(r) < \tau_a^u(M+K-q)$, we have $\tau_a^d/\tau_l^d \leq q/(r-K+q) \leq q/(q-n) \leq n+1$, where the integer n satisfies $q-M < n \leq q-1$. We prove the lower bound of NDLT in (21) and the multiplicative gap in (23).

D. Outer Bound of Compute-Download Latency Region

Based on the feasible set $\mathcal{R}$ in (15) and the convexity of $\mathcal{T}^*(\tau^u)$ in Remark 1, for an NULT $\tau^u = \tau_a^u(r)$ in (12) for some r, an outer bound $\mathcal{T}_{out}(\tau^u)$ of the compute-download latency region is given as the convex hull of set $\{(\tau_l^c(r,q), \tau_l^d(r,q)): q \in [\lceil\frac{1}{\mu}\rceil + K - r : K]\}$.

REFERENCES

[1] K. Li, M. Tao, J. Zhang, and O. Simeone, "Multi-cell mobile edge coded computing: Trading communication and computing for distributed matrix multiplication," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, 2020, pp. 215–220.

[2] S. Barbarossa, S. Sardellitti, and P. D. Lorenzo, "Communicating while computing: Distributed mobile cloud computing over 5G heterogeneous networks," *IEEE Signal Process. Mag.*, vol. 31, no. 6, pp. 45–55, Nov. 2014.

[3] ETSI, "Mobile edge computing: A key technology towards 5G," *White Paper*, Sep. 2015. [Online]. Available: http://www.etsi.org/images/files/ETSIWhitePapers/etsi_wp11_mec_a_key_technology_towards_5g.pdf

[4] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta, and D. Sabella, "On multi-access edge computing: A survey of the emerging 5g network edge cloud architecture and orchestration," *IEEE Commun. Surveys Tutorials*, vol. 19, no. 3, pp. 1657–1681, 2017.

[5] J. Dean and L. A. Barroso, "The tail at scale," *Communications of the ACM*, vol. 56, no. 2, pp. 74–80, 2013.

[6] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, 2009.

[7] R. Bekkerman, M. Bilenko, and J. Langford, *Scaling up machine learning: Parallel and distributed approaches*.   Cambridge University Press, 2011.

[8] S. Li, M. A. Maddah-Ali, Q. Yu, and A. S. Avestimehr, "A fundamental tradeoff between computation and communication in distributed computing," *IEEE Trans. Inf. Theory*, vol. 64, no. 1, pp. 109–128, 2017.

[9] K. Lee, M. Lam, R. Pedarsani, D. Papailiopoulos, and K. Ramchandran, "Speeding up distributed machine learning using codes," *IEEE Trans. Inf. Theory*, vol. 64, no. 3, pp. 1514–1529, 2017.

[10] S. Li, M. A. Maddah-Ali, and A. S. Avestimehr, "A unified coding framework for distributed computing with straggling servers," in *Proc. IEEE Globecom Workshops (GC Wkshps)*, Dec. 2016, pp. 1–6.

[11] J. Zhang and O. Simeone, "Improved latency-communication trade-off for map-shuffle-reduce systems with stragglers," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Process. (ICASSP)*, May 2019, pp. 8172–8176.

[12] A. Severinson, A. Graell i Amat, and E. Rosnes, "Block-diagonal and LT codes for distributed computing with straggling servers," *IEEE Trans. Commun.*, vol. 67, no. 3, pp. 1739–1753, 2019.

[13] J. Zhang and O. Simeone, "On model coding for distributed inference and transmission in mobile edge computing systems," *IEEE Commun. Letters*, vol. 23, no. 6, pp. 1065–1068, Jun. 2019.

[14] S. Dutta, V. Cadambe, and P. Grover, "Coded convolution for parallel and distributed computing within a deadline," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2017, pp. 2403–2407.

[15] S. Kiani, N. Ferdinand, and S. C. Draper, "Exploitation of stragglers in coded computation," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2018, pp. 1988–1992.

[16] Y. H. Ezzeldin, M. Karmoose, and C. Fragouli, "Communication vs distributed computation: An alternative trade-off curve," in *Proc. IEEE Inf. Theory Workshop (ITW)*, 2017, pp. 279–283.

[17] S. R. Srinivasavaradhan, L. Song, and C. Fragouli, "Distributed computing trade-offs with random connectivity," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, 2018, pp. 1281–1285.

[18] Q. Yan, S. Yang, and M. Wigger, "Storage, computation, and communication: A fundamental tradeoff in distributed computing," in *Proc. IEEE Inf. Theory Workshop (ITW)*, 2018, pp. 1–5.

[19] Q. Yan, M. Wigger, S. Yang, and X. Tang, "A fundamental storage-communication tradeoff in distributed computing with straggling nodes," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, 2019, pp. 2803–2807.

[20] N. Naderializadeh, M. A. Maddah-Ali, and A. S. Avestimehr, "Fundamental limits of cache-aided interference management," *IEEE Trans. Inf. Theory*, vol. 63, no. 5, pp. 3092–3107, May 2017.

[21] A. Sengupta, R. Tandon, and O. Simeone, "Fog-aided wireless networks for content delivery: Fundamental latency tradeoffs," *IEEE Trans. Inf. Theory*, vol. 63, no. 10, pp. 6650–6678, 2017.

[22] F. Xu, M. Tao, and K. Liu, "Fundamental tradeoff between storage and latency in cache-aided wireless interference networks," *IEEE Trans. Inf. Theory*, vol. 63, no. 11, pp. 7464–7491, Nov. 2017.

[23] N. Naderializadeh, M. A. Maddah-Ali, and A. S. Avestimehr, "Cache-aided interference management in wireless cellular networks," *IEEE Trans. Commun.*, vol. 67, no. 5, pp. 3376–3387, 2019.

[24] M. Tao, D. Gündüz, F. Xu, and J. S. P. Roig, "Content caching and delivery in wireless radio access networks," *IEEE Trans. Commun.*, vol. 67, no. 7, pp. 4724–4749, Jul. 2019.

[25] K. Li, M. Tao, and Z. Chen, "Exploiting computation replication for mobile edge computing: A fundamental computation-communication tradeoff study," *IEEE Trans. Wireless Commun.*, vol. 19, no. 7, pp. 4563–4578, 2020.

[26] S. Li, M. A. Maddah-Ali, and A. S. Avestimehr, "Communication-aware computing for edge processing," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2017, pp. 2885–2889.

[27] F. Li, J. Chen, and Z. Wang, "Wireless MapReduce distributed computing," *IEEE Trans. Inf. Theory*, vol. 65, no. 10, pp. 6101–6114, 2019.

[28] R. Lidl and H. Niederreiter, *Introduction to Finite Fields and their Applications*, 2nd ed.   Cambridge University Press, 1994.

[29] M. A. Maddah-Ali and D. Tse, "Completely stale transmitter channel state information is still very useful," *IEEE Trans. Inf. Theory*, vol. 58, no. 7, pp. 4418–4431, 2012.

[30] S. A. Jafar, "Blind interference alignment," *IEEE J. Sel. Topics Signal Process.*, vol. 6, no. 3, pp. 216–227, 2012.

[31] M. J. Abdoli, A. Ghasemi, and A. K. Khandani, "On the degrees of freedom of K-user SISO interference and X channels with delayed CSIT," *IEEE Trans. Inf. Theory*, vol. 59, no. 10, pp. 6542–6561, 2013.

[32] J. Hachem, U. Niesen, and S. N. Diggavi, "Degrees of freedom of cache-aided wireless interference networks," *IEEE Trans. Inf. Theory*, vol. 64, no. 7, pp. 5359–5380, 2018.

[33] B. C. Arnold, N. Balakrishnan, and H. Nagaraja, *A First Course in Order Statistics*.   SIAM, 2008, vol. 54.

[34] K. Li, M. Tao, and Z. Chen, "Exploiting computation replication in multi-user multi-server mobile edge computing networks," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2018, pp. 1–7.