Edge-Facilitated Wireless Distributed Computing

Songze Li*, Qian Yu*, Mohammad Ali Maddah-Ali †, and A. Salman Avestimehr*
* Department of Electrical Engineering, University of Southern California, Los Angeles, CA, USA
† Nokia Bell Labs, Holmdel, NJ, USA

Abstract

We propose a framework for edge-facilitated wireless distributed computing, in which several mobile users that are connected to an access point collaborate for a distributed computing task. We characterize the minimum communication load, both in uplink (from users to the access point) and downlink (from access point to the users), required for distributed computing in this framework. In particular, we develop a communication scheme and a dataset placement strategy that induces a particular overlap of computations at the users, which can then be exploited for coding at both users and the access point to significantly reduce the communication load. We demonstrate that the reduction in communication load (compared to uncoded solutions) can scale linearly with the size of the network (i.e., the number of users), hence our proposed scheme can result in a “scalable” design for edge-facilitated wireless distributed computing (i.e., accommodating any number of users without incurring extra communication load). Furthermore, we establish the optimality of the proposed scheme by developing a tight information theoretic outer-bound, and demonstrate that the proposed scheme achieves the minimum uplink and downlink communication load simultaneously. We also generalize the results to a decentralized setting, in which a random and a priori unknown subset of users may participate in distributed computing at each time, and characterize the minimum communication load for uniformly random dataset placement at users.

I. INTRODUCTION

In recent years, there has been a rapid growth in developing computationally intensive applications for mobile devices, such as mapping services, voice/image recognition, and augmented reality. The current trend for developing these applications is to offload computationally heavy tasks to a “cloud”, which has greater computational resources. While this trend has its merits, there is also a critical need for enabling wireless distributed computing, in which computation is carried out using the storage and computational resources of a cluster of wireless devices collaboratively. Wireless distributed computing eliminates, or at least de-emphasizes, the need for a core computing environment (i.e., the cloud), which is critical in several important applications, such as autonomous control and navigation for vehicles and drones, in which access to the cloud can be very limited. It is also expected to provide significant advantages to users by improving the QoS, increasing their computing capabilities, and enabling complex applications in machine learning, data analytics, and autonomous operation (see e.g., [1], [2]).

Communication, or data shuffling, is however a key bottleneck for enabling wireless distributed computing. In fact, even when the processing nodes are connected via high-bandwidth inter-server communication bus links, it is observed in [3] that 33% of the job execution time is spent on data shuffling. The communication bottleneck is expected to get much more severe as we move to using low-bandwidth wireless communication links for distributed computing (see e.g., [4], [5]).

As such motivated, our goal is to study the optimal design for wireless distributed computing, in order to minimize its communication load. We propose an edge-facilitated wireless distributed computing scenario, in which a group of mobile users, each can store parts of the dataset for computing, and collaborate for distributed computing tasks with the help of an access point at the edge of the network that they all wirelessly connect to. We consider a general distributed computing model at the mobile users, motivated by prevalent structures like MapReduce [6] and Spark [7], in which the overall computation is decomposed into three stages: “Map”, “Shuffle”, and “Reduce”. Firstly in the Map stage, users process parts of the dataset they have stored locally, generating some intermediate values according to their designed Map functions. Next, they exchange the calculated intermediate values in the Shuffle stage, in order to calculate the final results using their designed Reduce functions in the Reduce stage.

To illustrate this model, let us consider a motivating example, in which several mobile users aim to run an image recognition application to detect objects (e.g., people, automobiles, consumer products, etc.) that are captured on the cameras of their smartphones. Such a problem arises in many applications, such as content based search, augmented reality, automated navigation and surveillance, and medical computer vision. However, the dataset of these applications (i.e., the feature repository of objects) is very large, and cannot be downloaded and stored on a single device. Still, the aggregated memory size of these devices is large enough to collectively store the entire dataset. In the absence of a central processing unit (e.g. cloud), each user has to store only a fraction of the dataset. Later, users can exchange their images and collaborate on detecting the objects in the following manner. First, each user processes each image according to its locally stored dataset and maps it to an intermediate result (e.g. a short list). Then, in the data shuffling stage, each intermediate result is sent to the corresponding user via communication through an access point at the edge of the network. Each user will then locally process the received intermediate results, and reduce them to its final result. This motivating example represents a wide class of processing tasks that can be decomposed into (local) mapping, data shuffling, and then (local) reducing.
In such edge-facilitated wireless distributed computing frameworks, the communication loads both in uplink (from users to the access point) and downlink (from access point to the users) could become a bottleneck. Thus, the main objective of this paper is to minimize the load of the communication in both uplink and downlink. The design parameters are (1) dataset placement at users, i.e., the subset of the dataset that each device downloads and stores in its local memory, (2) the communication schemes in both uplink and downlink, i.e., efficiently forming the communication signals, considering what each user needs and exploiting what each user already has. We note that when the aggregated size of the memories is greater than the size of the dataset, the subsets of the dataset stored in different devices have some overlap. As we will see later, these overlaps create some coding opportunities that can improve the load of the communication. The challenge is how to design the dataset placement and communication schemes jointly to achieve the optimum uplink and downlink communication loads.

Our main result is to characterize the optimal region of uplink and downlink communication loads required for edge-facilitated wireless distributed computing. We propose an explicit dataset placement algorithm, and uplink and downlink communication schemes that can achieve any of the uplink-downlink load pairs in the region. The proposed scheme takes advantage of the imposed overlap structure of datasets placed at the users, and creates multicast coding opportunities that can significantly reduce the communication load. In fact, the load improvement is a linear function of the aggregated size of the local memories at the users, thus scales with the size of the network (number of the users in the system). As a result, our proposed scheme provides a scalable design for edge-facilitated wireless distributed computing (i.e., adding more users to the system does not effectively increase the required uplink and downlink communication loads). We also develop a tight information-theoretic outer-bound and establish the optimality of the proposed scheme. In particular, we show that the proposed scheme simultaneously achieves the minimum uplink and downlink communication loads.

We also generalize the framework to a decentralized setting, in which a random and a priori unknown subset of users participate in distributed computing at each time. We consider a uniform and random dataset placement strategy at the users, in which each user randomly, uniformly, and independently chooses and stores a subset of the dataset, up to the memory size. We completely characterize the optimal region of uplink and downlink communication loads required for distributed computing in such a decentralized setting, by proposing a coded uplink-downlink communication scheme and establishing a tight information-theoretic outer bound on the load region. We show that the proposed scheme performs very close to the centralized scheme in terms of the communication loads. In particular, as the number of participating users increases, the uplink and downlink communication loads of the decentralized setting converge to those of the centralized setting. This is in contrary to the fact that in decentralized setting, unlike the centralized one, the set and the number of participating users are not known in the dataset placement phase.

Relation to Prior Works. The problem of characterizing the minimum communication load required for distributed computing in MapReduce frameworks was recently proposed in [8], [9], for a wireline scenario where the computing nodes can directly communicate with each other through a shared link. In this paper, we extend this problem to a wireless setting, where distributed computing is performed on a set of wireless devices, and the data exchange across them can only be performed via a wireless access point. The idea of efficiently creating and exploiting coded multicasting opportunities was also initially proposed in the context of cache networks in [10], [11], and extended in [12], [13], where caches pre-fetch part of the content in a way to enable coding during the content delivery, minimizing the network traffic. In this paper, we demonstrate that such coding opportunities can also be utilized to reduce the communication load of edge-facilitated wireless distributed computing, by taking advantage of the overlap of computations at the users.

There have also been several recent works on communication design and resource allocation for mobile-edge computation offloading (see e.g., [14], [15]), in which the computation is offloaded to clouds located at the edges of cellular networks. In contrast to these works, in this paper our focus is on the scenario that the “edge” only facilitates the communication required for distributed computing, and all computations are done distributedly at the users.

II. SYSTEM MODEL AND PROBLEM FORMULATION

In this section, we describe the system model and problem formulation for the proposed edge-facilitated wireless distributed computing framework.

A. System Model

We consider a system that has \( K \) mobile users, for some \( K \in \mathbb{N} \). As illustrated in Fig. 1, all \( K \) users are connected wirelessly to an access point (e.g., a cellular base station or a Wi-Fi router) located at the edge of a mobile network. The uplink channels of the \( K \) users towards the access point are orthogonal to each other, and the signals transmitted by the access point on the downlink are received by all the users.

The system has a dataset (e.g. the feature repository of objects in the image recognition example mentioned in Section I) that is evenly partitioned into \( N \) files \( w_1, \ldots, w_N \in \mathbb{F}_2^D \), for some \( N, F \in \mathbb{N} \). Every User \( k, k \in \{1, \ldots, K\} \) has a length-\( D \) input \( d_k \in \mathbb{F}_2^D \) (e.g. user’s image in the image recognition example) to process using the \( N \) files. To do that, as shown in Fig. 1, User \( k, k \in \{1, \ldots, K\} \) needs to compute

\[
\phi(d_k; w_1, \ldots, w_N),
\]

input dataset (1)
where $\phi: \mathbb{F}_{2^D} \times (\mathbb{F}_{2^F})^N \rightarrow \mathbb{F}_{2^B}$ is an output function that maps the input $d_k$ to an output result (e.g., the returned result after processing the image) of length $B \in \mathbb{N}$.

We assume that every mobile user has a local memory that can store a $\mu$ fraction of the dataset, for some $0 < \mu \leq 1$. For each $k \in \{1, \ldots, K\}$, we denote the set of indices of the files stored by User $k$ as $\mathcal{U}_k$ where $|\mathcal{U}_k| \leq \mu N$. The selections of $\mathcal{U}_k$s are design parameters, and we denote the design of $\mathcal{U}_1, \ldots, \mathcal{U}_K$ as dataset placement. The dataset placement is performed in prior to the computation (e.g., users download parts of the feature repository when installing the image recognition application, before they start to use the application).

**B. Distributed Computing Model**

Motivated by prevalent distributed computing structures like MapReduce and Spark, we assume that as illustrated in Fig. 2 the computation of the output function for input $d_k$, $k \in \{1, \ldots, K\}$ can be decomposed as follows:

$$\phi(d_k; w_1, \ldots, w_N) = h(g_1(d_k; w_1), \ldots, g_N(d_k; w_N)), \quad (2)$$

where

- The “Map” functions $g_n(d_1, \ldots, d_K; w_n) = (g_n(d_1; w_n), \ldots, g_n(d_K; w_n)): (\mathbb{F}_{2^D})^K \times \mathbb{F}_{2^F} \rightarrow (\mathbb{F}_{2^T})^K$, $n \in \{1, \ldots, N\}$ maps the file $w_n$ into $K$ length-$T$ intermediate values $v_{k,n} = g_n(d_k; w_n) \in \mathbb{F}_{2^T}$, $k \in \{1, \ldots, K\}$, for some $T \in \mathbb{N}$.
- The “Reduce” function $h: (\mathbb{F}_{2^T})^N \rightarrow \mathbb{F}_{2^B}$ maps the intermediate values for input $d_k$ in all files into the output value $h(v_{k,1}, \ldots, v_{k,N})$, for all $k \in \{1, \ldots, K\}$.

We assume that the inputs $d_1, \ldots, d_K$ are known at every user (or the overhead of disseminating the inputs is negligible). Following the decomposition in (2), the overall computation proceeds in three phases: **Map**, **Shuffle**, and **Reduce**.

**Map Phase**: User $k$, $k \in \{1, \ldots, K\}$ computes the Map functions of the files in $\mathcal{U}_k$ stored locally. For each file $w_n$ in $\mathcal{U}_k$, User $k$ computes $g_n(d_1, \ldots, d_K; w_n) = (v_{k,1,n}, \ldots, v_{K,n})$.

**Shuffle Phase**: In order to compute the output value for the input $d_k$, User $k$ needs the intermediate values that are not computed locally in the Map phase, i.e., $\{v_{k,n} : n \notin \mathcal{U}_k\}$. In the Shuffle phase, users exchange the needed intermediate values with the help of the access point they all wirelessly connect to. As a result, the Shuffle phase breaks into two sub-phases: uplink communication and downlink communication.
On the uplink, every user \( k, k \in \{1, \ldots, K\} \), creates a message \( W_k \) as a function of the intermediate values computed locally during the Map phase, i.e.,

\[
W_k = \psi_k (\{ \tilde{g}_n : n \in \mathcal{U}_k \}), \tag{3}
\]

and then communicates \( W_k \) to the access point.

**Definition 1 (Uplink Communication Load).** We define the uplink communication load, denoted by \( L_u \), as the total number of bits in all uplink messages \( W_1, \ldots, W_K \), normalized by \( NT \), which is the total number of bits in all \( N \) intermediate values required by a user.

We assume that the access point does not have access to the dataset. Upon decoding all the messages from the \( K \) users \( W_1, \ldots, W_K \), the access point generates a message \( X \) that only depends on the decoded messages on the uplink, i.e.,

\[
X = \rho(W_1, \ldots, W_K). \tag{4}
\]

Then the server broadcasts \( X \) to all users on the downlink.

**Remark 2.** Here we assume that the \( K \) users that are scheduled to collaboratively perform the computing jobs all have the similar downlink channel strength. Therefore, the access point can broadcast the same message to all users. \( \square \)

**Definition 2 (Downlink Communication Load).** We define the downlink communication load, denoted by \( L_d \), as the number of bits in the downlink message \( X \), normalized by \( NT \).

**Reduce Phase:** User \( k, k \in \{1, \ldots, K\} \) uses the locally computed results \( \{ \tilde{g}_n : n \in \mathcal{U}_k \} \) and the decoded downlink message \( X \) to construct the inputs to the corresponding Reduce function, and calculates the output value \( h(v_{k,1}, \ldots, v_{k,N}) \).

**C. Problem Formulation**

We consider two scenarios, namely the centralized setting and the decentralized setting. In the centralized setting, the dataset placement is designed knowing which users will participate in the computing, while the dataset placement of the decentralized setting is performed without such information.

1) **Centralized Setting:** The set of the \( K \) users participating in the computation is fixed. The dataset placement \( \{\mathcal{U}_k\}_{k=1}^K \) is performed based on the identities of the \( K \) participating users. We assume that \( \mu K \geq 1 \), and every file in the dataset is stored by some user, i.e., \( k=1, \ldots, K \mathcal{U}_k = \{1, \ldots, N\} \).

For a centralized edge-facilitated wireless distributed computing framework with \( N \) files, and \( K \) mobile users that each can store a \( \mu \) fraction of the dataset, we say that a communication load pair \( (L_u, L_d) \in \mathbb{R}^2 \) is feasible if there exist a dataset placement \( \{\mathcal{U}_k : \mathcal{U}_k \subseteq \{1, \ldots, N\},|\mathcal{U}_k| \leq \mu N\}_{k=1}^K \), uplink encoding functions at the users and downlink encoding functions at the access point that achieve an uplink communication load \( L_u \) and a downlink communication load \( L_d \), such that User \( k \) can successfully calculate the intended output result \( h(v_{k,1}, \ldots, v_{k,N}) \), for all \( k \in \{1, \ldots, K\} \). We define the load region for the centralized setting

\[
\mathcal{L}_{\text{cent}}^* \triangleq \{(L_u, L_d) : (L_u, L_d) \text{ is feasible}\}. \tag{5}
\]

**Centralized Problem.** For centralized edge-facilitated wireless distributed computing with \( N \) files, \( K \) users each with a storage size of \( \mu \), we are interested in the following questions.

**Q1:** What is the load region \( \mathcal{L}_{\text{cent}}^* \)?

**Q2:** What are the dataset placements, uplink and downlink communication schemes to achieve the load pairs in \( \mathcal{L}_{\text{cent}}^* \)?

2) **Decentralized Setting:** We note that for most wireless distributed computing applications, among the many users who have installed the application, only a random subset of them are using it at each time. In such a scenario, the dataset placement at each user takes place at different times and is performed in a decentralized manner, i.e., a user downloads a part of the dataset without knowing the other users who have installed the application and when they will use it. As such motivated, we consider the following decentralized setting. We assume that there are many users in the system. In a computing instance, a random and a priori unknown subset of users, denoted by \( K \), participate in the computation. For each user \( k \) in the system, the dataset placement \( \mathcal{U}_k \) is designed (possibly randomly) without knowing the users who will participate in the computation (i.e., the design of \( \mathcal{U}_k \) is independent of \( K \)). In this paper, we limit our attention to a uniformly random dataset placement in which every user independently stores \( \mu N \) files uniformly at random.

We assume that for such a decentralized setting, once the computation starts, the participating users in \( K \) of size \( K \) are fixed, and their identities are revealed to all the participating users. Then they collaboratively perform the computation as in the centralized setting. The participating users process their inputs over the available part of the dataset stored in their local memories, i.e., \( \bigcup_{k \in K} \mathcal{U}_k \). More specifically, every participating user \( k \) in \( K \) now computes

\[
\phi\left(d_k : \{w_n : n \in \bigcup_{k \in K} \mathcal{U}_k\}\right). \tag{6}
\]

**Remark 3.** When the uniformly random dataset placement is used, with high probability for large \( N \), the fraction of the dataset available to participating users \( \frac{|\bigcup_{k \in K} \mathcal{U}_k|}{N} \) approximately equals \( 1 - (1 - \mu)^K \), which converges quickly to 1 when the number of participating users becomes large. \( \square \)
The computation proceeds in the three phases the same as before. The uplink and the downlink communication loads are also defined the same as before.

For decentralized edge-facilitated wireless distributed computing with $N$ files and every user independently storing a $\mu$ fraction of the dataset uniformly at random, we say that a load pair $(L_u, L_d) \in \mathbb{R}^2$ is $K$-feasible if with high probability for large $N$ and every subset of $K$ participating users, there exist uplink encoding functions at the users and downlink encoding functions at the access point that achieve an uplink communication load $L_u$ and a downlink communication load $L_d$, such that the participating user $k$ can successfully calculate the output function in (6), for all $k \in K$. For the decentralized setting with $K$ participating users and the uniformly random dataset placement, we define the $K$-participating load region

$$\mathcal{L}^*_{U,\text{decent}}(K) \triangleq \{(L_u, L_d) : (L_u, L_d) \text{ is } K\text{-feasible}\}. \quad (7)$$

**Decentralized Problem.** For decentralized edge-facilitated wireless distributed computing with $N$ files, $K$ participating users each with a storage size of $\mu$, and the uniformly random dataset placement, we are interested in the following questions.

- **Q3:** What is the $K$-participating load region $\mathcal{L}^*_{U,\text{decent}}(K)$?
- **Q4:** What are the uplink and downlink communication schemes to achieve the load pairs in $\mathcal{L}^*_{U,\text{decent}}(K)$?

### III. MAIN RESULTS

In this section, we present our main results which answer all four questions Q1-Q4 raised above. In particular, we characterize the optimal load regions, and provide matching converses demonstrating that the obtained regions cannot be improved by any other schemes.

#### A. Centralized Setting

We first state our result for the centralized setting.

**Theorem 1.** For the centralized edge-facilitated wireless distributed computing problem with a dataset of $N$ files, and $K$ users that each can store $\mu \geq \frac{1}{R}$ fraction of the files, the communication load region is characterized as

$$\mathcal{L}^*_{\text{cent}} = \{(L_u, L_d) : L_u \geq L^*_{\text{cent},u}(\mu), L_d \geq L^*_{\text{cent},d}(\mu)\}, \quad (8)$$

for sufficiently large $N$, where when $\mu \in \left\{\frac{1}{10}, \frac{2}{10}, \ldots, 1\right\}$,

$$L^*_{\text{cent},u}(\mu) = \frac{1}{\mu} - 1, \quad (9)$$

$$L^*_{\text{cent},d}(\mu) = \frac{\mu K}{\mu K + 1} \cdot \left(\frac{1}{\mu} - 1\right). \quad (10)$$

Otherwise,

$$L^*_{\text{cent},u}(\mu) = \text{Conv}\left(\frac{1}{\mu} - 1\right), \quad (11)$$

$$L^*_{\text{cent},d}(\mu) = \text{Conv}\left(\frac{\mu K}{\mu K + 1} \cdot \left(\frac{1}{\mu} - 1\right)\right), \quad (12)$$

where $\text{Conv}(f(\mu))$ denotes the lower convex envelop of the points $\{(\mu, f(\mu)) : \mu \in \left\{\frac{1}{10}, \ldots, 1\right\}\}$ for function $f(\mu)$.

**Remark 4.** Theorem 1 completely characterizes all feasible uplink-downlink load pairs for centralized edge-facilitated wireless distributed computing. In Section [IV], we propose an explicit design of the dataset placement, uplink and downlink communication schemes to achieve all load pairs in $\mathcal{L}^*_{\text{cent}}$. We also prove in Appendix, a matching information-theoretic outer bound of the load region, which implies that none of the load pairs outside $\mathcal{L}^*_{\text{cent}}$ is feasible.

![Fig. 3: Comparison of the load region $\mathcal{L}^*_{\text{cent}}$ in Theorem 1 with the load pairs achieved by the uncoded communication scheme $\mathcal{L}^*_{\text{cent unc}}$, for a system of $K = 20$ users and a storage size of $\mu = \frac{1}{10}$.](image)

**Remark 5.** Theorem 1 demonstrates that the load region $\mathcal{L}^*_{\text{cent}}$ has a very simple shape with only one corner point $(L^*_{\text{cent},u}(\mu), L^*_{\text{cent},d}(\mu))$, as illustrated in Fig. 3. This implies that $L^*_{\text{cent},u}(\mu)$ and $L^*_{\text{cent},d}(\mu)$ respectively denote the minimum communication loads required in the uplink and downlink to accomplish distributed computing. In other words, we can design a dataset placement strategy and a communication scheme that simultaneously minimize both uplink and downlink communication loads.
loads. Therefore, there is no tension between uplink communication and downlink communication in edge-facilitated wireless distributed computing.

**Remark 6.** Theorem 1 implies that, for large $K$,

$$L_{\text{cent},d}(\mu) \approx L_{\text{cent},u}(\mu) = \frac{1}{\mu} - 1,$$  \hspace{1cm} (13)

which is independent of the number of users $K$. Hence, quite surprisingly, Theorem 1 demonstrates that we can accommodate any number of users for wireless distributed computing without incurring extra communication load. This also implies that the dataset placement strategy that we propose in Section [V] provides a scalable design for edge-facilitated wireless distributed computing. As we show in Section [IV], the reason for this phenomenon is that, as more users joint the network, with an appropriate dataset placement, we can create coded multicasting opportunities at the users and the access point to reduce the communication loads by a factor that scales linearly with $K$. Such phenomenon was also observed in the context of cache networks (see e.g., [10]).

**Remark 7.** To further illustrate the impact of Theorem 1, we can consider an “uncoded” communication scheme, where each user receives the needed intermediate values sent uncodedly by some other users on the uplink and forwarded by the access point on the downlink. Since $K\mu N$ out of the total required $KN$ intermediate values are already available after the Map phase, the uncoded scheme achieves the load pairs

$$L_{\text{uncoded}} = \{ (L_u, L_d) : L_u, L_d \geq L_{\text{uncoded}}(\mu) = \mu K \cdot \left( \frac{1}{\mu} - 1 \right) \}.$$

We compare $L_{\text{uncoded}}$ with $L_{\text{cent}}^*$ in Theorem 1, and note that utilizing coding at the mobile users and the access point, we can reduce the minimum uplink and downlink communication load by a factor of $\mu K$ and $\mu K + 1$ respectively, which scale linearly with the aggregated storage size of the system. This is numerically demonstrated in Fig. 3 for a system with $K = 20$ users and a storage size of $\mu = \frac{1}{2}$ at each user.

### B. Decentralized Setting

We state our result for the decentralized setting as follows.

**Theorem 2.** In a decentralized edge-facilitated wireless distributed computing problem with a dataset of $N$ files, $K$ participating users that each can store a $\mu$ fraction of the dataset, and with uniformly random dataset placement at the users, the $K$-participating load region is characterized as

$$L_{\text{U-decent}}(K) = \{ (L_u, L_d) : L_u \geq L_{\text{U-decent},u}(K, \mu) = \sum_{j=1}^{K-1} \left( \begin{array}{c} K \vspace{0cm} \\ j+1 \end{array} \right) \mu^{j+1} (1-\mu)^{K-j}, \right.$$

$$L_d \geq L_{\text{U-decent},d}(K, \mu) = \sum_{j=1}^{K-1} \left( \begin{array}{c} K \vspace{0cm} \\ j+1 \end{array} \right) \mu^{j+1} (1-\mu)^{K-j} \}.$$  \hspace{1cm} (14)

**Remark 8.** Theorem 2 completely characterizes all feasible uplink and downlink load pairs for decentralized edge-facilitated wireless distributed computing. In Section [V] we propose uplink and downlink communication schemes to achieve all load pairs in $L_{\text{U-decent}}^*(K)$. We also prove a matching information-theoretic converse, which implies that none of the load pairs outside $L_{\text{U-decent}}^*(K)$ is feasible.

**Remark 9.** Theorem 2 illustrates that, similar to the centralized setting, the $K$-participating load region $L_{\text{U-decent}}^*(K)$ of the decentralized setting has a simple shape characterized by the corner point $(L_{\text{U-decent},u}(K, \mu), L_{\text{U-decent},d}(K, \mu))$. Hence under the uniformly random dataset placement, for any number of participating users $K$, we can design communication schemes to simultaneously achieve the minimum uplink and downlink communication loads.

---

![Minimum Communication Load](image.png)

Fig. 4: Comparison of the minimum communication loads of the centralized setting and the decentralized setting with uniformly random dataset placement, for a network of $K = 20$ participating users.
Remark 10. In Fig. 4 we numerically evaluate the minimum uplink and downlink communication loads for both the centralized setting and the decentralized setting with uniformly random dataset placement, in a network with 20 participating users. We observe that although the loads of the decentralized setting are higher than the loads of the centralized setting, the communication performances under these two settings are very close to each other. Hence, making the system decentralized only incurs a small increase in the communication load.

Remark 11. It is easy to show that
\[
\begin{align*}
\lim_{K \to \infty} L_{U-decent,u}(K, \mu) &= \frac{1}{\mu} - 1, \\
\lim_{K \to \infty} L_{U-decent,d}(K, \mu) &= \frac{1}{\mu} - 1.
\end{align*}
\]  
As a result, the minimum required uplink and downlink communication loads in both the centralized setting and the decentralized setting with uniformly random dataset placement all converge to \(\frac{1}{\mu} - 1\), as more users join the network.

IV. Coded Wireless Distributed Computing

In this section, we present a dataset placement strategy, an uplink communication scheme and a downlink communication scheme that achieve the load pairs stated in Theorem 1, particularly the corner point \(\left(\frac{1}{\mu} - 1, \frac{\mu K}{\mu K + 1} \cdot \left(\frac{1}{\mu} - 1\right)\right)\). First, we present an illustrative example to demonstrate the key ingredients of the schemes.

A. Example for \(N = 6, K = 3\) and \(\mu = \frac{2}{3}\)

We consider a problem with \(N = 6\) files, and \(K = 3\) users each can locally store \(\mu N = 4\) files.

1) Dataset Placement and Map Phase Execution: Every \(\mu K = 2\) users store a common file. More precisely as illustrated in Fig. 5, the sets of the files stored by the \(3\) users are \(U_1 = \{1, 2, 3, 4\}\), \(U_2 = \{3, 4, 5, 6\}\), and \(U_3 = \{5, 6, 1, 2\}\). In the Map phase, User \(k\), \(k \in \{1, 2, 3\}\) computes the intermediate values of all \(3\) inputs in the files stored locally in \(U_k\), i.e., \(\{v_{k,n} : k \in \{1, 2, 3\}, n \in U_k\}\).

2) Uplink Communication: We associate the locally computed intermediate values to the users, such that User 1 is responsible for sending \(\{v_{2,1}, v_{3,3}\}\), User 2 is responsible for sending \(\{v_{1,5}, v_{3,4}\}\), and User 3 is responsible for sending \(\{v_{1,6}, v_{2,2}\}\). Then as illustrated in Fig. 5 every user communicates the bit-wise XOR of its associated intermediate values specified above to the access point on the uplink.

3) Downlink Communication: Having decoded the messages from all \(3\) users on the uplink, the access point generates \(\mu K = 2\) random linear combinations of the XORed intermediate values \(C_1(v_{2,1} \oplus v_{3,3}, v_{1,5} \oplus v_{3,4}, v_{1,6} \oplus v_{2,2})\) and \(C_2(v_{2,1} \oplus v_{3,3}, v_{1,5} \oplus v_{3,4}, v_{1,6} \oplus v_{2,2})\). Then as shown in Fig. 5 the access point broadcasts these two combinations to all \(3\) users.

Since User 1 knows \(v_{2,1}, v_{3,3}, v_{3,4}\) and \(v_{2,2}\), he can decode \(v_{1,5}\) and \(v_{1,6}\) from the received two random linear combinations. Similarly, it is not difficult to verify that the other two users can also decode all the required intermediate values, with the help of their locally computed Map functions.

4) Communication Loads: Each user sends an XORed intermediate value in uplink, resulting in an uplink communication load \(L_u = \frac{2T}{NT} = \frac{3}{6} = \frac{3}{6}\). The access point broadcasts 2 random linear combinations of size \(T\) in downlink, yielding a downlink communication load \(L_d = \frac{2T}{NT} = \frac{3}{6} = \frac{1}{2}\).

Fig. 5: Illustration of the coded scheme to process inputs from \(K = 3\) mobile users each can store \(\mu N = 4\) out of \(N = 6\) files. Each file is mapped by \(\mu K = 2\) users store a common file. More precisely as illustrated in Fig. 5, the sets of the files stored by the \(3\) users are \(U_1 = \{1, 2, 3, 4\}\), \(U_2 = \{3, 4, 5, 6\}\), and \(U_3 = \{5, 6, 1, 2\}\). In the Map phase, User \(k\), \(k \in \{1, 2, 3\}\) computes the intermediate values of all \(3\) inputs in the files stored locally in \(U_k\), i.e., \(\{v_{k,n} : k \in \{1, 2, 3\}, n \in U_k\}\).
B. General Scheme

For a general edge-facilitated wireless distributed computing problem with \( N \) files, and \( K \) users each can store a subset of \( \mu N \in \{ \frac{N}{K}, \frac{2N}{K}, \ldots, N \} \) files, we assume that \( N \) is sufficiently large such that \( N = (\frac{K}{\mu K}) \eta \) for some \( \eta \in \mathbb{N} \).

1) Dataset Placement and Map Phase Execution: We evenly partitioned the \( N \) files into \( (\frac{K}{\mu K}) \) disjoint batches, each containing a subset of \( \eta \) files. Each batch of \( \eta \) files corresponds to a subset \( T \subset \{1, \ldots, K\} \) of \( |T| = \mu K \) users. That is

\[
\{1, \ldots, N\} = \{B_T : T \subset \{1, \ldots, K\}, |T| = \mu K\},
\]

where \( B_T \) denotes the batch of files corresponding to \( T \).

User \( k, k \in \{1, \ldots, K\} \) stores the files in \( B_T \) into \( U_k \) if \( k \in T \). After the Map phase, Node \( k, k \in \{1, \ldots, K\} \) knows the intermediate values of all \( K \) output functions in the files in \( U_k \), i.e., \( \{v_{q,n} : q \in \{1, \ldots, K\}, n \in U_k\} \).

2) Uplink Communication: For a subset \( S \subset \{1, \ldots, K\} \) and \( k \in \{1, \ldots, K\} \setminus S \), we denote the set of intermediate values needed by User \( k \) and known exclusively by all users in \( S \) as \( V_S^k \) after the Map phase. More formally:

\[
y_S^k = \{v_{k,n} : n \in \cap_{i \in S} U_i, n \notin \cup_{i \notin S} U_i\}.
\]

For all subsets \( S \subseteq \{1, \ldots, K\} \) of size \( \mu K + 1 \):

i) For each User \( k \in S \), we evenly and arbitrarily split \( V_S^k \), as defined in (18), into \( \mu K \) disjoint segments \( V_S^{k\setminus \{k\}} = \{V_{S\setminus \{k\}}, i : i \in S\setminus \{k\}\} \), where \( V_{S\setminus \{k\}, i} \) denotes the segment associated with User \( i \in S \setminus \{k\} \).

ii) User \( i, i \in S \), sends the bit-wise XOR, denoted by \( \oplus \), of all the segments associated with it in \( S \), i.e., Node \( i \) sends the coded segment \( W_i^S \triangleq \sum_{k \in S\setminus \{i\}} V_{S\setminus \{k\}, i} \).

Since the coded message \( W_i^S \) contains \( \frac{\mu K + 1}{\mu K} \cdot T \) bits for all \( i \in S \), there is a total of \( \frac{(\mu K + 1)T}{\mu K} \) bits communicated on the uplink in every subset \( S \) of size \( \mu K + 1 \). Therefore, the uplink communication load achieved by this coded scheme is

\[
L^*_{\text{cent, u}}(\mu) = \frac{(\frac{\mu K + 1}{\mu K}) \cdot (\mu K + 1) - \frac{1}{\mu K}}{\mu K + 1}, \mu \in \{1, \frac{2}{\mu K}, \ldots, 1\}.
\]

3) Downlink Communication: For all \( S \subseteq \{1, \ldots, K\} \) of size \( \mu K + 1 \), the access point computes \( \mu K \) random linear combinations of the messages received from the users in \( S \):

\[
C_j^S(\{W_i^S : i \in S\}), j = 1, \ldots, \mu K,
\]

and multicasts them to all users in \( S \).

Since each linear combination contains \( \frac{N}{\mu K} \cdot T \) bits, the coded scheme achieves a downlink communication load

\[
L^*_{\text{cent, d}}(\mu) = \frac{(\frac{\mu K + 1}{\mu K}) \cdot \frac{N}{\mu K} \cdot T}{\mu K + 1}, \mu \in \{1, \frac{2}{\mu K}, \ldots, 1\}.
\]

After receiving the random linear combinations \( C_1^S, \ldots, C_{\mu K}^S \), User \( k \in S \) can first cancel the message \( W_k^S \) he has sent on the uplink. Then for any user \( i \in S \setminus \{k\} \), User \( k \) can also cancel the segments \( V_{S \setminus \{k\}, i} \), associated with User \( i \) and needed by some other user \( k' \in S \setminus \{i, k\} \). Consequently, User \( k \) obtains \( \mu K \) random linear combinations of the required \( \mu K \) segments \( V_{S \setminus \{k\}, i} \), \( i \in S \setminus \{k\} \). Finally, since this holds for all \( S \subseteq \{1, \ldots, K\} \) of size \( |S| = \mu K + 1 \) and all \( k \in S \), the uplink and the downlink communication schemes together satisfy the data requests of all \( K \) users.

Remark 12. The above uplink and downlink communication schemes require coding at both the users and the access point, creating multicasting messages that are simultaneously useful for many users. Such idea of efficiently creating and exploiting coded multicasting opportunities was initially proposed in the context of cache networks in [10]–[12], and was recently utilized to substantially reduce the load of data shuffling across distributed computing nodes in [8], [9], [17].

When \( \mu K \) is not an integer, we can first expand \( \mu = \alpha \mu_1 + (1 - \alpha) \mu_2 \) as a convex combination of \( \mu_1 \triangleq \lfloor \mu K \rfloor / K \) and \( \mu_2 \triangleq \lceil \mu K \rceil / K \), for some \( 0 \leq \alpha \leq 1 \). Then we partition the set of \( N \) files \( \{w_1, \ldots, w_N\} \) into two disjoint subsets \( I_1 \) and \( I_2 \) of sizes \( |I_1| = \alpha N \) and \( |I_2| = (1 - \alpha) N \). We next apply the above coded scheme respectively to the files in \( I_1 \) where each file is stored at \( \mu_1 K \) users, and the files in \( I_2 \) where each file is stored at \( \mu_2 K \) users. This results in an uplink communication load and a downlink communication load of

\[
L^*_{\text{cent, u}}(\mu) = \alpha \left( \frac{1}{\mu_1} - 1 \right) + (1 - \alpha) \left( \frac{1}{\mu_2} - 1 \right),
\]

\[
L^*_{\text{cent, d}}(\mu) = \alpha \frac{\mu_1 K}{\mu_1 K + 1} \cdot \left( \frac{1}{\mu_1} - 1 \right) + (1 - \alpha) \frac{\mu_2 K}{\mu_2 K + 1} \cdot \left( \frac{1}{\mu_2} - 1 \right).
\]

V. DECENTRALIZED SETTING

We consider a decentralized edge-facilitated wireless distributed computing problem, with a dataset of \( N \) files, and \( K \) participating users each can store a \( \mu \) fraction of the dataset, and a uniformly random dataset placement. Such placement has also been considered in the decentralized caching problem in [11]. In this section, we prove the achievability part of Theorem 2 by demonstrating for each \( K \in \mathbb{N} \), an uplink communication scheme, and a downlink communication scheme that achieve the \( K \)-participating load pairs stated in Theorem 2. We also prove the lower bounds on the minimum uplink and downlink communication loads of the centralized setting.

1For small number of files \( N < \left(\frac{K}{\mu K}\right)\), we can apply the coded wireless distributed computing scheme to a subset of users, achieving a part of the gain in reducing the communication load.
A. Communication Scheme

Given a realization of the dataset placement, we perform the uplink and the downlink communications the same way as in the centralized setting, except that the selection of the subset $S$ is now over all subsets of sizes $|S| = 2, \ldots, K$.

Since the dataset placement is now randomized, in order for the above communication scheme to work properly, we zero-pad all elements in $\{Y_{S \setminus \{k\}}^k : k \in S\}$ to the maximum length $\max_{k \in S} |Y_{S \setminus \{k\}}^k|$, where $Y_{S \setminus \{k\}}^k$ is the set of bits needed by User $k$ and exclusively known by all users in $S \setminus \{k\}$.

B. Performance Analysis

Using the uniformly random dataset placement, for subset $S \subseteq \{1, \ldots, K\}$, the number of files exclusively stored by all users in $S$ can be characterized by $\mu|S|(1 - \mu)^{K - |S|}N + o(N)$.

Thus, when the proposed communication scheme proceeds on a subset $S$ of size $|S| = j + 1$ users, the resulting uplink communication load converges to $\frac{j + 1}{j} \mu^j (1 - \mu)^{K - j}$ for large $N$, and the downlink communication load converges to $\mu^j (1 - \mu)^{K - j}$ for large $N$.

Hence, we have the following total uplink and downlink communication loads

$$L^*_{\text{U-decent},u}(K, \mu) = \sum_{j=1}^{K-1} \left( \frac{K}{j+1} \right) \frac{j+1}{j} \mu^j (1 - \mu)^{K-j} ,$$

$$L^*_{\text{U-decent},d}(K, \mu) = \sum_{j=1}^{K-1} \left( \frac{K}{j+1} \right) \mu^j (1 - \mu)^{K-j} .$$

C. Converse of Theorem 2

We prove the optimality of the above communication schemes by deriving tight lower bounds on the uplink and downlink communication loads, for the decentralized setting with uniformly random dataset placement.

For any number of participating users $K$ and any realization of the data placement $\mathcal{U} \equiv \{\mathcal{U}_k\}_{k=1}^K$, we denote the minimum $K$-feasible uplink communication load and the minimum $K$-feasible downlink communication load as $L^*_u(K, \mathcal{U})$ and $L^*_d(K, \mathcal{U})$ respectively.

From Lemma 1 and Corollary 1 proved in Appendix, the following bounds on the minimum communication loads for a realization of the dataset placement $\mathcal{U}$ hold:

$$L^*_u(K, \mathcal{U}) \geq \sum_{j=1}^{K} \frac{a^j_{\mathcal{U}}}{N} \cdot \frac{K - j}{j} ,$$

$$L^*_d(K, \mathcal{U}) \geq \sum_{j=1}^{K} \frac{a^j_{\mathcal{U}}}{N} \cdot \frac{K - j}{j + 1} ,$$

where $a^j_{\mathcal{U}}$ denotes the number of files that are stored at $j$ users.

Using the uniformly random dataset placement for the decentralized setting, $\frac{a^j_{\mathcal{U}}}{N}$ converges to $\binom{K}{j} \mu^j (1 - \mu)^{K-j}$ in probability for large $N$.

Thus, any $K$-feasible pair of communication loads $(L_u, L_d)$ satisfies

$$L_u \geq \sum_{j=1}^{K-1} \left( \frac{K}{j+1} \right) \frac{j+1}{j} \mu^j (1 - \mu)^{K-j} = L^*_{\text{U-decent},u}(K, \mu) ,$$

$$L_d \geq \sum_{j=1}^{K-1} \left( \frac{K}{j+1} \right) \mu^j (1 - \mu)^{K-j} = L^*_{\text{U-decent},d}(K, \mu) .$$

APPENDIX

CONVERSE OF THEOREM 1

In this appendix, we prove the lower bounds on the minimum uplink and downlink communication loads of the centralized setting, denoted by

$$L^*_{\text{cent},s} = \inf\{L_s : L_s \text{ is feasible}\}, \ s \in \{u, d\} .$$  \hspace{1cm} (19)
A. Lower Bound on \( L^*_{cent,u} \)

To prove a lower bound on the minimum uplink communication load \( L^*_{cent,u} \) that is over all possible storage designs, we first characterize a lower bound for a particular storage design \( U \triangleq \{ U_k \}_{k=1}^K \). We denote the minimum uplink communication load under the storage design \( U \) as \( L^*_{cent,u}(U) \).

For a given storage design \( U \), we denote the number of files that are stored at \( j \) users as \( a_j^U \), for all \( j \in \{1, \ldots, K\} \):

\[
a_j^U = \sum_{A \subseteq \{1, \ldots, K\}: \#(A) = j} |( \bigcap_{k \in A} U_k ) \setminus ( \bigcup_{k \notin A} U_k )|.
\] (20)

For a particular storage design \( U \), we present a lower bound on \( L^*_{cent,u}(U) \) in the following lemma.

**Lemma 1.** \( L^*_{cent,u}(U) \geq \sum_{j=1}^{K} \frac{a_j^U}{N} \cdot \frac{K-j}{j} \).

In the rest of this subsection, we first demonstrate the lower bound on the minimum uplink communication load \( L^*_{cent,u} \) using Lemma 1, and then give the proof of Lemma 1.

**Proof of the Lower Bound on \( L^*_{cent,u} \).** It is clear that \( L^*_{cent,u} \) is lower bounded by the minimum value of \( L^*_{cent,u}(U) \) over all possible storage design for a storage size \( \mu \) at each user:

\[
L^*_{cent,u} \geq \inf_U L^*_{cent,u}(U).
\] (21)

Then by Lemma 1, we have

\[
L^*_{cent,u} \geq \inf_U \sum_{j=1}^{K} \frac{a_j^U}{N} \cdot \frac{K-j}{j}.
\] (22)

For every storage design \( U \), \( \{a_j^U\}_{j=1}^K \) satisfy

\[
\sum_{j=1}^{K} a_j^U = N,
\] (23)

\[
\sum_{j=1}^{K} ja_j^U = \mu NK.
\] (24)

Then since the function \( \frac{K-j}{j} \) in (22) is convex in \( j \), and by (23) that \( \sum_{j=1}^{K} a_j^U \frac{1}{N} = 1 \) and (24), (22) becomes

\[
L^*_{cent,u} \geq \inf_U \sum_{j=1}^{K} \frac{a_j^U}{N} \cdot \frac{K-j}{j} = \frac{K-\mu K}{\mu K} = \frac{1}{\mu} - 1.
\] (25)

We can further improve the lower bound in (25) when \( \mu K \notin \mathbb{N} \) as follows. For a variable \( \frac{1}{K} \leq k \leq 1 \), we first find the line \( p + qk \) connecting the two points \( (\mu_1, \frac{1}{\mu_1} - 1) \) and \( (\mu_2, \frac{1}{\mu_2} - 1) \), where \( \mu_1 \triangleq \lfloor \mu K \rfloor / K \) and \( \mu_2 \triangleq \lceil \mu K \rceil / K \), for some \( p, q \in \mathbb{R} \). Then by the convexity of the function \( \frac{1}{k} - 1 \), for all \( k \in \{ \frac{1}{K}, \ldots, 1 \} \),

\[
\frac{1}{k} - 1 \geq p + qk.
\] (26)

Then (22) reduces to

\[
L^*_{cent,u} \geq \inf_U \sum_{j=1}^{K} \frac{a_j^U}{N} \cdot \frac{K-j}{j} = \inf_U \sum_{k=\frac{1}{K}, \ldots, 1} \frac{a_j^U}{N} \cdot \left( \frac{1}{k} - 1 \right)
\] (27)

\[
= \frac{p + q\mu}{K}.
\] (28)

Therefore, for general \( \frac{1}{K} \leq \mu \leq 1 \), \( L^*_{cent,u} \) is lower bounded by the lower convex envelop of the points \( \{ (\mu_1, \frac{1}{\mu_1} - 1) \colon \mu \in \{ \frac{1}{K}, \frac{2}{K}, \ldots, 1 \} \} \).

Next, we proceed to prove Lemma 1. To do that, we develop a lower bound on the number of bits communicated by every subset of users, by induction on the size of the subset. In particular, for a subset of users, we first characterize the minimum number of bits required by one of the users, in order to recover the intermediate values needed for the intended Reduce function, and then combine this result with the lower bound on the number of bits communicated by the rest of the users in that subset, which is given by the inductive argument.
Proof of Lemma 1. For \( k \in \{1, \ldots, K\} \), \( n \in \{1, \ldots, N\} \), we let \( V_{k,n} \) be i.i.d. random variables uniformly distributed on \( F_{2^T} \). We let the intermediate values \( v_{k,n} \) be the realizations of \( V_{k,n} \). For some \( \mathcal{W} \subseteq \{1, \ldots, K\} \) and \( \mathcal{M} \subseteq \{1, \ldots, N\} \), we define
\[
V_{\mathcal{W}, \mathcal{M}} \triangleq \{v_{k,n} : k \in \mathcal{W}, n \in \mathcal{M}\}.
\]

Since each message \( W_k \) is generated as a function of the intermediate values that are computed at User \( k \), the following equation holds for all \( k \in \{1, \ldots, K\} \):
\[
H(W_k|V, \mathcal{U}_k) = 0.
\]

Since each user can decode the intended intermediate values using the locally computed Map functions and the downlink message \( X \) sent by the access point, for all \( k \in \{1, \ldots, K\} \), the following equation holds:
\[
H(V_{\{k\}}, |X, V, \mathcal{U}_k|) = 0.
\]

Since the downlink message \( X \) is a function of the uplink messages \( W_1, \ldots, W_K \), we have
\[
H(V_{\{k\}}, |W_k, X, \mathcal{U}_k|) = H(V_{\{k\}}, |W_k, \mathcal{U}_k|) = 0.
\]

For a subset \( S \subseteq \{1, \ldots, K\} \), let \( S^c = \{1, \ldots, K\}\setminus S \), and we define
\[
Y_{S^c} \triangleq (V_{S^c}, \mathcal{U}_{S^c}).
\]

For a storage design \( \mathcal{U} \), we denote the number of files that are exclusively stored by \( j \) users in \( S \) as \( a_{ij}^S \):
\[
a_{ij}^S = \sum_{J \subseteq S: |J| = j} |\{( \cap_{k \in J} \mathcal{U}_k) \setminus ( \cup_{i \notin J} \mathcal{U}_i)\}|.
\]

Then we prove the following statement by induction:

Claim 1. For any subset \( S \subseteq \{1, \ldots, K\} \), we have
\[
H(W_S|Y_{S^c}) \geq T \sum_{j=1}^{\lfloor |S|/2 \rfloor} a_{ij}^S \cdot \frac{|S| - j}{j}.
\]

a. If \( S = \emptyset \), obviously
\[
H(X_{\emptyset}|Y_{S^c}) = 0 = T \sum_{j=1}^0 a_{ij}^\emptyset \cdot \frac{0 - j}{j}.
\]

b. Suppose the statement is true for all subsets of size \( S_0 \).

For any \( S \subseteq \{1, \ldots, K\} \) of size \( |S| = S_0 + 1 \), and all \( k \in S \), the subset version of \( (32) \) and \( (34) \) can be derived:
\[
H(W_k|V_{\mathcal{M}_k}, Y_{S^c}) = 0,
\]
\[
H(V_{\{k\}}, |W_k, V, \mathcal{U}_k, Y_{S^c}|) = 0.
\]

Consequently, the following equation holds:
\[
H(W_S|V, \mathcal{U}_k, Y_{S^c}) = H(W_S|V_{\{k\}}, V, \mathcal{U}_k, Y_{S^c}) + H(V_{\{k\}}, |V, \mathcal{U}_k, Y_{S^c}|).
\]

Next we lower bound \( H(W_S|Y_{S^c}) \) as follows:
\[
H(W_S|Y_{S^c}) = \frac{1}{|S|} \sum_{k \in S} H(W_S, W_k|Y_{S^c})
\]
\[
= \frac{1}{|S|} \sum_{k \in S} (H(W_S|W_k, Y_{S^c}) + H(W_k|Y_{S^c}))
\]
\[
\geq \frac{1}{|S|} \sum_{k \in S} H(W_S|W_k, Y_{S^c}) + \frac{1}{|S|} H(W_S|Y_{S^c}).
\]

From \( (44) \), we can derive a lower bound on \( H(W_S|Y_{S^c}) \) that equals the LHS of \( (41) \) scaled by \( \frac{1}{|S|} \):
\[
H(W_S|Y_{S^c}) \geq \frac{1}{|S| - 1} \sum_{k \in S} H(W_S|W_k, Y_{S^c})
\]
\[
\geq \frac{1}{S_0} \sum_{k \in S} H(W_S|W_k, V, \mathcal{U}_k, Y_{S^c})
\]
\[
= \frac{1}{S_0} \sum_{k \in S} H(W_S|V, \mathcal{U}_k, Y_{S^c}).
\]
The first term on the RHS of (41) is lower bounded by the induction assumption:

\[ H(W_S|V_{\{k\}}, V_{:U_k}, Y_{S^c}) = H(W_{S\backslash\{k\}}|Y_{(S\backslash\{k\})^c}) \geq T \sum_{j=1}^{S_0} \alpha_{U_j}^{j,S\backslash\{k\}} \cdot \frac{S_0 - j}{j}. \] (49)

The second term on the RHS of (41) can be calculated based on the independence of intermediate values:

\[ H(V_{\{k\}}|V_{:U_k}, Y_{S^c}) = H(V_{\{k\}}|V_{:U_k}, V_{S^c, :}, V_{:U_{S^c}}) \]

\[ = T \sum_{j=0}^{S_0} \alpha_{U_j}^{j,S\{k\}} \]

\[ \geq T \sum_{j=1}^{S_0} \alpha_{U_j}^{j,S\{k\}}. \] (52)

Thus by (41), (47), (49) and (52), we have

\[ H(W_S|Y_{S^c}) \geq \frac{1}{S_0} \sum_{k \in S} H(W_S|V_{:U_k}, Y_{S^c}) \]

\[ = \frac{1}{S_0} \sum_{k \in S} (H(W_S|V_{\{k\}}, V_{:U_k}, Y_{S^c}) + H(V_{\{k\}}|V_{:U_k}, Y_{S^c})) \]

\[ \geq T \sum_{j=1}^{S_0} \sum_{k \in S} \alpha_{U_j}^{j,S\{k\}} \cdot \frac{S_0}{j}. \]

By the definition of \( \alpha_{U_j}^{j,S} \), we have the following equations:

\[ \sum_{k \in S} \alpha_{U_j}^{j,S}(k) = \sum_{k \in S} \sum_{f=1}^{N} \mathbb{I}(\text{file \( f \) is only stored by users in \( S\backslash\{k\} \)) \cdot \mathbb{I}(f \text{ is stored by } j \text{ users}) \]

\[ = \sum_{f=1}^{N} \mathbb{I}(\text{file \( f \) is only stored by } j \text{ users in } S) \sum_{k \in S} \mathbb{I}(f \text{ is not stored by User } k) \]

\[ = \sum_{f=1}^{N} \mathbb{I}(\text{file \( f \) is only stored by } j \text{ users in } S)(|S| - j) \]

\[ = a_{U_j}^{j,S}(S_0 + 1 - j). \]

Applying (60) to (66) yields

\[ H(X_S|Y_{S^c}) \geq T \sum_{j=1}^{S_0} a_{U_j}^{j,S} \cdot \frac{S_0 + 1 - j}{j}. \]

Thus for all subsets \( S \subseteq \{1, ..., K\} \), the following equation holds:

\[ H(W_S|Y_{S^c}) \geq T \sum_{j=1}^{|S|} a_{U_j}^{j,S} \cdot \frac{|S| - j}{j}. \]

which proves Claim 1.

Then by Claim 1, let \( S = \{1, ..., K\} \) be the set of all \( K \) users,

\[ L_{cent,u}^*(U) \geq \frac{H(W_S|Y_{S^c})}{NT} \geq \sum_{j=1}^{K} \frac{a_{U_j}^{j,S}}{N} \cdot \frac{K - j}{j}. \]

This completes the proof of Lemma 1.
B. Lower Bound on $L_{\text{cent},d}^*$

The lower bound of the minimum downlink communication load $L_{\text{cent},d}^*$ can be proved following the similar steps of lower bounding the minimum uplink communication load $L_{\text{cent},u}^*$, after making the following enhancements to the downlink communication system:

- We consider the access point as the $(K+1)$th user who has stored all $N$ files and has a virtual input to process. Thus the enhanced downlink communication system has $K+1$ users, and the storage design for the enhanced system is

$$\mathcal{U} \triangleq \{\mathcal{U}, \mathcal{U}_{K+1}\},$$

where $\mathcal{U}_{K+1}$ is constantly equal to $\{1, \ldots, N\}$.

- We assume that every one of the $K+1$ users can broadcast to the rest of the users, where the broadcast message is generated by mapping the locally stored files.

Apparently the minimum downlink communication load of the system cannot increase after the above enhancements. Thus the lower bound on the minimum downlink communication load of the enhanced system is also a lower bound for the original system.

Then we can apply the same arguments in the proof of Lemma 1 to the enhanced downlink system of $K+1$ users, obtaining a lower bound on $L_{\text{cent},d}^*(\mathcal{U})$ for a particular storage design $\mathcal{U}$, as described in the following corollary:

**Corollary 1.** $L_{\text{cent},d}^*(\mathcal{U}) \geq \sum_{j=1}^{K} \frac{a_j}{N} \cdot \frac{K+1-j}{j+1}$.

**Proof.** Applying Lemma 1 to the enhanced downlink system yields

$$L_{\text{cent},d}^*(\mathcal{U}) \geq \sum_{j=1}^{K+1} \frac{a_j}{N} \cdot \frac{K+1-j}{j} \geq \sum_{j=2}^{K+1} \frac{a_j}{N} \cdot \frac{K+1-j}{j}.$$

Since the access point has stored every file, $\frac{a_j}{N} = \frac{a_j}{N}$, for all $j \in \{1, \ldots, K\}$. Therefore, (66) can be re-written as

$$L_{\text{cent},d}^*(\mathcal{U}) \geq \sum_{j=1}^{K} \frac{a_j}{N} \cdot \frac{K-j}{j+1}.$$

Then following the same arguments as in the proof for the minimum uplink communication load, we have

$$L_{\text{cent},d}^* \geq \frac{K+1}{\mu K+1} = \frac{\mu K}{\mu K+1} \cdot (\frac{1}{\mu} - 1).$$

For general $\frac{1}{K} \leq \mu \leq 1$, $L_{\text{cent},d}^*$ is lower bounded by the lower convex envelop of the points $\{(\mu, \frac{\mu K}{\mu K+1} \cdot (\frac{1}{\mu} - 1)) : \mu \in \{\frac{1}{K}, \frac{2}{K}, \ldots, 1\}\}$.

**REFERENCES**


