A Fundamental Storage-Communication Tradeoff in Distributed Computing with Straggling Nodes

Qifa Yan, Michèle Wigger, Sheng Yang, Xiaohu Tang

A Fundamental Storage-Communication Tradeoff in Distributed Computing with Straggling Nodes

Qifa Yan, Michèle Wigger
LTCI, Télécom ParisTech
75013 Paris, France
Email: {qifa.yan, michele.wigger}
@telecom-paristech.fr

Sheng Yang
L2S, CentraleSupélec
91190 Gif-sur-Yvette, France
Email: sheng.yang@centralesupelec.fr

Xiaohu Tang
Information Security and National
Computing Grid Laboratory,
Southwest Jiaotong University,
611756, Chengdu, Sichuan, China
Email: xhutang@swjtu.edu.cn

***Abstract*—The optimal storage-computation tradeoff is characterized for a MapReduce-like distributed computing system with straggling nodes, where only a part of the nodes can be utilized to compute the desired output functions. The result holds for arbitrary output functions and thus generalizes previous results that restricted to linear functions. Specifically, in this work, we propose a new information-theoretical converse and a new matching coded computing scheme, that we call *coded computing for straggling systems* (CCS).**

I. Introduction

Distributed computing has emerged as one of the most important paradigms to speed up large-scale data analysis tasks such as machine learning. A well-known computing framework, MapReduce, can deal with tasks with large data size. In such systems, the tasks are typically decomposed into computing map and reduce functions, where the map functions can be computed by different nodes across the network, and the final outputs are computed by combining the outputs of the map functions by means of reduce functions. Such architectures can be used, for example, to perform learning tasks in neural networks [1].

Recently, Li *et al.* proposed a scheme named coded distributed computing (CDC), which through clever coding reduces the communication load for data shuffling between the map and reduce phases [2]. The CDC scheme achieves the optimal *storage-communication tradeoff*, i.e., the smallest communication load under a total memory constraint. More recently, this result was extended in [3], [4] to account also for the computation load during the map phase.

In this paper, we consider a variant of this problem where each node takes a random time to compute its desired map functions. In this case, waiting that all nodes have finished their computation can be too time consuming. Instead, the data shuffling and reduce computations are started as soon as *any* set of Q nodes, for Q a constant in $\{1,\dots,\mathsf{K}\}$, and K the number of total nodes, has terminated the map computations. One of the difficulties in such systems is that when assigning the map computations to the nodes, it is yet unclear which set of Q nodes will perform the data shuffling and compute the reduce functions. In this sense, the assignment of map computations needs to be convenient irrespective of the set of Q nodes that computes the reduce functions.

This setup with straggling nodes, hereafter referred to as the *straggling system*, has been introduced in [5]. That work used maximum separable codes (MDS) to set up coded computing. Similar codes were also used in [6], [7], and it was shown that they achieve the order optimal storage-communcation tradeoff in such straggling systems when the global task is to compute *linear functions*.

Many important tasks in practice, such as computations in neural networks, are however non-linear. This motivates us to investigate the MapReduce framework with straggling nodes for general map and reduce functions. In this work, we completely characterize the optimal storage-communcation tradeoff for systems with straggling nodes. This optimal tradeoff is achieved by our new scheme that we name *coded computing for straggling systems (CCS)*. We also present an information-theoretical converse matching our new CCS scheme.

Related to the present work are also distributed computing schemes with straggling nodes for master-worker network models, where the worker nodes perform the map computations and the master node the reduce computations. Specifically, [8]–[10] proposed coding schemes for systems with linear map and reduce functions and [11] proposed coding schemes for calculating the derivative of likelihood functions.

Notations: For positive integers n, k where $k \leq n$, we will use the notations $C_n^k \triangleq \frac{n!}{k!(n-k)!}$, $[n] \triangleq \{1,2,\dots,n\}$, and $[k:n] \triangleq \{k, k+1,\dots,n\}$. We use sans-serif font, e.g., K, to denote system parameters. The binary field is denoted by $\mathbb{F}_2$ and the n dimensional vector space over $\mathbb{F}_2$ is denoted by $\mathbb{F}_2^n$. The operator $|\cdot|$ is used in the following way: for a set $\mathcal{A}$, we use $|\mathcal{A}|$ to denote its cardinality, while for a signal X, we use $|X|$ to denote its length measured in bits. The notation $\Pr\{\cdot\}$ is used to denote probability of an event.

II. System Model

In this section, we define our model, referred to as the (K,Q) straggling system. This model is parameterized by positive integers $\mathsf{K},\mathsf{Q},\mathsf{N},\mathsf{D},\mathsf{U},\mathsf{V},\mathsf{W}$. Consider a system that aims to compute D output functions through K distributed computing nodes, denoted by $\mathcal{K} \triangleq [\mathsf{K}]$. Each output function ϕ_d ($d \in [\mathsf{D}]$) takes all files in the library $\mathcal{W} = \{w_n : n \in [\mathsf{N}]\}$

as inputs, where w_n is a file of size W bits, and outputs a bit stream of length U, i.e.,

$$u_d = \phi_d(w_1, \ldots, w_{\mathsf{N}}) \in \mathbb{F}_2^{\mathsf{U}},$$

where $\phi_d : \mathbb{F}_2^{\mathsf{NW}} \to \mathbb{F}_2^{\mathsf{U}}$. Assume that the computation of each output function ϕ_d can be decomposed as:

$$\phi_d(w_1, \ldots, w_{\mathsf{N}}) = h_d(f_{d,1}(w_1), \ldots, f_{d,\mathsf{N}}(w_{\mathsf{N}})),$$

where

- The *map function* $f_{d,n} : \mathbb{F}_2^{\mathsf{W}} \to \mathbb{F}_2^{\mathsf{V}}$ maps the file w_n into a binary stream of length V, called intermediate values (IVA), i.e.,

$$v_{d,n} \triangleq f_{d,n}(w_n) \in \mathbb{F}_2^{\mathsf{V}}, \quad \forall\ n \in [\mathsf{N}].$$

- The *reduce function* $h_d : \mathbb{F}_2^{\mathsf{NV}} \to \mathbb{F}_2^{\mathsf{U}}$ maps the IVAs $\{v_{d,n}\}_{n=1}^{\mathsf{N}}$ into the output stream

$$u_d = h_d(v_{d,1}, \ldots, v_{d,\mathsf{N}}).$$

Hence, the computation is carried out through three phases.

1) **Map Phase:** Each node $k \in \mathcal{K}$ stores a subset of files $\mathcal{M}_k \subseteq \mathcal{W}$, and tries to compute all the IVAs from the files in $\mathcal{M}_k$, denoted by $\mathcal{C}_k$:

$$\mathcal{C}_k \triangleq \{v_{d,n} : d \in [\mathsf{D}], w_n \in \mathcal{M}_k\}. \tag{1}$$

Each node has a random time to compute its corresponding IVAs. To limit latency of the system, the distributed computing scheme proceeds with the shuffle and reduce phases as soon as a fixed number of Q nodes has terminated the map computations. This subset of nodes will be called *active set*.

Notice that Q is a fixed system parameter. Moreover, for simplicity we assume that each subset $\mathcal{Q} \subseteq \mathcal{K}$ of size $|\mathcal{Q}| = \mathsf{Q}$ is active with same probability. Define

$$\mathfrak{T}_t \triangleq \{\mathcal{T} \subseteq \mathcal{K} : |\mathcal{T}| = t\}, \quad \forall\ t \in [\mathsf{K}],$$

and let $\mathbf{Q}$ denote the active set. Then,

$$\Pr\{\mathbf{Q} = \mathcal{Q}\} = \frac{1}{C_{\mathsf{K}}^{\mathsf{Q}}}, \quad \forall\ \mathcal{Q} \in \mathfrak{T}_{\mathsf{Q}}.$$

We also assume that all the files can be recovered[1] from any active set of size Q. Hence, for any file $w_n \in \mathcal{W}$, the number of nodes storing this file must satisfy $t_n \geq \mathsf{K}-\mathsf{Q}+1, \quad \forall\ n \in [\mathsf{N}]$, and hence $r \geq \mathsf{K} - \mathsf{Q} + 1$.

The output functions $\phi_1, \ldots, \phi_{\mathsf{D}}$ are then uniformly assigned to the nodes in $\mathbf{Q}$. Denote the indices of the output functions assigned to a given node $k \in \mathbf{Q}$ by $\mathcal{D}_{\mathbf{Q},k}$.[2] Thus, $\mathbf{D}_{\mathbf{Q}} \triangleq \{\mathcal{D}_{\mathbf{Q},k}\}_{k\in\mathbf{Q}}$ forms a partition of $[\mathsf{D}]$, and each $\mathcal{D}_{\mathbf{Q},k}$ is of cardinality $\frac{\mathsf{D}}{\mathsf{Q}}$. Denote the set of all uniform partitions of $[\mathsf{D}]$ by $\mathfrak{B}$.

2) **Shuffle Phase:** The nodes in $\mathbf{Q}$ proceed to exchange their computed IVAs. Each node k multicasts a signal

$$X_k^{\mathbf{Q}} = \varphi_{\mathbf{Q}}^{(k)}(\mathcal{C}_k, \mathbf{D}_{\mathbf{Q}})$$

to all other nodes in $\mathbf{Q}$, where $\varphi_{\mathbf{Q}}^{(k)} : \mathbb{F}_2^{|\mathcal{C}_k|\mathsf{V}} \times \mathfrak{B} \to \mathbb{F}_2^{|X_k^{\mathbf{Q}}|}$ is the encoding function of node k. Thus each active node $k \in \mathbf{Q}$ receives the signals $X^{\mathbf{Q}} \triangleq \left\{X_k^{\mathbf{Q}} : k \in \mathbf{Q}\right\}$ error-free.

3) **Reduce Phase:** With the received signals $X^{\mathbf{Q}}$ exchanged during the shuffle phase and the IVAs $\mathcal{C}_k$ computed locally during the map phase, node k restores all the IVAs

$$\{(v_{d,1}, \ldots, v_{d,\mathsf{N}})\}_{d\in\mathcal{D}_{\mathbf{Q},k}} = \psi_{\mathbf{Q}}^{(k)}\left(X^{\mathbf{Q}}, \mathcal{C}_k, \mathbf{D}_{\mathbf{Q}}\right),$$

where $\psi_{\mathbf{Q}}^{(k)} : \mathbb{F}_2^{\sum_{k\in\mathbf{Q}}|X_k^{\mathbf{Q}}|} \times \mathbb{F}_2^{|\mathcal{C}_k|\mathsf{V}} \times \mathfrak{B} \to \mathbb{F}_2^{\frac{\mathsf{NDV}}{\mathsf{Q}}}$. Subsequently, it proceeds to compute

$$u_d = h_d(v_{d,1}, \ldots, v_{d,\mathsf{N}}), \quad \forall\ d \in \mathcal{D}_{\mathbf{Q},k}. \tag{2}$$

Remark 1: Notice that a decomposition into map and reduce functions is always possible. In fact, trivially, one can set the map and reduce functions to be the identity and output functions respectively, i.e., $f_{d,n}(w_n) = w_n$, and $h_d = \phi_d, \ \forall\ n \in [\mathsf{N}], \ d \in [\mathsf{D}]$, in which case $\mathsf{V} = \mathsf{W}$. However, to mitigate the communication cost in shuffle phase, one would prefer a decomposition such that the length of the IVAs is small but suffices to compute the final outputs.

To measure the storage and computation costs, we introduce the following definitions:

Definition 1 (Storage Space): Storage space $\overline{r}$ is defined as the total number of files stored across the K nodes normalized by the total number of files N, i.e.,

$$\overline{r} \triangleq \frac{\sum_{k=1}^{\mathsf{K}} |\mathcal{M}_k|}{\mathsf{N}}.$$

Definition 2 (Communication Load): Communication load $\overline{L}$ is defined as the average total number of bits sent in the shuffle phase, normalized by the total number of bits of all intermediate values, i.e.,

$$\overline{L} = \mathbf{E}\left[\frac{\sum_{k\in\mathbf{Q}} |X_k^{\mathbf{Q}}|}{\mathsf{NDV}}\right],$$

where the expectation is taken with respect to the active set $\mathbf{Q}$.

Definition 3 (Optimal SC Tradeoff): A pair of real numbers (r, L) is achievable if for any $\epsilon > 0$, there exist positive integers $\mathsf{N}, \mathsf{D}, \mathsf{U}, \mathsf{V}, \mathsf{W}$, a storage design $\{\mathcal{M}_k\}_{k=1}^{\mathsf{K}}$ of storage space less than $r + \epsilon$, a set of uniform assignments of output functions $\{\mathbf{D}_{\mathcal{Q}}\}_{\mathcal{Q}\in\mathfrak{T}_{\mathsf{Q}}}$, and a collection of encoding functions $\left\{\left\{\varphi_{\mathcal{Q}}^{(k)}\right\}_{k\in\mathcal{Q}}\right\}_{\mathcal{Q}\in\mathfrak{T}_{\mathsf{Q}}}$ with communication load less than $L+\epsilon$, such that all the output functions $\phi_1, \ldots, \phi_{\mathsf{D}}$ can be computed successfully. For a fixed $\mathsf{Q} \in [\mathsf{K}]$, we define the optimal storage-communication (SC) tradeoff as

$$L_{\mathsf{K},\mathsf{Q}}^*(r) \triangleq \inf\{L : (r, L) \text{ is achievable}\}.$$

The goal of this paper is to characterize $L_{\mathsf{K},\mathsf{Q}}^*(r)$ for $r \in [\mathsf{K} - \mathsf{Q} + 1, \mathsf{K}]$ for each $\mathsf{Q} \in [\mathsf{K}]$.

[1]In this paper, we exclude the "outage" event in which some active set cannot compute the given function due to missing files.

[2]Here we assume for simplicity that Q divides D. Note that otherwise we can always add empty functions for the assumption to hold.

III. Main Result

We summarize our main result in the following theorem.

Theorem 1: For a (K, Q) straggling system, with a given integer storage space $r \in [\mathsf{K}-\mathsf{Q}+1 : \mathsf{K}]$, the optimal storage-communication tradeoff is

$$L^*_{\mathsf{K},\mathsf{Q}}(r) = \left(1 - \frac{r}{\mathsf{K}}\right) \cdot \sum_{l=r+\mathsf{Q}-\mathsf{K}}^{\min\{r,\mathsf{Q}-1\}} \frac{1}{l} \cdot \frac{C_r^l \cdot C_{\mathsf{K}-r-1}^{\mathsf{Q}-l-1}}{C_{\mathsf{K}-1}^{\mathsf{Q}-1}}.$$

In general, for $r \in [\mathsf{K}-\mathsf{Q}+1, \mathsf{K}]$, $L^*_{\mathsf{K},\mathsf{Q}}(r)$ is given by the lower convex envelope formed by the above points.

Theorem 1 characterizes the storage-communication tradeoff for all positive integers $\mathsf{Q} \leq \mathsf{K}$. Fig. 1 shows the curves $\left\{L^*_{\mathsf{K},\mathsf{Q}}(r) : \mathsf{Q} \in [\mathsf{K}]\right\}$ for $\mathsf{K} = 10$. Notice that, when $\mathsf{Q} = 1$, the curve reduces to a single point $(\mathsf{K}, 0)$, while when $\mathsf{Q} = \mathsf{K}$, the curve corresponds to the optimal tradeoff without straggling node obtained in [2]. In fact, in the case $\mathsf{Q} = \mathsf{K}$, our proposed scheme becomes the CDC scheme in [2].

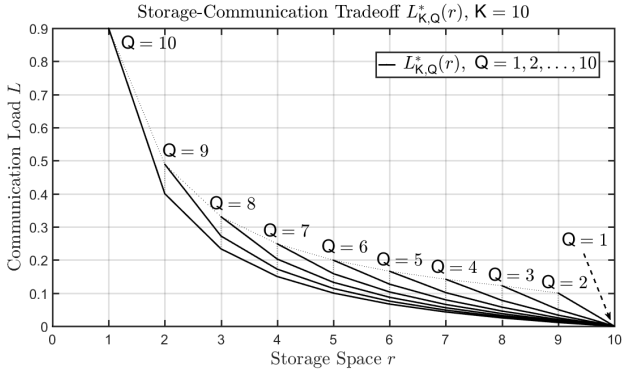

Fig. 1. Optimal Storage-Communication Tradeoff $L^*_{\mathsf{K},\mathsf{Q}}(r)$ for $\mathsf{Q} \in [\mathsf{K}]$ when $\mathsf{K} = 10$.

It is worth pointing out that the CCS scheme achieves the tradeoff without any assumption on the structure of map and reduce functions. Therefore, the files have to be stored in an uncoded fashion. Our converse relies on this assumption. If map and the reduce functions have certain properties, for example, linearity, a better SC tradeoff can be obtained with coded storage [6], [7]. A more detailed comparison of CCS scheme with linear map and reduce functions can be found in the long version of this paper [12].

We will see that, the storage design $\{\mathcal{M}_k\}_{k=1}^{\mathsf{K}}$ does not depend on the parameter Q but only on the available storage space r (See (3)). The proposed storage design is thus universally optimal irrespective of the size of the active set. In practice, the map phase can thus be carried out even without knowing how many nodes will be participating in the reduce phase. It turns out that, the CCS scheme can be generalized to a more general class of coded computing schemes with this property by means of a combination concept called placement delivery array [13], see [12] for details.

IV. Achievability

In this section, we first present our proposed scheme, and then analyze its performance to prove the achievability part of Theorem 1.

A. Coded Computing for Straggling Systems (CCS)

Let r be an integer value in $[\mathsf{K}-\mathsf{Q}+1 : \mathsf{K}]$. Consider N a multiple of C_{K}^r. Partition the files into C_{K}^r batches, each containing $\eta_r \triangleq \frac{\mathsf{N}}{C_{\mathsf{K}}^r}$ files. Each batch is then associated with a subset $\mathcal{T}$ of $\mathcal{K}$ of cardinality r. Let $\mathcal{W}_{\mathcal{T}}$ denote the batch of η_r files associated with set $\mathcal{T}$. Then,

$$\mathcal{W} = \{w_1, \ldots, w_{\mathsf{N}}\} = \bigcup_{\mathcal{T} \in \mathfrak{T}_r} \mathcal{W}_{\mathcal{T}}.$$

We now describe the map, shuffle, and reduce procedures of the CCS scheme.

1) Map Phase: Each node k stores batches $\mathcal{W}_{\mathcal{T}}$ such that $k \in \mathcal{T}$, i.e.,

$$\mathcal{M}_k = \bigcup_{\mathcal{T}: \mathcal{T} \in \mathfrak{T}_r,\, k \in \mathcal{T}} \mathcal{W}_{\mathcal{T}}, \tag{3}$$

and tries to compute all IVAs in (1).

2) Shuffle Phase: Assume the random active set $\mathbf{Q}$ be $\mathcal{Q}$. Pick any uniform assignment of output functions $\mathbf{D}_{\mathcal{Q}} = \{\mathcal{D}_{\mathcal{Q},k}\}_{k \in \mathcal{Q}}$. For any subset $\mathcal{T} \subseteq \mathcal{K}$ of size r and any $k \in \mathcal{K}$, let

$$\mathcal{V}_{\mathcal{T},k} \triangleq \{v_{d,n} : d \in \mathcal{D}_k,\, w_n \in \mathcal{W}_{\mathcal{T}}\}.$$

The shuffle phase is accomplished in $\min\{\mathsf{K}-\mathsf{Q}+1, \mathsf{K}-r\}$ rounds, where each round is indexed by an integer $l \in \big[r + \mathsf{Q} - \mathsf{K} : \min\{r, \mathsf{Q}-1\}\big]$. Consider round l, where we define all subsets of $\mathcal{K}$ of size r with exactly l active nodes:

$$\mathfrak{T}^l_{\mathcal{Q},r} \triangleq \{\mathcal{T} \subseteq \mathcal{K} :\ |\mathcal{T}| = r,\ |\mathcal{T} \cap \mathcal{Q}| = l\}.$$

For any $j \in \mathcal{Q}$ and any $\mathcal{T} \in \mathfrak{T}^l_{\mathcal{Q},r}$ with $j \notin \mathcal{T}$, evenly split the IVAs $\mathcal{V}_{\mathcal{T},j}$ into l disjoint packets that we label as $\{V^i_{\mathcal{T},j} : i \in \mathcal{T} \cap \mathcal{Q}\}$. So,

$$\mathcal{V}_{\mathcal{T},j} = \{V^i_{\mathcal{T},j} : i \in \mathcal{T} \cap \mathcal{Q}\}. \tag{4}$$

Each node $k \in \mathcal{Q}$ then multicasts the bit-wise XOR

$$X^k_{\mathcal{T}} \triangleq \bigoplus_{j \in \mathcal{T} \cap \mathcal{Q}} V^k_{(\mathcal{T}\setminus\{j\})\cup\{k\},j}, \quad \forall\, \mathcal{T} \in \mathfrak{T}^l_{\mathcal{Q},r} \text{ s.t. } k \notin \mathcal{T}. \tag{5}$$

Node k can compute all these signals because it can compute all IVAs $\left\{\mathcal{V}_{\mathcal{T}',j} : \mathcal{T}' \in \mathfrak{T}^l_{\mathcal{Q},r} \text{ and } k \in \mathcal{T}'\right\}$, see (3).

3) Reduce Phase: Any given node $k \in \mathcal{Q}$ can recover all IVAs

$$\{\mathcal{V}_{\mathcal{T},k} :\ \mathcal{T} \in \mathfrak{T}_r,\ k \in \mathcal{T}\}$$

locally, see again (3). To obtain the missing IVAs

$$\{\mathcal{V}_{\mathcal{T},k} :\ \mathcal{T} \in \mathfrak{T}_r,\ k \notin \mathcal{T}\}, \tag{6}$$

it again proceeds in rounds $l = r+\mathsf{Q}-\mathsf{K}, \ldots, \min\{r, \mathsf{Q}-1\}$. In round l, it forms for each $\mathcal{T} \in \mathfrak{T}^l_{\mathcal{Q},r}$ s.t. $k \notin \mathcal{T}$ and $j \in \mathcal{T} \cap \mathcal{Q}$ the IVA packet

$$V^j_{\mathcal{T},k} = \bigoplus_{i \in (\mathcal{T}\setminus\{j\}) \cap \mathcal{Q}} V^j_{(\mathcal{T}\setminus\{i\})\cup\{k\},i} \oplus X^j_{(\mathcal{T}\setminus\{j\})\cup\{k\}}. \tag{7}$$

Notice that,

1) Node k can form the signal (7) since it obtained the signal $X^j_{(\mathcal{T}\backslash\{j\})\cup\{k\}}$ from node j, and it can compute the IVAs $\left\{V^j_{(\mathcal{T}\backslash\{i\})\cup\{k\},i} : i \in (\mathcal{T}\backslash\{j\})\cap\mathcal{Q}\right\}$ locally.
2) Node k can decode all the IVAs in (6), since for any $\mathcal{T}\in\mathfrak{T}_r$ such that $k\notin\mathcal{T}$, $r+\mathsf{Q}-\mathsf{K}\le|\mathcal{T}\cap\mathcal{Q}|\le\min\{r,\mathsf{Q}-1\}$, and thus, $\mathcal{T}$ must be an element of $\mathfrak{T}^l_{\mathcal{Q},r}$, for some $l\in[r+\mathsf{Q}-\mathsf{K}:\min\{r,\mathsf{Q}-1\}]$.

Finally, it computes the output functions through (2).

B. Performance Analysis

By (3), each node k stores $C^{r-1}_{\mathsf{K}-1}$ batches, each of which containing η_r files. Thus, the storage space is

$$\overline{r}=\frac{\sum_{k=1}^{\mathsf{K}}|\mathcal{M}_k|}{\mathsf{N}}=\frac{\mathsf{K}\cdot\binom{\mathsf{K}-1}{r-1}\cdot\eta_r}{\mathsf{N}}=r.$$

For each $l\in\left[r+\mathsf{Q}-\mathsf{K}:\min\{r,\mathsf{Q}-1\}\right]$, by (4) and (5), there are in total $(\mathsf{Q}-l)\cdot C^l_{\mathsf{Q}}\cdot C^{r-l}_{\mathsf{K}-\mathsf{Q}}$ signals sent in the round l, each of size $\frac{\eta_r}{l}\cdot\frac{\mathsf{D}}{\mathsf{Q}}\cdot\mathsf{V}$ bits. Notice that, the communication load does not depend on the realization of the active set $\mathcal{Q}$ but only on its size Q. Hence, the average communication load is

$$\begin{aligned}\overline{L}&=\sum_{l=r+\mathsf{Q}-\mathsf{K}}^{\min\{r,\mathsf{Q}-1\}}\frac{(\mathsf{Q}-l)\cdot C^l_{\mathsf{Q}}\cdot C^{r-l}_{\mathsf{K}-\mathsf{Q}}}{\mathsf{NDV}}\cdot\frac{\eta_r}{l}\cdot\frac{\mathsf{D}}{\mathsf{Q}}\cdot\mathsf{V}\\&=\left(1-\frac{r}{\mathsf{K}}\right)\cdot\sum_{l=r+\mathsf{Q}-\mathsf{K}}^{\min\{r,\mathsf{Q}-1\}}\frac{1}{l}\cdot\frac{C^l_r\cdot C^{\mathsf{Q}-l-1}_{\mathsf{K}-r-1}}{C^{\mathsf{Q}-1}_{\mathsf{K}-1}}.\end{aligned}$$

We have proved the achievability when $r\in[\mathsf{K}-\mathsf{Q}+1:\mathsf{K}]$. For general $r\in[\mathsf{K}-\mathsf{Q}+1,\mathsf{K}]$, the lower convex envelope can be achieved with memory- and time- sharing.

V. Converse

Let $\{\mathcal{M}_k\}_{k=1}^{\mathsf{K}}$ be a storage design and (r,L) be a storage-communication pair achieved based on $\{\mathcal{M}_k\}_{k=1}^{\mathsf{K}}$. For each $s\in[\mathsf{K}-\mathsf{Q}+1:\mathsf{K}]$, define

$$a_{\mathcal{M},s}\triangleq\sum_{\mathcal{I}\subseteq\mathcal{K}:|\mathcal{I}|=s}\left|\left(\bigcap_{k\in\mathcal{I}}\mathcal{M}_k\right)\bigg\backslash\left(\bigcup_{\bar{k}\in\mathcal{K}\backslash\mathcal{I}}\mathcal{M}_{\bar{k}}\right)\right|,$$

i.e., $a_{\mathcal{M},s}$ is the number of files stored s times across all the nodes. Then by definition, $a_{\mathcal{M},s}$ satisfies

$$a_{\mathcal{M},s}\ge 0, \tag{8}$$

$$\sum_{s=\mathsf{K}-\mathsf{Q}+1}^{\mathsf{K}}a_{\mathcal{M},s}=\mathsf{N}, \tag{9}$$

$$\sum_{s=\mathsf{K}-\mathsf{Q}+1}^{\mathsf{K}}sa_{\mathcal{M},s}=\overline{r}\mathsf{N}. \tag{10}$$

For any $\mathcal{Q}\in\mathfrak{T}_{\mathsf{Q}}$ and any $l\in[\mathsf{Q}]$, define

$$b^{\mathcal{Q}}_{\mathcal{M},l}\triangleq\sum_{\mathcal{I}\subseteq\mathcal{Q}:|\mathcal{I}|=l}\left|\left(\bigcap_{k\in\mathcal{I}}\mathcal{M}_k\right)\bigg\backslash\left(\bigcup_{\bar{k}\in\mathcal{Q}\backslash\mathcal{I}}\mathcal{M}_{\bar{k}}\right)\right|,$$

i.e., $b^{\mathcal{Q}}_{\mathcal{M},l}$ is the number of files stored exactly l times in the nodes of $\mathcal{Q}$. Since any file that is stored s times across the all nodes, has l occurrences in exactly $C^l_s\cdot C^{\mathsf{Q}-l}_{\mathsf{K}-s}$ subsets of $\mathfrak{T}_{\mathsf{Q}}$, we have

$$\sum_{\mathcal{Q}\in\mathfrak{T}_{\mathsf{Q}}}b^{\mathcal{Q}}_{\mathcal{M},l}=\sum_{s=\max\{l,\mathsf{K}-\mathsf{Q}+1\}}^{\mathsf{K}-\mathsf{Q}+l}a_{\mathcal{M},s}\cdot C^l_s\cdot C^{\mathsf{Q}-l}_{\mathsf{K}-s}. \tag{11}$$

Consider the case that the active set is $\mathcal{Q}$, all the output functions are computed through the nodes in $\mathcal{Q}$. For the sub-system with computing nodes in $\mathcal{Q}$, under the storage $\{\mathcal{M}_k\}_{k\in\mathcal{Q}}$, the optimal communication load $L^*_{\mathcal{M},\mathcal{Q}}$ has the following lower bound by [2, Lemma 1]:

$$L^*_{\mathcal{M},\mathcal{Q}}\ge\sum_{l=1}^{\mathsf{Q}}\frac{b^{\mathcal{Q}}_{\mathcal{M},l}}{\mathsf{N}}\cdot\frac{\mathsf{Q}-l}{\mathsf{Q}l}.$$

Then the communication load

$$\begin{aligned}\overline{L}&=\mathbf{E}\left[\frac{\sum_{k\in\mathcal{Q}}|X^{\mathcal{Q}}_k|}{\mathsf{NDV}}\right]\\&=\sum_{\mathcal{Q}\in\mathfrak{T}_{\mathsf{Q}}}\mathbf{E}\left[\frac{\sum_{k\in\mathcal{Q}}|X^{\mathcal{Q}}_k|}{\mathsf{NDV}}\,\middle|\,\mathbf{Q}=\mathcal{Q}\right]\cdot\Pr\{\mathbf{Q}=\mathcal{Q}\}\\&\ge\sum_{\mathcal{Q}\in\mathfrak{T}_{\mathsf{Q}}}L^*_{\mathcal{M},\mathcal{Q}}\cdot\Pr\{\mathbf{Q}=\mathcal{Q}\}\\&\ge\frac{1}{C^{\mathsf{Q}}_{\mathsf{K}}}\cdot\sum_{\mathcal{Q}\in\mathfrak{T}_{\mathsf{Q}}}\sum_{l=1}^{\mathsf{Q}}\frac{b^{\mathcal{Q}}_{\mathcal{M},l}}{\mathsf{N}}\cdot\frac{\mathsf{Q}-l}{\mathsf{Q}l}\\&=\frac{1}{C^{\mathsf{Q}}_{\mathsf{K}}}\cdot\sum_{l=1}^{\mathsf{Q}}\left(\sum_{\mathcal{Q}\in\mathfrak{T}_{\mathsf{Q}}}\frac{b^{\mathcal{Q}}_{\mathcal{M},l}}{\mathsf{N}}\right)\cdot\frac{\mathsf{Q}-l}{\mathsf{Q}l}\\&\overset{(a)}{=}\sum_{l=1}^{\mathsf{Q}}\left(\sum_{s=\max\{l,\mathsf{K}-\mathsf{Q}+1\}}^{\mathsf{K}-\mathsf{Q}+l}\frac{a_{\mathcal{M},s}}{\mathsf{N}}\cdot\frac{C^l_sC^{\mathsf{Q}-l}_{\mathsf{K}-s}}{C^{\mathsf{Q}}_{\mathsf{K}}}\right)\cdot\frac{\mathsf{Q}-l}{\mathsf{Q}l}\\&=\sum_{s=\mathsf{K}-\mathsf{Q}+1}^{\mathsf{K}}\frac{a_{\mathcal{M},s}}{\mathsf{N}}\cdot\sum_{l=s+\mathsf{Q}-\mathsf{K}}^{\min\{s,\mathsf{Q}\}}\frac{\mathsf{Q}-l}{\mathsf{Q}l}\cdot\frac{C^l_sC^{\mathsf{Q}-l}_{\mathsf{K}-s}}{C^{\mathsf{Q}}_{\mathsf{K}}},\end{aligned} \tag{12}$$

where (a) follows from (11).

Let $Z^{\mathsf{Q}}_{\mathsf{K}}(x)$ be the function defined on the interval $[\mathsf{K}-\mathsf{Q}+1,\mathsf{K}]$ by connecting the points $\left(s,Z^{\mathsf{Q}}_{\mathsf{K}}(s)\right)$ sequentially, where

$$Z^{\mathsf{Q}}_{\mathsf{K}}(s)\triangleq\sum_{l=s+\mathsf{Q}-\mathsf{K}}^{\min\{s,\mathsf{Q}\}}\frac{\mathsf{Q}-l}{\mathsf{Q}l}\cdot C^l_s\cdot C^{\mathsf{Q}-l}_{\mathsf{K}-s} \tag{13}$$

is defined for any $s\in[\mathsf{K}-\mathsf{Q}+1:\mathsf{K}]$. In the Appendix, we will prove the following claim.

Claim 1: The function $Z^{\mathsf{Q}}_{\mathsf{K}}(x)$ is a convex decreasing function over $[\mathsf{K}-\mathsf{Q}+1,\mathsf{K}]$.

By (8) and (9), $\frac{a_{\mathcal{M},s}}{\mathsf{N}}$ are coefficients between 0 and 1 whose summation is 1. Hence by (12) and Claim 1, we have

$$\begin{aligned}\overline{L}&\ge\frac{1}{C^{\mathsf{Q}}_{\mathsf{K}}}\cdot\sum_{s=\mathsf{K}-\mathsf{Q}+1}^{\mathsf{K}}\frac{a_{\mathcal{M},s}}{\mathsf{N}}\cdot Z^{\mathsf{Q}}_{\mathsf{K}}(s)\\&\ge\frac{1}{C^{\mathsf{Q}}_{\mathsf{K}}}\cdot Z^{\mathsf{Q}}_{\mathsf{K}}\left(\sum_{s=\mathsf{K}-\mathsf{Q}+1}^{\mathsf{K}}\frac{sa_{\mathcal{M},s}}{\mathsf{N}}\right)\\&\overset{(a)}{=}\frac{Z^{\mathsf{Q}}_{\mathsf{K}}(\overline{r})}{C^{\mathsf{Q}}_{\mathsf{K}}},\end{aligned}$$

where (a) follows from (10). Combining the facts $L+\epsilon \geq \overline{L}, r+\epsilon \geq \overline{r}$ for any $\epsilon > 0$ and $Z_{\mathsf{K}}^{\mathsf{Q}}(x)$ is continuous over the interval $[\mathsf{K}-\mathsf{Q}+1, \mathsf{K}]$, we obtain

$$L \geq \frac{Z_{\mathsf{K}}^{\mathsf{Q}}(r)}{C_{\mathsf{K}}^{\mathsf{Q}}}, \quad \forall\, r \in [\mathsf{K}-\mathsf{Q}+1, \mathsf{K}].$$

In particular, when $r \in [\mathsf{K}-\mathsf{Q}+1 : \mathsf{K}]$, we have

$$L \geq \left(1-\frac{r}{\mathsf{K}}\right) \cdot \sum_{l=r+\mathsf{Q}-\mathsf{K}}^{\min\{r,\mathsf{Q}-1\}} \frac{1}{l} \cdot \frac{C_r^l \cdot C_{\mathsf{K}-r-1}^{\mathsf{Q}-l-1}}{C_{\mathsf{K}-1}^{\mathsf{Q}-1}}.$$

ACKNOWLEDGMENT

The work of Q. Yan and M. Wigger has been supported by the ERC under grant agreement 715111.

APPENDIX
PROOF OF CLAIM 1

We prove Claim 1 by verifying that, the sequence $\left\{Z_{\mathsf{K}}^{\mathsf{Q}}(s)\right\}_{s=\mathsf{K}-\mathsf{Q}+1}^{\mathsf{K}}$ is a discrete monotonically decreasing convex sequence, i.e,

$$Z_{\mathsf{K}}^{\mathsf{Q}}(s+1) - Z_{\mathsf{K}}^{\mathsf{Q}}(s) < 0, \tag{14}$$
$$Z_{\mathsf{K}}^{\mathsf{Q}}(s+1) - Z_{\mathsf{K}}^{\mathsf{Q}}(s) > Z_{\mathsf{K}}^{\mathsf{Q}}(s) - Z_{\mathsf{K}}^{\mathsf{Q}}(s-1). \tag{15}$$

Notice that by (13),

$$\begin{aligned} Z_{\mathsf{K}}^{\mathsf{Q}}(s) &= \sum_{l=s+\mathsf{Q}-\mathsf{K}}^{\min\{s,\mathsf{Q}\}} \frac{C_s^l \cdot C_{\mathsf{K}-s}^{\mathsf{Q}-l}}{l} - \sum_{l=s+\mathsf{Q}-\mathsf{K}}^{\min\{s,\mathsf{Q}\}} \frac{C_s^l \cdot C_{\mathsf{K}-s}^{\mathsf{Q}-l}}{\mathsf{Q}} \\ &\overset{(a)}{=} \sum_{l=s+\mathsf{Q}-\mathsf{K}}^{\min\{s,\mathsf{Q}\}} \frac{C_s^l \cdot C_{\mathsf{K}-s}^{\mathsf{Q}-l}}{l} - \frac{C_{\mathsf{K}}^{\mathsf{Q}}}{\mathsf{Q}}, \end{aligned}$$

where in (a), we used the equality $\sum_{l=s+\mathsf{Q}-\mathsf{K}}^{\min\{s,\mathsf{Q}\}} C_s^l \cdot C_{\mathsf{K}-s}^{\mathsf{Q}-l} = C_{\mathsf{K}}^{\mathsf{Q}}$. Thus the constant $\frac{C_{\mathsf{K}}^{\mathsf{Q}}}{\mathsf{Q}}$ shows up in both $Z_{\mathsf{K}}^{\mathsf{Q}}(s+1)$ and $Z_{\mathsf{K}}^{\mathsf{Q}}(s)$ for all $s \in [\mathsf{K}-\mathsf{Q}+1 : \mathsf{K}-1]$, then

$$\begin{aligned} &Z_{\mathsf{K}}^{\mathsf{Q}}(s+1) - Z_{\mathsf{K}}^{\mathsf{Q}}(s) \\ &= \sum_{l=s+1+\mathsf{Q}-\mathsf{K}}^{\min\{s+1,\mathsf{Q}\}} \frac{C_{s+1}^l C_{\mathsf{K}-s-1}^{\mathsf{Q}-l}}{l} - \sum_{l=s+\mathsf{Q}-\mathsf{K}}^{\min\{s,\mathsf{Q}\}} \frac{C_s^l C_{\mathsf{K}-s}^{\mathsf{Q}-l}}{l} \\ &\overset{(a)}{=} \sum_{l=s+1+\mathsf{Q}-\mathsf{K}}^{\min\{s+1,\mathsf{Q}\}} \frac{\left(C_s^l + C_s^{l-1}\right) \cdot C_{\mathsf{K}-s-1}^{\mathsf{Q}-l}}{l} \\ &\qquad - \sum_{l=s+\mathsf{Q}-\mathsf{K}}^{\min\{s,\mathsf{Q}\}} \frac{C_s^l \cdot \left(C_{\mathsf{K}-s-1}^{\mathsf{Q}-l} + C_{\mathsf{K}-s-1}^{\mathsf{Q}-l-1}\right)}{l} \\ &= \sum_{l=s+1+\mathsf{Q}-\mathsf{K}}^{\min\{s+1,\mathsf{Q}\}} \frac{C_s^{l-1} C_{\mathsf{K}-s-1}^{\mathsf{Q}-l}}{l} - \sum_{l=s+\mathsf{Q}-\mathsf{K}}^{\min\{s,\mathsf{Q}-1\}} \frac{C_s^l C_{\mathsf{K}-s-1}^{\mathsf{Q}-l-1}}{l} \\ &\overset{(b)}{=} \sum_{l=s+\mathsf{Q}-\mathsf{K}}^{\min\{s,\mathsf{Q}-1\}} \frac{C_s^l C_{\mathsf{K}-s-1}^{\mathsf{Q}-l-1}}{l+1} - \sum_{l=s+\mathsf{Q}-\mathsf{K}}^{\min\{s,\mathsf{Q}-1\}} \frac{C_s^l C_{\mathsf{K}-s-1}^{\mathsf{Q}-l-1}}{l} \\ &= -\sum_{l=s+\mathsf{Q}-\mathsf{K}}^{\min\{s,\mathsf{Q}-1\}} \frac{C_s^l C_{\mathsf{K}-s-1}^{\mathsf{Q}-l-1}}{l(l+1)} \\ &< 0, \end{aligned} \tag{16}$$

where in (a), we applied the equality $C_{n+1}^{m+1} = C_n^{m+1} + C_n^m$ and in (b), we used the variable change $l' = l-1$ in the first summation.

Moreover, following similar steps as above, by (16), for $s \in [\mathsf{K}-\mathsf{Q}+2 : \mathsf{K}-1]$, we can prove

$$\begin{aligned} &\left(Z_{\mathsf{K}}^{\mathsf{Q}}(s+1) - Z_{\mathsf{K}}^{\mathsf{Q}}(s)\right) - \left(Z_{\mathsf{K}}^{\mathsf{Q}}(s) - Z_{\mathsf{K}}^{\mathsf{Q}}(s-1)\right) \\ &= \sum_{l=\mathsf{Q}+s-1-\mathsf{K}}^{\min\{s,\mathsf{Q}\}-1} \frac{C_{s-1}^l C_{\mathsf{K}-s}^{\mathsf{Q}-l-1}}{l(l+1)} - \sum_{l=\mathsf{Q}+s-\mathsf{K}}^{\min\{s,\mathsf{Q}-1\}} \frac{C_s^l C_{\mathsf{K}-s-1}^{\mathsf{Q}-l-1}}{l(l+1)} \\ &= \sum_{l=s+\mathsf{Q}-\mathsf{K}}^{\min\{s,\mathsf{Q}-1\}} \frac{2C_{s-1}^{l-1} C_{\mathsf{K}-s-1}^{\mathsf{Q}-l-1}}{(l-1)l(l+1)} \\ &> 0. \end{aligned}$$

We have proved (14) and (15). As a result, the piecewise linear function $Z_{\mathsf{K}}^{\mathsf{Q}}(x)$ connecting points $\left\{\left(s, Z_{\mathsf{K}}^{\mathsf{Q}}(s)\right)\right\}_{s=\mathsf{K}-\mathsf{Q}+1}^{\mathsf{K}}$ sequentially is a convex function.

REFERENCES

[1] Y. Liu, J. Yang, Y. Huang, L. Xu, S. Li, and M. Qi, "MapReduce based parallel neural networks in enabling large scale machine learning," *Comput. Intell. Neurosci.*, 2015.

[2] 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, Jan. 2018.

[3] Q. Yan, S. Yang, and M. Wigger, "Storage, computation, and communication: A fundamental tradeoff in distributed computing," arXiv:1806.07565

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

[5] 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, Mar. 2018.

[6] S. Li, M. A. Maddah-Ali, and A. S. Avestimehr, "A unified coding framework for distributed computing with straggling servers," in *Proc. IEEE Globecom Works (GC Wkshps),* Washington, DC, USA, Dec. 2016.

[7] J. Zhang and O. Simeone, "Improved latency-communication trade-off for map-shuffle-reduce systems with stragglers," arXiv:1808.06583

[8] Q. Yu, M. Maddah-Ali, and S. Avestimehr, "Polynomial codes: An optimal design for high-dimensional coded matrix multiplication," in *Proc. The 31st Annual Conf. Neural Inf. Processing System (NIPS)*, Long Beach, CA, USA, May 2017.

[9] Q. Yu, M. Maddah-Ali, and S. Avestimehr, "Straggler mitigation in distributed matrix multiplication: Fundamental limits and optimal coding," in *Proc. IEEE Int. Symp. Inf. Theory,* Vail, CO, USA, pp. 2022–2026, Jun. 2018.

[10] T. Baharav, K. Lee, O. Ocal, and K. Ramchandran, "Straggler-proofing massive-scale distributed matrix multiplication with d-dimensional product codes," in *Proc. IEEE Int. Symp. Inf. Theory,* Vail, CO, USA, pp. 1993–1997, Jun. 2018.

[11] N. Raviv, I. Tamo, R. Tandon, and A. G. Dimakis, "Gradient coding from cyclic MDS codes and expander graphs," arXiv: 1707.03858.

[12] Q. Yan, M. Wigger, S. Yang, and X. Tang, "A fundamental storage-communication tradeoff in distributed computing with stragging nodes," arXiv:1901.07793.

[13] Q. Yan, M. Cheng, X. Tang, and Q. Chen, "On the placement delivery array design for centralized coded caching scheme," *IEEE Trans. Inf. Theory,* vol. 63, no. 9, pp. 5821–5833, Sep. 2017.