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

Qifa Yan, Michèle Wigger, Sheng Yang, Xiaohu Tang

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

Qifa Yan, *Member, IEEE,* Michèle Wigger, *Senior Member, IEEE,*
Sheng Yang, *Member, IEEE,* and Xiaohu Tang *Senior Member, IEEE*

***Abstract*—Placement delivery arrays for distributed computing (Comp-PDAs) have recently been proposed as a framework to construct universal computing schemes for MapReduce-like systems. In this work, we extend this concept to systems with straggling nodes, i.e., to systems where a subset of the nodes cannot accomplish the assigned map computations in due time. Unlike most previous works that focused on computing linear functions, our results are universal and apply for arbitrary map and reduce functions. Our contributions are as follows. Firstly, we show how to construct a universal coded computing scheme for MapReduce-like systems with straggling nodes from any given Comp-PDA. We also characterize the storage and communication loads of the resulting scheme in terms of the Comp-PDA parameters. Then, we prove an information-theoretic converse bound on the storage-communication (SC) tradeoff achieved by universal computing schemes with straggling nodes. We show that the information-theoretic bound matches the performance achieved by the coded computing schemes with straggling nodes corresponding to the Maddah-Ali and Niesen (MAN) PDAs, i.e., to the Comp-PDAs describing Maddah-Ali and Niesen's coded caching scheme. Interestingly, the MAN-PDAs are optimal for any number of straggling nodes. This implies that the map phase of optimal coded computing schemes does not need to be adapted to the number of stragglers in the system. We show that the points that lie exactly on the fundamental SC tradeoff cannot be achieved with Comp-PDAs that require smaller number of files than the MAN-PDAs. This is however possible for some of the points that lie close to the SC tradeoff. For these latter points, the decrease in the requested number of files can be exponential in the number of nodes of the system. We also model the total execution time, and numerically show that the active set size should be chosen to balance the duration of the map phase and the durations of the shuffle and reduce phases.**

***Index Terms*—Distributed computing, storage, communication, straggler, MapReduce, placement delivery array.**

I. Introduction

Distributed computing has emerged as one of the most important paradigms to speed up large-scale data analysis tasks. One of the most popular programming models is MapReduce [2] which has been used to parallelize computations across distributed computing nodes, e.g., for machine learning tools [3], [4]. Consider the task of computing D output functions from N files through K nodes in a MapReduce system. Each output function ϕ_d, $(1 \leq d \leq \mathsf{D})$, can be decomposed into

- N *map functions* $f_{d,1}, \ldots, f_{d,\mathsf{N}}$, each depending on exactly one different file; and
- a *reduce function* h_d that combines the outputs of the N map functions.

Each node k is responsible for computing a subset of $\frac{\mathsf{D}}{\mathsf{K}}$ output functions through three phases. In the first *map phase*, a central server stores a subset of files $\mathcal{M}_k$ at node k, for each $k \in [\mathsf{K}]$. Each node k then computes all the D *intermediate values* (IVAs) $f_{1,n}(w_n), \ldots, f_{\mathsf{D},n}(w_n)$ from each of its stored files $w_n \in \mathcal{M}_k$. In the subsequent *shuffle phase*, it creates a signal from its computed IVAs and sends the signal to all the other nodes. Based on the received exchanged signals and the locally computed IVAs, in the final *reduce phase* it reconstructs all the IVAs pertaining to its own output functions and calculates the desired outputs.

Recently, Li *et al.* proposed a scheme named coded distributed computing (CDC) to reduce the communication load in the shuffling phase [5]. The idea is to create multicast opportunities by duplicating the files and computing the corresponding map functions at different nodes. It is shown that the CDC scheme achieves the fundamental *storage-communication tradeoff*, i.e., it has the lowest communication load for a given storage constraint. This result has been extended in various directions. For example, [6]–[8] account also for the computation load during the map phase; [9] studies the computation resource-allocation problem; [10]–[13] consider wireless (noisy) networks between computation nodes; [14] considers a model where during the shuffle phase each node broadcasts only to a random subset of the nodes.

In this paper, we consider a setup where during the map phase each node takes a random amount of time to compute its desired map functions [15]. In this case, instead of waiting for all the nodes to finish the assigned computations, which can cause an intolerable delay, data shuffling starts as soon as any set of Q nodes, $\mathsf{Q} \in [\mathsf{K}]$, terminate their map procedures. The set of Q nodes that first terminate the map procedure are called *active nodes*, while the remaining $\mathsf{K} - \mathsf{Q}$ nodes are called *straggling nodes* or *stragglers*. The stragglers are not identified prior to the beginning of the map phase and hence the map phase has to be designed without such knowledge.

Q. Yan and M. Wigger are with LTCI, Télécom Paris, IP Paris, 91120 Palaiseau, France. E-mails: qifay2014@163.com, michele.wigger@telecom-paristech.fr.

S. Yang is with L2S, (UMR CNRS 8506), CentraleSupélec-CNRS-Université Paris-Sud, 91192 Gif-sur-Yvette, France. Email: sheng.yang@centralesupelec.fr.

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

Part of this work has been presented in ISIT 2019 [1]. The work of Q. Yan and M. Wigger has been supported by the ERC under grant agreement 715111. The work of X. Tang was supported in part by the National Natural Science Foundation of China under Grant 61941106.

Distributed computing systems with straggling nodes have mainly been studied in the context of a server-worker framework. In this framework, a central server distributes the raw data to the workers like in the above described map phase, but following this map phase the workers directly communicate their intermediate results to the server, which then produces the final outputs. (Thus, under the server-worker framework, all final outputs are calculated at the server and not at the distributed computing nodes as is the case in MapReduce systems.) The described server-worker framework with stragglers was treated, for example, in [15]–[25] with a focus on high-dimensional matrix-by-matrix or matrix-by-vector multiplications and in [26]–[30] with a focus on gradient computing.

Fewer works studied MapReduce systems with straggling nodes (hereafter referred to as straggling systems) which are more relevant for the present article. Specifically, Li *et al.* [31] proposed to incorporate MDS codes into the CDC scheme to cope with straggler nodes. Their construction however works only when the map functions accomplish matrix-by-vector multiplications. Improved constructions were proposed by Zhang *et al.* by choosing the parameters of MDS code and CDC scheme separately in a more flexible way [32], but also their techniques are applicable only for map functions that are matrix-by-matrix multiplications. In many practical applications such as computations in neural networks and machine learning, the map functions are non-linear and can be very complicated with little structure. This motivates us to investigate the MapReduce framework with straggling nodes for general map and reduce functions. In particular, we will present universal coded computing schemes that can be applied to arbitrary straggling systems, irrespective of the specific map and reduce functions. Moreover, we will show the optimality of our schemes among the class of universal schemes that do not rely on special properties of the map and reduce functions.

More specifically, we first propose a systematic construction of universal coded computing schemes for straggling systems from any *placement delivery array for distributed computing (Comp-PDA)* [8]. A placement delivery array (PDA) is an array whose entries are either a special symbol "$*$" or some integer numbers called ordinary symbols. It was introduced in [33] to represent both the placement and the delivery of coded caching schemes with uncoded prefetching in a single array. In particular, the coded caching schemes proposed by Maddah-Ali and Niesen in [34] can be represented as PDAs [33]. The corresponding PDAs will be referred to as MAN-PDAs, and, as we will see, they play a fundamental role also in coded computing with stragglers. PDAs have further been generalized to other coded caching scenarios such as device-to-device models [35], combination networks [36], networks with private demands [37], and medical data sharing problems [38]. Moreover, several different PDA constructions have been proposed in [39]–[43]. In this paper our focus is on a subclass of PDAs, called Comp-PDAs, which were introduced in [8] to design coded computing schemes for MapReduce systems without straggling nodes. In this paper, we show that Comp-PDAs can also be used to construct coded computing schemes for straggling systems, and we express the storage and computation loads of the obtained schemes in terms of the Comp-PDA parameters.

We then proceed to characterize the fundamental storage-communication (SC) tradeoff for straggling systems by showing that the SC tradeoff curve achieved by MAN-PDA based schemes matches a new information-theoretic converse for universal computing schemes with stragglers. That means, our converse bounds the SC tradeoffs achieved by any coded computing schemes that apply to arbitrary map and reduce functions. For special map and reduce functions, e.g., linear functions, it is possible to find tailored coded computing schemes that achieve better SC tradeoffs, than the one implied by our information-theoretic converse, see e.g., [32]. It is worth pointing out that the MAN-PDA based coded computing schemes adopt a fixed storage strategy irrespective of the active set size Q. This implies that the fundamental SC-tradeoff curve remains unchanged even if the active set size Q has not yet been determined during the map phase. The proposed schemes are thus also optimal in a scenario where the active set size Q is unknown during the map phase.

Finally, we study the complexity of optimal (or near-optimal) coded computing schemes. In fact, a major practical limitation of the SC-optimal coded computing schemes based on MAN-PDAs is that they can only be implemented if the number of files N in the system grows exponentially with the number of nodes K. However, as we show in this paper, in most cases, MAN-PDAs achieve their corresponding fundamental SC pairs with smallest possible number of files, i.e., with smallest *file complexity*, among all Comp-PDA based coded computing schemes. This practical limitation is thus not a weakness of the MAN-PDAs, but seems inherent to PDA-based coded computing schemes for stragglers. Interestingly, the problem can be circumvented by slightly backing off from the SC-optimal tradeoff curve. We show that the schemes based on the Comp-PDAs in [33] achieve SC pairs close to the fundamental SC tradeoff curve but with significantly smaller number of files than the optimal MAN-PDAs. More precisely, we fix an integer q, let the number of nodes K be a multiple of q, and the storage load r be such that $\frac{r}{\mathsf{K}} \in \{\frac{1}{q}, \frac{q-1}{q}\}$ holds. We compare the Comp-PDA in [33] and the MAN-PDA for such pairs (K, r) while we let both of them tend to infinity proportionally. This comparison shows that the ratio of the minimum required number of files of the Comp-PDA in [33] and the MAN-PDA vanishes as $O\left(e^{\mathsf{K}(1-\frac{1}{q})\ln\frac{q}{q-1}}\right)$, while the ratio of their communication loads approaches 1.

At last we conduct numerical simulations to gain insights on how to choose the active set size Q in a practical system. In fact, choosing a large Q increases the duration of the map phase because it takes longer until Q nodes have terminated their computations. Increasing Q however also decreases the durations of the shuffle and reduce phases, because they are proportional to the communication load (which decreases with Q) and to the number of output functions computed at each node (which is inversely proportional to Q). In this paper we present a model for the total execution time of a coded computing scheme and numerically find the optimal choice of the active set size Q for this model.

We summarize the contributions of this paper:

1) We establish a general framework for constructing universal coded computing schemes for straggling systems from Comp-PDAs, and evaluate their SC pairs in terms of the Comp-PDA parameters.
2) We derive the fundamental SC tradeoff for any universal straggling system by means of an information theoretic converse, which matches the SC pairs achieved by schemes based on the MAN-PDAs.
3) We show that in most cases, points on the fundamental SC tradeoff curve can be achieved only with the same file complexity as MAN-PDA based schemes. Some points close to the fundamental SC tradeoff curve can however be achieved with significantly smaller file complexities.

The remainder of this paper is organized as follows. Section II formally describes our model, and Section III reviews the definitions of PDAs and Comp-PDAs; Section IV presents the main results of this paper; Section V presents numerical results; Sections VI to VIII present the major proofs of our results; and Section IX concludes this paper.

Notations: For positive integers n, k such that $n \geq k$, we use the notations $[n] \triangleq \{1, 2, \ldots, n\}$, and $[k : n] \triangleq \{k, k+1, \ldots, n\}$. The binomial coefficient is denoted by $C_n^k \triangleq \frac{n!}{k!(n-k)!}$ for $n \geq k \geq 0$; we set $C_n^k = 0$ when $k < 0$ or $k > n$. For $k \leq n$, we use $\mathbf{\Omega}_n^k$ to denote the collection of all subsets of $[n]$ of cardinality k, i.e., $\mathbf{\Omega}_n^k \triangleq \{\mathcal{T} \subseteq [n] : |\mathcal{T}| = k\}$. The binary field is denoted by $\mathbb{F}_2$ and the n dimensional vector space over $\mathbb{F}_2$ is denoted by $\mathbb{F}_2^n$. We use $|\mathcal{A}|$ to denote the cardinality of the set $\mathcal{A}$, while for a signal X, $|X|$ is the number of bits in X. The order of set operations is from left to right. Finally, $\mathbb{1}(\cdot)$ denotes the indicator function that evaluates to 1 if the statement in the parenthesis is true and it evaluates to 0 otherwise.

II. System Model

A (K, Q) *straggling system* is parameterized by the positive integers

$$\mathsf{K}, \mathsf{Q}, \mathsf{N}, \mathsf{D}, \mathsf{U}, \mathsf{V}, \mathsf{W},$$

as described in the following. The system aims to compute D output functions $\phi_1, \ldots, \phi_{\mathsf{D}}$ through K distributed computing nodes from N files. Each output function $\phi_d : \mathbb{F}_2^{\mathsf{NW}} \to \mathbb{F}_2^{\mathsf{U}}$ ($d \in [\mathsf{D}]$) takes as inputs the length-W files in the library $\mathcal{W} = \{w_1 \ldots, w_{\mathsf{N}}\}$, and outputs a bit stream of length U, i.e.,

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

Assume that the computation of the output functions 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 value (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

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

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 during the shuffle phase, one would prefer a decomposition such that the length of the IVAs is as small as possible while still allowing the nodes to compute the final outputs. The computation is carried out through three phases, namely, the map, shuffle, and reduce phases.

1) **Map Phase:** Each node $k \in [\mathsf{K}]$ 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 amount of time to compute its corresponding IVAs. To limit latency of the system, the coded computing scheme proceeds with the shuffle and reduce phases as soon as a fixed number of $\mathsf{Q} \in [\mathsf{K}]$ nodes have terminated the map computations. These nodes are called *active nodes*, and the set of all active nodes is called *active set*, whereas the other $\mathsf{K}-\mathsf{Q}$ nodes are called *straggling nodes*. For simplicity, we consider the symmetric case in which each subset $\mathcal{Q} \subseteq [\mathsf{K}]$ of size $|\mathcal{Q}| = \mathsf{Q}$ is active with same probability. Let the random variable $\mathbf{Q}$ denote the random active set. Then,

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

In our model, we also assume that the map phase has been designed in a way 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 t_n must satisfy

$$t_n \geq \mathsf{K} - \mathsf{Q} + 1, \quad \forall\, n \in [\mathsf{N}]. \tag{2}$$

The output functions $\phi_1, \ldots, \phi_{\mathsf{D}}$ are then uniformly assigned[2] to the nodes in $\mathbf{Q}$. Let $\mathcal{D}_k^{\mathbf{Q}}$ be the set of indices of output functions assigned to a given node $k \in \mathbf{Q}$. Thus, $\mathbf{\Gamma}^{\mathbf{Q}} \triangleq \left\{\mathcal{D}_k^{\mathbf{Q}}\right\}_{k \in \mathbf{Q}}$ forms a partition of $[\mathsf{D}]$, and each set $\mathcal{D}_k^{\mathbf{Q}}$ is of cardinality $\frac{\mathsf{D}}{\mathsf{Q}}$. Denote the set of all the partitions of $[\mathsf{D}]$ into Q equal-sized subsets by $\mathbf{\Delta}$.

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

$$X_k^{\mathbf{Q}} = \varphi_k^{\mathbf{Q}}\left(\mathcal{C}_k, \mathbf{\Gamma}^{\mathbf{Q}}\right)$$

to all the other nodes in $\mathbf{Q}$. For each $k \in \mathbf{Q}$, here $\varphi_k^{\mathbf{Q}} : \mathbb{F}_2^{|\mathcal{C}_k|\mathsf{V}} \times \mathbf{\Delta} \to \mathbb{F}_2^{|X_k^{\mathbf{Q}}|}$ denotes the encoding

[1] In this paper, we thus exclude the "outage" event in which some active sets 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.

function of node k.
We assume a perfect multicast channel, i.e., each active node $k \in \mathbf{Q}$ receives perfectly all the transmitted signals

$$X^{\mathbf{Q}} \triangleq \left\{X_k^{\mathbf{Q}} : k \in \mathbf{Q}\right\}.$$

3) **Reduce Phase:** Using the received signals $X^{\mathbf{Q}}$ from the shuffle phase and the local IVAs $\mathcal{C}_k$ computed in the map phase, node k has to be able to compute all the IVAs

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

where $\psi_k^{\mathbf{Q}} : \mathbb{F}_2^{\sum_{k\in\mathbf{Q}} |X_k^{\mathbf{Q}}|} \times \mathbb{F}_2^{|\mathcal{C}_k|\mathsf{V}} \times \mathbf{\Delta} \to \mathbb{F}_2^{\frac{\mathsf{NDV}}{\mathsf{Q}}}$. Finally, with the restored IVAs, it computes each assigned function via the reduce function, namely,

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

To measure the storage and communication costs, we introduce the following definitions.

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

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

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.,*

$$\overline{L} = \mathbf{E}\left[\frac{\sum_{k\in\mathbf{Q}} |X_k^{\mathbf{Q}}|}{\mathsf{NDV}}\right], \tag{3}$$

where the expectation is taken over the random active set $\mathbf{Q}$*.*

Definition 3 (Storage-Communication (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 load less than* $r + \epsilon$*, a set of uniform assignments of output functions* $\left\{\mathbf{\Gamma}^{\mathcal{Q}}\right\}_{\mathcal{Q}\in\mathbf{\Omega}_{\mathsf{K}}^{\mathsf{Q}}}$*, and a collection of encoding functions* $\left\{\left\{\varphi_k^{\mathcal{Q}}\right\}_{k\in\mathcal{Q}}\right\}_{\mathcal{Q}\in\mathbf{\Omega}_{\mathsf{K}}^{\mathsf{Q}}}$ *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 fundamental storage-communication (SC) tradeoff as*

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

Note that the non-trivial interval for the values of r is $[\mathsf{K} - \mathsf{Q} + 1, \mathsf{K}]$. Indeed, if $r > \mathsf{K}$, then each node can store all the files and compute any function locally. On the other hand, from the assumption (2), we have for any feasible scheme

$$\begin{aligned}\overline{r} &= \frac{\sum_{k=1}^{\mathsf{K}} |\mathcal{M}_k|}{\mathsf{N}} \\ &= \frac{\sum_{n=1}^{\mathsf{N}} t_n}{\mathsf{N}} \\ &\geq \mathsf{K} - \mathsf{Q} + 1.\end{aligned}$$

Therefore, throughout the paper, we only focus on the interval $r \in [\mathsf{K}-\mathsf{Q}+1, \mathsf{K}]$ for any given $\mathsf{Q} \in [\mathsf{K}]$. Further, for a given storage design $\{\mathcal{M}_k\}_{k=1}^{\mathsf{K}}$, by the symmetry assumption of the reduce functions and the fact that each node has all the IVAs of all D output functions in the files it has stored, the optimal communication load is independent of the reduce function assignment. This is similar to the case without straggling nodes (see [5, Remark 3]).

Definition 4 (File Complexity)**.** *The smallest number of files* N *required to implement a given scheme is called the file complexity of this scheme.*

In above problem definition, the various nodes store entire files during the map phase and during the shuffle phase they reconstruct all the IVAs corresponding to their output functions. This system definition does not allow to reduce storage or communication loads by exploiting special structures of the map or reduce functions as proposed for example in [31], [32]. As a consequence, all the coded computing schemes presented in this paper universally apply to arbitrary map and reduce functions and the SC tradeoff in Definition 3 applies only to such universal schemes. In fact, as we will explain, for linear reduce functions the SC tradeoff derived in [32] improves over the one in Definition 3, since it was derived for a system where nodes do not have to store individual files and reconstruct all the required IVAs, but linear combinations thereof suffice.

III. Placement Delivery Arrays for Straggling Systems

A. Definitions

Placement delivery arrays (PDA) introduced in [33] are the main tool of this paper. To adapt to our setup, we use the following definitions.

Definition 5 (Placement Delivery Array (PDA))**.** *For positive integers* $\mathsf{K}, \mathsf{F}, \mathsf{T}$ *and a nonnegative integer* S*, an* $\mathsf{F} \times \mathsf{K}$ *array* $\mathbf{A} = [a_{j,k}]$*,* $j \in [\mathsf{F}], k \in [\mathsf{K}]$*, composed of* T *special symbols "*$*$*" and some ordinary symbols* $1, \ldots, \mathsf{S}$*, each occurring at least once, is called a* $(\mathsf{K}, \mathsf{F}, \mathsf{T}, \mathsf{S})$ *PDA, if for any two distinct entries* a_{j_1,k_1} *and* a_{j_2,k_2}*, we have* $a_{j_1,k_1} = a_{j_2,k_2} = s$ *with* s *an ordinary symbol only if*

a) $j_1 \neq j_2$*,* $k_1 \neq k_2$*, i.e., they lie in distinct rows and distinct columns; and*
b) $a_{j_1,k_2} = a_{j_2,k_1} = *$*,*

i.e., the corresponding 2×2 *subarray formed by rows* j_1, j_2 *and columns* k_1, k_2 *must be of the following form*

$$\begin{bmatrix} s & * \\ * & s \end{bmatrix} \text{ or } \begin{bmatrix} * & s \\ s & * \end{bmatrix}.$$

A PDA with all "$*$*" entries is called trivial. Notice that in this case* $\mathsf{S} = 0$ *and* $\mathsf{KF} = \mathsf{T}$*. A PDA is called a* g*-regular PDA if each ordinary symbol occurs exactly* g *times.*

Example 1. The following array is a 3-regular $(4, 6, 12, 4)$ PDA.

$$\mathbf{A} = \begin{bmatrix} * & * & 1 & 2 \\ * & 1 & * & 3 \\ * & 2 & 3 & * \\ 1 & * & * & 4 \\ 2 & * & 4 & * \\ 3 & 4 & * & * \end{bmatrix}.$$

For our purpose, we introduce the following definitions similarly to the ones in [8].

Definition 6 (PDA for Distributed Computing (Comp-PDA))**.** *A Comp-PDA is a PDA with at least one "$*$"-symbol in each row.*

Definition 7 (Minimum Storage Number)**.** *Given a Comp-PDA* $\mathbf{A}$*, its* minimum storage number τ *is defined as the minimum number of "$*$"-symbols in any of the rows of* $\mathbf{A}$*.*

Definition 8 (Symbol Frequencies)**.** *For a given nontrivial* $(\mathsf{K},\mathsf{F},\mathsf{T},\mathsf{S})$ *Comp-PDA, let* S_t *denote the number of ordinary symbols that occur exactly* t *times, for* $t \in [\mathsf{K}]$*. The symbol frequencies* $\theta_1,\theta_2,\ldots,\theta_{\mathsf{K}}$ *of the Comp-PDA are then defined as*

$$\theta_t \triangleq \frac{\mathsf{S}_t t}{\mathsf{KF}-\mathsf{T}}, \quad t \in [\mathsf{K}].$$

They indicate the fractions of ordinary entries of the Comp-PDA that occur exactly $1,2,\ldots,\mathsf{K}$ *times, respectively. For completeness, we also define* $\theta_t \triangleq 0$ *for* $t > K$*.*

B. Constructing a Coded Computing Scheme from a Comp-PDA: A Toy Example

In this subsection we illustrate the connection between Comp-PDAs and coded computing schemes with stragglers at hand of a toy example. Section VI ahead describes a general procedure to obtain a coded computing scheme with stragglers from any Comp-PDA, and it presents a performance analysis for the obtained scheme.

Consider the $(4,6,12,4)$ Comp-PDA $\mathbf{A}$ in Example 1, and assume a $(\mathsf{K},\mathsf{Q}) = (4,3)$ straggling system with $\mathsf{N} = 6$ files and $\mathsf{D} = 3$ output functions. The scheme is illustrated in Fig. 1 for the case that node 3 is straggling. In this Fig. 1, the line "files" in each of the four boxes indicates the files stored at the nodes. The remaining lines in the boxes illustrate the computed IVAs, where red circles, green triangles, and blue squares depict IVAs pertaining to output functions ϕ_1, ϕ_2, and ϕ_3, respectively. More specifically, a red circle with the number $i \in \{1,2,\ldots,6\}$ in the middle stands for IVA $v_{1,i}$, and so on. The lines below the boxes of the active nodes 1, 2, and 4 indicate the IVAs that the nodes have to learn to compute their output functions. In this example it is assumed that node 1 computes ϕ_1, node 2 computes ϕ_2, and node 4 computes ϕ_3. The signals on the left/right side of the boxes indicate the signals sent by the nodes. Here, splitting of IVAs indicates that the IVA is decomposed into a substring consisting of the first half of the bits and a substring consisting of the second half of the bits, and the plus symbol stands for a bit-wise XOR-operation on the substrings.

We now explain the distributed coding scheme associated with the PDA $\mathbf{A}$. We start by associating column k of $\mathbf{A}$ with node k and row j of $\mathbf{A}$ with file w_j in the system, $(k \in [4], j \in [6])$. In the map phase, node k stores file w_j if the (j,k)-th entry of $\mathbf{A}$ is a "$*$"-symbol. For example, the first column of $\mathbf{A}$ indicates that, node 1, stores files w_1, w_2 and w_3. Each node then computes all the IVAs corresponding to the files it has stored. So node 1 computes $v_{1,1}, v_{1,2}, v_{1,3}, v_{2,1}, v_{2,2}, v_{2,3}, v_{3,1}, v_{3,2}, v_{3,3}$.

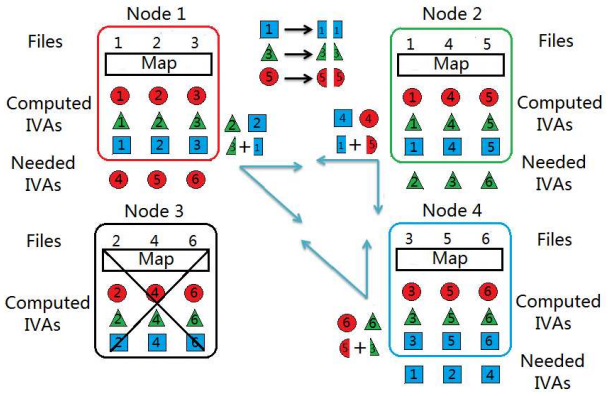

Fig. 1: An example of the CCS scheme for a system with $\mathsf{K} = 4$, $\mathsf{N} = 6$ and $\mathsf{Q} = 3$, where the third node is a straggling node.

Assume that node 3 is the only straggler. Nodes 1, 2, and 4 thus form the active set and continue with the shuffle and reduce procedures. Accordingly, we extract from the PDA $\mathbf{A}$ the subarray $\mathbf{A}^{\{1,2,4\}}$ consisting of columns $1,2$ and 4 (the columns corresponding to the active set):

$$\mathbf{A}^{\{1,2,4\}} = \begin{bmatrix} * & * & 2 \\ * & 1 & 3 \\ * & 2 & * \\ 1 & * & 4 \\ 2 & * & * \\ 3 & 4 & * \end{bmatrix}.$$

Notice that $\mathbf{A}^{\{1,2,4\}}$ is also a Comp-PDA (in particular it has at least one "$*$" symbol in each row) and the node corresponding to a given column has stored all the files indicated by the "$*$"-symbols in this column. After the shuffling phase, we are thus in the same situation as described in [8], [44] when a coded computing scheme without stragglers is to be constructed from a Comp-PDA, and as a consequence, the same shuffle and reduce procedures can be applied. We described these procedures here in detail for completeness.

The shuffle phase is as follows. The output functions ϕ_1,ϕ_2,ϕ_3 are allocated to nodes $1,2,4$ respectively. For each $s \in \{1,2,3,4\}$ occuring g times ($g = 2$ or 3), pick out the $g \times g$ array containing s. For example, symbol $s = 2$ is associated with the following 3-by-3 subarray:

	1	2	4
w_1	$*$	$*$	2
w_3	$*$	2	$*$
w_5	2	$*$	$*$

Each occurence of the symbol "2" in this subarray stands for an IVA desired by the node in the corresponding column and computed at the other nodes in this subarray. The row of the symbol indicates the file this IVA pertains to. The "$*$" symbols in this row indicate that the IVA can indeed be computed by all nodes in this subarray except for the one associated to the column of the "2" symbol. In the above example, the three "2" symbols from top to down represent the IVAs $v_{3,1}$, $v_{2,3}$, and $v_{1,5}$, respectively. These IVAs are shuffled in a coded manner. To this end, they are first split into $g-1 = 2$ equally-large

sub-IVAs, and each of these sub-IVAs is labeled by one of the nodes where the IVA has been computed (i.e., by the columns with "$*$" symbols). The signal sent by a given node i is then simply the componentwise XOR of the sub-IVAs with label i. In our example, we split $v_{3,1} = (v_{3,1}^1, v_{3,1}^2)$, $v_{2,3} = (v_{2,3}^1, v_{2,3}^4)$ and $v_{1,5} = (v_{1,5}^2, v_{1,5}^4)$. So, nodes 1, 2, 4 send $v_{2,3}^1 \oplus v_{3,1}^1$, $v_{3,1}^2 \oplus v_{1,5}^2$ and $v_{1,5}^4 \oplus v_{2,3}^4$, respectively. The same procedure is applied for all other ordinary symbols 1, 3, and 4 in subarray $\mathbf{A}^{\{1,2,4\}}$. The following table lists all the signals sent at the 4 nodes, where the first line lists their associated ordinary symbols:

Symbol	1	2	3	4
Node 1	$v_{2,2}$	$v_{2,3}^1 \oplus v_{3,1}^1$	$v_{3,2}$	
Node 2	$v_{1,4}$	$v_{3,1}^2 \oplus v_{1,5}^2$		$v_{3,4}$
Node 3	(straggling)			
Node 4		$v_{1,5}^4 \oplus v_{2,3}^4$	$v_{1,6}$	$v_{2,6}$

(4)

In the reduce phase, the nodes extract their missing IVAs as follows. Since node 1 has computed $v_{1,1}, v_{1,2}$ and $v_{1,3}$ in the map phase, it still needs to decode $v_{1,4}, v_{1,5}, v_{1,6}$. It directly obtains the IVAs $v_{1,4}$ and $v_{1,6}$ from the uncoded signals sent by nodes 2 and 4 respectively. Moreover, it reconstructs the two sub-IVAs $v_{1,5}^2$ and $v_{1,5}^4$, by XORing the signals $v_{3,1}^2 \oplus v_{1,5}^2$ and $v_{1,5}^4 \oplus v_{2,3}^4$ shuffled by nodes 2 and 4 with its locally stored sub-IVAs $v_{3,1}^2$ and $v_{2,3}^4$. Nodes 2 and 4 reconstruct their missing IVAs in a similar way.

A similar procedure is also applied for any other possible realization of the active set $\mathcal{Q}$ of size $\mathsf{Q} = 3$.

In the above scheme, the total number of IVAs computed at all nodes is $3 \times 4 = 12$, and the storage load is thus $r = \frac{12}{N} = 2$. The total length of the transmitted signals is 7.5V, and remains unchanged, irrespective of the realization of the active set $\mathcal{Q}$ (as long as it is of size $\mathsf{Q} = 3$). The communication load of the system is thus $L = \frac{7.5\mathsf{V}}{6\times 3\times \mathsf{V}} = \frac{5}{12}$.

IV. Main Results

In this section, we present our main results. Details and proofs are deferred to Sections VI–VIII.

A. Coded Computing Schemes for Straggling Systems from Comp-PDAs

In Section VI, we propose a coded computing scheme for a (K, Q) straggling system based on any Comp-PDA with K columns and minimum storage number $\tau \geq \mathsf{K} - \mathsf{Q} + 1$. Theorem 1 is proved by analyzing the coded computing scheme, which is deferred to Section VI-B.

Theorem 1. *From any given $(\mathsf{K}, \mathsf{F}, \mathsf{T}, \mathsf{S})$ Comp-PDA $\mathbf{A}$ with symbol frequencies $\{\theta_t\}_{t=1}^{\mathsf{K}}$ and minimum storage number $\tau \in [\mathsf{K} - \mathsf{Q} + 1 : \mathsf{K}]$, one can construct a coded computing scheme for a (K, Q) straggling system achieving the SC pair*

$$r_{\mathbf{A}} = \frac{\mathsf{T}}{\mathsf{F}},$$
$$L_{\mathbf{A}} = \left(1 - \frac{\mathsf{T}}{\mathsf{FK}}\right) \cdot \frac{1}{C_{\mathsf{K}-1}^{\mathsf{Q}-1}} \cdot \sum_{t=1}^{\mathsf{K}} \theta_t \left(C_{\mathsf{K}-t}^{\mathsf{Q}-1} + \sum_{l=\max\{1,t-\mathsf{K}+\mathsf{Q}-1\}}^{\min\{t,\mathsf{Q}\}-1} \frac{1}{l} C_{t-1}^{l} C_{\mathsf{K}-t}^{\mathsf{Q}-l-1} \right), \quad (5)$$

with file complexity F.

Theorem 1 characterizes the performance of the coded computing scheme obtained from a Comp-PDA as described in Section VI in terms of the Comp-PDA parameters. In the following, we will simply say that a Comp-PDA achieves this performance.

Notice that the file complexity of any Comp-PDA based scheme coincides with the number of rows F of the Comp-PDA. We shall therefore call the parameter F of a Comp-PDA its file complexity.

As we show in the following, Theorem 1 can be simplified for regular Comp-PDAs.

Corollary 1. *From any given g-regular $(\mathsf{K}, \mathsf{F}, \mathsf{T}, \mathsf{S})$ Comp-PDA $\mathbf{A}$, with $g \in [\mathsf{K}]$ and minimum storage number $\tau \in [\mathsf{K} - \mathsf{Q} + 1 : \mathsf{K}]$, one can construct a coded computing scheme for a (K, Q) straggling system achieving the SC pair*

$$r_{\mathbf{A}} = \frac{\mathsf{T}}{\mathsf{F}},$$
$$L_{\mathbf{A}} = \left(1 - \frac{\mathsf{T}}{\mathsf{KF}}\right) \cdot \left(\frac{C_{\mathsf{K}-g}^{\mathsf{Q}-1}}{C_{\mathsf{K}-1}^{\mathsf{Q}-1}} + \sum_{l=\max\{1,g-\mathsf{K}+\mathsf{Q}-1\}}^{\min\{g,\mathsf{Q}\}-1} \frac{1}{l} \cdot \frac{C_{g-1}^{l} \cdot C_{\mathsf{K}-g}^{\mathsf{Q}-l-1}}{C_{\mathsf{K}-1}^{\mathsf{Q}-1}} \right),$$

with file complexity F.

Proof: From Theorem 1, we only need to evaluate $L_{\mathbf{A}}$ when $\mathbf{A}$ is a g-$(\mathsf{K}, \mathsf{F}, \mathsf{T}, \mathsf{S})$ Comp-PDA. In this case, all the S symbols occur g times, i.e.,

$$\theta_g = 1, \quad \text{and} \quad \theta_t = 0, \quad \forall\, t \in [\mathsf{K}] \backslash \{g\}.$$

Then the conclusion directly follows from Theorem 1. ∎

Corollary 1 is of particular interest since there are several explicit regular PDA constructions for coded caching in the literature, such as [33], [42], [43], which are also Comp-PDAs. In particular, the following PDAs obtained from the coded caching scheme proposed by Maddah-Ali and Niesen [34] are important.

Definition 9 (Maddah-Ali Niesen PDA (MAN-PDA))**.** *Fix any integer $i \in [\mathsf{K}]$, and let $\{\mathcal{T}_j\}_{j=1}^{C_{\mathsf{K}}^{i}}$ denote all subsets of $[\mathsf{K}]$ of size i. Also, choose an arbitrary bijective function κ from the collection of all subsets of $[\mathsf{K}]$ with cardinality $i+1$ to the set $\left[C_{\mathsf{K}}^{i+1}\right]$. Then, define the array $\mathbf{P}_i = [p_{j,k}]$ as*

$$p_{j,k} \triangleq \begin{cases} *, & \text{if } k \in \mathcal{T}_j \\ \kappa(\{k\} \cup \mathcal{T}_j), & \text{if } k \notin \mathcal{T}_j \end{cases}.$$

We observe that for any $i \in [\mathsf{K} - 1]$, the array $\mathbf{P}_i$ is an $(i+1)$-regular $\left(\mathsf{K}, C_{\mathsf{K}}^{i}, \mathsf{K}C_{\mathsf{K}-1}^{i-1}, C_{\mathsf{K}}^{i+1}\right)$ Comp-PDA (see [33] for details). For $i = \mathsf{K}$, the Comp-PDA $\mathbf{P}_i$ consists only of "$*$"-entries and is thus a trivial PDA. By Corollary 1, we directly obtain the following result.

Corollary 2. *Consider a (K, Q) straggling system and a positive integer $r \in [\mathsf{K} - \mathsf{Q} + 1 : \mathsf{K}]$. On such a straggling*

system, the MAN-PDA $\mathbf{P}_r$ achieves the storage load r and communication load

$$L_{\mathbf{P}_r} \triangleq \left(1-\frac{r}{\mathsf{K}}\right)\cdot \sum_{l=r+\mathsf{Q}-\mathsf{K}}^{\min\{r,\mathsf{Q}-1\}} \frac{1}{l}\cdot \frac{C_r^l \cdot C_{\mathsf{K}-r-1}^{\mathsf{Q}-l-1}}{C_{\mathsf{K}-1}^{\mathsf{Q}-1}}.$$

The coded computing scheme associated to $\mathbf{P}_r$ is equivalent to our proposed *coded computing for straggling systems (CCS)* in [1]. Here, we present it as a special case of the more general Comp-PDA framework. As we shall see, the Comp-PDA framework allows us to design new coded computing schemes with smaller file complexities.

B. The Fundamental Storage-Communication Tradeoff

We are ready to present our result on the fundamental SC tradeoff, which is proved in Section VII.

Theorem 2. *For a (K,Q) straggling system, with a given integer storage load r in the discrete set $[\mathsf{K}-\mathsf{Q}+1:\mathsf{K}]$, the fundamental SC tradeoff is*

$$L^*_{\mathsf{K},\mathsf{Q}}(r) = \left(1-\frac{r}{\mathsf{K}}\right)\cdot \sum_{l=r+\mathsf{Q}-\mathsf{K}}^{\min\{r,\mathsf{Q}-1\}} \frac{1}{l}\cdot \frac{C_r^l \cdot C_{\mathsf{K}-r-1}^{\mathsf{Q}-l-1}}{C_{\mathsf{K}-1}^{\mathsf{Q}-1}}, \quad (6)$$

which is achievable with a scheme of file complexity C_{K}^r. For a general r in the interval $[\mathsf{K}-\mathsf{Q}+1,\mathsf{K}]$, the fundamental SC tradeoff $L^*_{\mathsf{K},\mathsf{Q}}(r)$ is given by the lower convex envelope formed by the above points in (6).

Fig. 2 shows the fundamental SC tradeoff curves for $\mathsf{K}=10$ and different values of Q. 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 fundamental tradeoff without straggling nodes (cf. [5, Fig. 1]). In this latter case without stragglers, the fundamental SC tradeoff curve is achieved by the CDC scheme in [5]. For a general value of Q and integer storage $r\in[\mathsf{K}-\mathsf{Q}+1:\mathsf{K}]$, the fundamental SC tradeoff pair $(r, L^*_{\mathsf{K},\mathsf{Q}}(r))$ is achieved by the MAN-PDA $\mathbf{P}_r$, see Corollary 2. This implies that for a fixed integer storage load $r\in[1:\mathsf{K}]$, the SC pairs $\{(r,L^*_{\mathsf{K},\mathsf{Q}}(r))\}_{\mathsf{Q}=\mathsf{K}-r+1}^{K}$ are all achieved by the same PDA $\mathbf{P}_r$, irrespective of the size of the active set Q. As we show in Section VI-A, the map procedures of the coded computing scheme corresponding to a given Comp-PDA at a given node k only depends on the "$*$"-symbols in the k-th column of the PDA. Therefore, all the points on the fundamental SC tradeoff curve with same integer storage load r can be attained with the same map procedures described by the MAN-PDA $\mathbf{P}_r$. (See also Remark 3 in Section VI-A.)

As a consequence, the fundamental SC-tradeoff points that have integer storage load $r\in[1,\mathsf{K}]$ remain achievable (and optimal) also in a related setup where the size of the active set Q is unknown during the map procedure. By simple time and memory-sharing arguments, this conclusion extends to all points on the fundamental SC tradeoff curve with arbitrary real-valued storage loads $r\in[1,\mathsf{K}]$. This also relates to the scenario where the system imposes a hard time-limit for the map phase and proceeds to the shuffle and reduce phases with the (random) number of nodes that have terminated within due time. For given storage load r, the MAN-PDA based coded computing scheme promises that when $\mathsf{Q}\geq\lceil\mathsf{K}+1-r\rceil$ nodes have terminated during the map phase, all IVAs are computed at least once and thus the system can proceed to data shuffling, and achieves the minimum required communication load $L^*_{\mathsf{K},\mathsf{Q}}(r)$. When only $\mathsf{Q}<\lceil\mathsf{K}+1-r\rceil$ nodes have terminated, some IVAs are not computed, and hence the system cannot proceed.

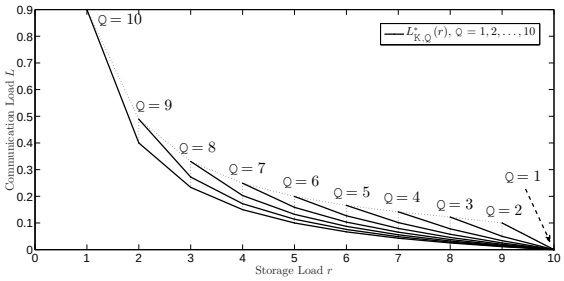

Fig. 2: Storage-communication tradeoff $L^*_{\mathsf{K},\mathsf{Q}}(r)$ for $\mathsf{Q}\in[\mathsf{K}]$ when $\mathsf{K}=10$.

It is further worth pointing out that all our PDA based coded computing schemes are universal and achieve the same performance for any choice of map and reduce functions. No structure is assumed on these functions. Similarly, our information-theoretic converse applies only to such universal coded computing schemes. If the map or reduce functions have certain properties, for example, linearity, it is possible to achieve better SC tradeoffs by storing combinations of files instead of each file separately [31], [32]. Fig. 3 compares Theorem 2 to the results in [31], [32]. It can be observed that the MAN-PDA based scheme outperforms the scheme in [31] but is inferior to the improved version in [32]. As already mentioned, the scheme in [32] however works only for linear map functions, and not for arbitrary functions as our schemes. Another advantage of our schemes is that they work over the binary field, and are thus easier to implement than the MDS-based schemes in [31], [32]. which require a large enough field size.

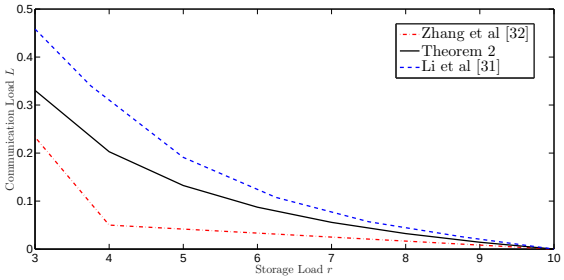

Fig. 3: Comparison with known results when applied to linear map functions, $\mathsf{K}=10,\mathsf{Q}=8$.

C. Optimality and Reduction of File Complexity

From Theorem 1 and Corollary 2, the coded computing scheme based on the MAN-PDA $\mathbf{P}_r$, for $r\in[\mathsf{K}-\mathsf{Q}+1:\mathsf{K}]$, has file complexity $\mathsf{F}=C_{\mathsf{K}}^r$ and achieves the fundamental SC tradeoff. The following theorem indicates that, this is the

smallest file complexity to achieve the same tradeoff point in most cases. The proof is deferred to Section VIII.

Theorem 3. *Consider a (K,Q) straggling system and a Comp-PDA based scheme achieving the fundamental SC tradeoff $\left(r, L^*_{\mathsf{K},\mathsf{Q}}(r)\right)$ for some $r \in [\mathsf{K}-\mathsf{Q}+1:\mathsf{K}]$. If $\mathsf{Q} \notin \{2,\mathsf{K}\}$ or $r \neq \mathsf{K}-\mathsf{Q}+1$, the file complexity of the scheme $\mathsf{F} \geq C^r_{\mathsf{K}}$.*

Remark 1. It is easy to verify that, in the case $\mathsf{Q} \in \{2,\mathsf{K}\}$ and $r = \mathsf{K}-\mathsf{Q}+1$, the fundamental SC tradeoff can be achieved with $\mathsf{F}=1$ with the Comp-PDAs $[*,*,\ldots,*,1]$ and $[*,1,2,\ldots,\mathsf{K}-1]$, respectively.

We next present Comp-PDAs with smaller file complexities F and achieving SC tradeoffs close to the optimal ones. We consider two existing PDA constructions originally proposed for coded caching in [33, Theorems 4 and 5]. Let $q \in [2:\mathsf{K}-1]$ be such that $q \,|\, \mathsf{K}$, and $m = \frac{\mathsf{K}}{q}$. There exists

P1) an m-regular $\left(mq, q^{m-1}, mq^{m-1}, (q-1)q^{m-1}\right)$ Comp-PDA with minimum storage number m;
P2) an $m(q-1)$-regular $\left(mq, (q-1)q^{m-1}, m(q-1)^2q^{m-1}, q^{m-1}\right)$ Comp-PDA with minimum storage number $m(q-1)$.

Corollary 3. *For any integer $r \in [\mathsf{K}-\mathsf{Q}+1:\mathsf{K}-1]$, such that either $r \,|\, \mathsf{K}$ or $(\mathsf{K}-r) \,|\, \mathsf{K}$, the communication load*

$$L_{\mathsf{K},\mathsf{Q}}(r) = \left(1-\frac{r}{\mathsf{K}}\right) \cdot \left(\frac{C^{\mathsf{Q}-1}_{\mathsf{K}-r}}{C^{\mathsf{Q}-1}_{\mathsf{K}-1}} + \sum_{l=\max\{1,r+\mathsf{Q}-\mathsf{K}-1\}}^{\min\{r,\mathsf{Q}\}-1} \frac{1}{l} \cdot \frac{C^l_{r-1} C^{\mathsf{Q}-l-1}_{\mathsf{K}-r}}{C^{\mathsf{Q}-1}_{\mathsf{K}-1}}\right), \quad (7)$$

can be achieved with file complexity $\mathsf{F} = \frac{r}{\mathsf{K}} \cdot \left(\frac{\mathsf{K}}{\min\{r,\mathsf{K}-r\}}\right)^{\min\{r,\mathsf{K}-r\}}$.

Proof: If $r \,|\, \mathsf{K}$, then specialize the Comp-PDA in P1) to parameter $q = \frac{\mathsf{K}}{r}$. This results in a r-regular $\left(\mathsf{K}, q^{r-1}, \mathsf{K}q^{r-2}, (q-1)q^{r-1}\right)$ Comp-PDA with minimum storage number r, and the proof is then immediate from Corollary 1. If $\mathsf{K}-r|\mathsf{K}$, then specialize the Comp-PDA in P2) to parameter $q = \frac{\mathsf{K}}{\mathsf{K}-r}$. This results in a r-regular $\left(\mathsf{K}, (q-1)q^{\mathsf{K}-r-1}, \mathsf{K}(q-1)^2q^{\mathsf{K}-r-2}, q^{\mathsf{K}-r-1}\right)$ Comp-PDA, and the proof again follows from Corollary 1. ∎

In the following proposition, we quantify how close the above SC tradeoff point is to the optimal, and by how much we can reduce the file complexity.

Proposition 1. Consider a (K,Q) straggling system and an integer $r \in [\mathsf{K}-\mathsf{Q}+1:\mathsf{K}]$ such that $\frac{r}{\mathsf{K}} = c \in \left\{\frac{1}{q}, \frac{q-1}{q}\right\}$ for some integer $q \in [2:\mathsf{K}-1]$. There exist $\alpha \in [0,2]$ and $\beta \in [0, \sqrt{2\pi}e^{\frac{1}{6}}]$, such that the SC tradeoff $L_{\mathsf{K},\mathsf{Q}}(r)$ and the file complexity F achieved by constructions P1) or P2) above satisfy

$$\frac{L_{\mathsf{K},\mathsf{Q}}(r)}{L^*_{\mathsf{K},\mathsf{Q}}(r)} = 1 + \frac{\alpha}{r},$$

and

$$\frac{\mathsf{F}}{\mathsf{F}^*} = \beta A_q \mathsf{K}^{\frac{1}{2}} B_q^{-\mathsf{K}},$$

where $A_q \triangleq \frac{\sqrt{q-1}}{cq}$ and $B_q \triangleq \left(\frac{q}{q-1}\right)^{\frac{q-1}{q}}$.

The proof is given in Appendix B. From the above proposition, for a fixed integer q, whenever $\frac{r}{\mathsf{K}} \in \left\{\frac{1}{q}, \frac{q-1}{q}\right\}$ and K, r scale proportionally to infinity, the communication load is close to optimal, while the file complexity can be reduced by a factor that increases exponentially in K.

Remark 2. In this work, we only consider two particular PDAs. There has been extensive research in coded caching schemes with low subpacketization level using various approaches. Most of them have an equivalent PDA representation. For example, PDAs can be constructed from hyper-graphs [42], bipartite graphs [43], linear block codes [45], Ruzsa-Szemerédi graphs [46]. The result in Theorem 1 makes it possible to apply all these results straightforwardly to straggling systems.

V. Numerical Results

The goal of this section is to provide insights on how to choose the active set size Q in a practical system that employs either the (SC-tradeoff optimal) MAN-PDA based schemes or the low-complexity Comp-PDA based schemes of Section IV-C.

In our system, the map-phase computation times at the various nodes are random and independent of each other. The map-phase computation time of node $k \in [\mathsf{K}]$ is denoted T_k and follows a shifted exponential distribution [15]:

$$\Pr\{T_k \leq t\} = 1 - e^{-\mu(t-t_0)}, \quad \forall\, t \geq t_0,\, k \in [\mathsf{K}],$$

where t_0 is the minimum time for node k to accomplish its computation, and $\mu > 0$ is a given delay parameter. The map phase is terminated as soon as a given number Q of nodes have terminated their computations. Thus, the duration of the map phase is given by the Q-th order statistics $T_{(\mathsf{Q})}$ of the tuple $T_1,\ldots,T_{\mathsf{K}}$. By a standard result of order statistics of exponential distributions [47, pp. 18], $T_{(\mathsf{Q})}$ follows the same distribution as the weighted sum $\frac{1}{\mu}\left(\sum_{i=1}^{\mathsf{Q}} \frac{Y_i}{\mathsf{K}-i+1}\right) + t_0$ when $Y_1,\ldots,Y_{\mathsf{Q}}$ are i.i.d. standard exponentially distributed random variables. Consequently:

$$\mathbf{E}[T_{(\mathsf{Q})}] = \frac{1}{\mu}\left(\sum_{i=1}^{\mathsf{Q}} \frac{1}{\mathsf{K}-i+1}\right) + t_0.$$

The total execution time of the distributed computing scheme is given by the sum of the durations of the map, shuffle, and reduce phases. We assume that

(i) the duration of the map phase is $T_{(\mathsf{Q})}$;
(ii) the duration of the shuffle phase is proportional to the communication load, so $\alpha L_{(\mathsf{Q})}(r)$, for some factor $\alpha > 0$ and given storage load r;
(iii) the duration of the reduce phase is proportional to the inverse of Q, so $\frac{\beta}{\mathsf{Q}}$, for some factor $\beta > 0$. This is motivated by the fact that the number of reduce functions that each node has to compute is $\frac{\mathsf{D}}{\mathsf{Q}}$.

For a fixed active set size Q and given r, the random total execution time of the distributed computing scheme is thus:

$$T_D = T_{(\mathsf{Q})} + \alpha \cdot L_{(\mathsf{Q})}(r) + \beta \cdot \frac{1}{\mathsf{Q}}, \quad (8)$$

and the expected running time is:

$$\mathbf{E}[T_D] = \mathbf{E}[T_{(\mathsf{Q})}] + \alpha \cdot \mathbf{E}[L_{(\mathsf{Q})}(r)] + \beta \cdot \frac{1}{\mathsf{Q}}.$$

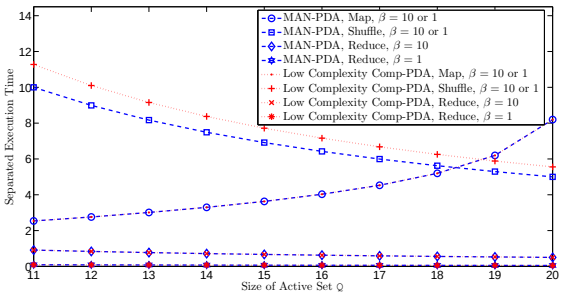

Fig. 4: Expected execution time of the various phases of distributed computing schemes as a function of Q, when $\mathsf{K} = 20, r = 10, t_0 = 1, \mu = 0.5, \alpha = 100$, and $\beta = 10$ or $\beta = 1$.

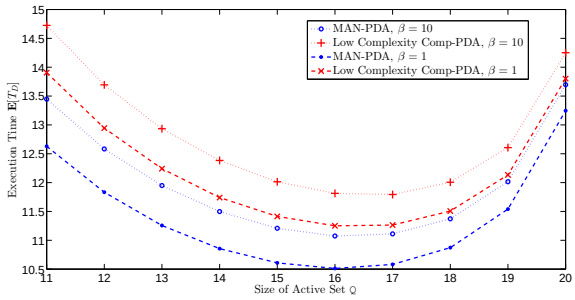

Fig. 5: Expected execution time $\mathbf{E}[T_D]$ as a function of Q, when $\mathsf{K} = 20, r = 10, \mu = 0.5, \alpha = 100$, and $\beta = 10$ or $\beta = 1$.

Notice that since $T_1, \ldots, T_{\mathsf{K}}$ are i.i.d. random variables, each subset of $[\mathsf{K}]$ of size Q is equally likely to be the active set. For a coded computing scheme based on a PDA $\mathbf{A}$, the expected communication load $\mathbf{E}[L_{(\mathsf{Q})}(r)]$ is thus given by (5). More specifically, for the MAN-PDA based schemes the communication load is characterised in (6) and for the low-complexity Comp-PDA based scheme in (7). Notice further that for given factors $\mu, \alpha, \beta > 0$ and parameters $r, \mathsf{K}, \mathsf{N}, \mathsf{D}, t_0$, the expected duration of the map phase, $\mathbf{E}[T_{(\mathsf{Q})}]$, is increasing in Q, whereas the durations of the shuffle and reduce phases, $\alpha \cdot L_{(\mathsf{Q})}(r)$ and $\beta \cdot \frac{1}{\mathsf{Q}}$, are both decreasing in Q. This can also be verified at hand of Fig. 4 which shows the durations of the map, shuffle, and reduce phases of the MAN-PDA based schemes and the low-complexity Comp-PDA schemes for parameters $\mathsf{K} = 20$, $r = 10$, $t_0 = 1$, $\mu = 0.5$, $\alpha = 100$ and $\beta = 10$ or $\beta = 1$, and for different values of the active set size Q. (The parameters have been chosen so that the shuffle phase dominates the other phases. This behaviour has been observed in [5] in their experiments on Amazon EC2 clusters.) The choice of the active set size Q that minimizes the total execution time thus depends on the weights μ, α, β. For example, for parameters $\mathsf{K} = 20$, $r = 10$, $t_0 = 1$, $\mu = 0.5$, $\alpha = 100$ and $\beta = 1$, both for the MAN-PDA schemes and for the low-complexity Comp-PDA scheme, the total execution time is minimized for active set size $\mathsf{Q} = 16$, see Fig. 5.

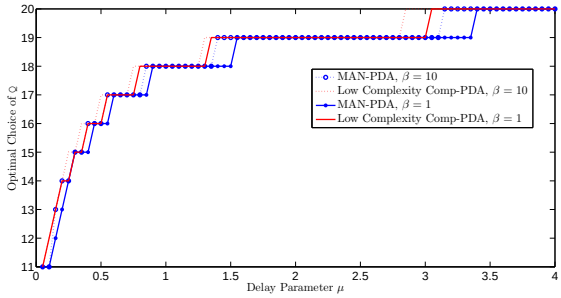

Fig. 6: Optimal choice of Q as a function of the delay parameter μ when $\mathsf{K} = 20, r = 10, \alpha = 100$, and $\beta = 10$ or $\beta = 1$.

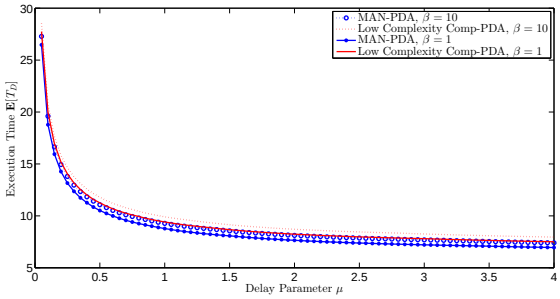

Fig. 7: Expected execution time $\mathbf{E}[T_D]$ as a function of the delay parameter μ under the optimal choice of Q when $\mathsf{K} = 20, r = 10, \alpha = 1, \beta = 10$ or 1.

Fig. 6 shows the optimal choice of Q as a function of the delay parameter μ, where the other parameters are set as described above. We observe that this optimal choice of Q is increasing in μ. The reason is that increasing values of μ imply shorter map-phase computation times at the nodes. In this case it is advantageous to choose the active sets Q large, because it will cause only a small increase in the map-phase but a substantial decrease in the durations of the shuffle and reduce phases.

Fig. 7 depicts the expected execution time under the optimal choice of Q for both the MAN-PDA based scheme and the low-complexity Comp-PDA based scheme as a function of μ, for two values of β. It is observed that the expected execution time of the low-complexity Comp-PDA based schemes of Section IV-C is very close to the expected execution time of the MAN-PDA based schemes. The reason is that the choice of the PDA structure only affects the average communication load $\mathbf{E}[L_{(\mathsf{Q})}]$ and thus only the duration of the shuffle phase, but not the duration of the map and reduce phases. Since the MAN-PDA based schemes and the low-complexity schemes have comparable communication loads, see Proposition 1, according to (8) the two schemes must also have comparable total execution times.

VI. Coded Computing Schemes for Straggling Systems from Comp-PDAs (Proof of Theorem 1)

In this section, we prove Theorem 1 by describing how to construct a coded computing scheme from a given Comp-PDA and analyzing its performance.

A. Constructing a Coded Computing Scheme for a Straggling System from a Comp-PDA

In [8], we described how to obtain a coded computing scheme without stragglers from any given Comp-PDA. A similar procedure is possible in the presence of stragglers if the minimum storage number $\tau \geq \mathsf{K}-\mathsf{Q}+1$. In fact, assume a given Comp-PDA $\mathbf{A}$. The storage design in the map phase corresponding to $\mathbf{A}$ is the same as without straggling nodes. As part of the map phase, each node computes all the IVAs that it can compute from its stored files. For the reduce phase of the straggling system, we restrict to the subarray $\mathbf{A}^{\mathbf{Q}}$ of $\mathbf{A}$ formed by the columns of $\mathbf{A}$ with indices in the active set $\mathbf{Q}$. Notice that $\mathbf{A}^{\mathbf{Q}}$ is again a Comp-PDA, because the minimum storage number of $\mathbf{A}$ is at least $\mathsf{K}-\mathsf{Q}+1$ and after eliminating $\mathsf{K}-\mathsf{Q}$ columns from $\mathbf{A}$ each row still contains at least one "$*$" symbol. Shuffle and reduce phases are performed as in a non-straggling setup, see [8], but where the Comp-PDA $\mathbf{A}$ is replaced by the new Comp-PDA $\mathbf{A}^{\mathbf{Q}}$. For completeness, we explain the map, shuffle, and reduce phases in detail.

Fix a $(\mathsf{K},\mathsf{F},\mathsf{T},\mathsf{S})$ Comp-PDA $\mathbf{A}=[a_{i,j}]$ with minimum storage number $\tau \geq \mathsf{K}-\mathsf{Q}+1$. Partition the N files into F batches $\mathcal{W}_1,\mathcal{W}_2,\ldots,\mathcal{W}_{\mathsf{F}}$, each containing

$$\eta \triangleq \frac{\mathsf{N}}{\mathsf{F}}$$

files and so that $\mathcal{W}_1,\mathcal{W}_2,\ldots,\mathcal{W}_{\mathsf{F}}$ form a partition for $\mathcal{W}$. It is implicitly assumed here that η is an integer number.

1) Map Phase: Each node k stores the files in

$$\mathcal{M}_k = \bigcup_{i\in[\mathsf{F}]\,:\,a_{i,k}=*} \mathcal{W}_i, \tag{9}$$

and computes the IVAs in (1). The map phase terminates whenever any Q nodes accomplish their computations. Throughout this section, let $\mathbf{Q}=\mathcal{Q}$ be the realization of the active set. Then, $\mathbf{A}^{\mathcal{Q}}$ denotes the subarray of $\mathbf{A}$ composed of the columns in $\mathcal{Q}$. Also, let $g_s^{\mathcal{Q}}$ denote the number of occurrences of the symbol s in $\mathbf{A}^{\mathcal{Q}}$, i.e.,

$$g_s^{\mathcal{Q}} = |\{(i,k): a_{i,k}=s, k\in\mathcal{Q}\}|,$$

and $\mathcal{I}^{\mathcal{Q}}$ be the set of symbols occuring only once in $\mathbf{A}^{\mathcal{Q}}$:

$$\mathcal{I}^{\mathcal{Q}} \triangleq \{s\in[S]: g_s^{\mathcal{Q}}=1\}.$$

The symbols in $\mathcal{I}^{\mathcal{Q}}$ are partitioned into Q subsets $\{\mathcal{I}_k^{\mathcal{Q}}: k\in\mathcal{Q}\}$ as follows. For each $s\in\mathcal{I}^{\mathcal{Q}}$, let (i,j) be the unique pair in $[\mathsf{F}]\times\mathcal{Q}$ such that $a_{i,j}=s$. Since the number of "$*$" symbols in the i-th row of $\mathbf{A}$ is equal or larger than $\mathsf{K}-\mathsf{Q}+1$ by the assumption $\tau\geq\mathsf{K}-\mathsf{Q}+1$, there exists at least one $k\in\mathcal{Q}$ such that $a_{i,k}=*$. Arbitrarily choose such a k and assign s into $\mathcal{I}_k^{\mathcal{Q}}$.

Let $\mathcal{A}_k^{\mathcal{Q}}$ denote the set of ordinary symbols in column k occurring at least twice:

$$\mathcal{A}_k^{\mathcal{Q}} \triangleq \{s\in[\mathsf{S}]: a_{i,k}=s \text{ for some } i\in[\mathsf{F}]\}\backslash\mathcal{I}^{\mathcal{Q}},\ k\in\mathcal{Q}. \tag{10}$$

Pick any uniform assignment of reduce functions $\mathbf{D}^{\mathcal{Q}}=\{\mathcal{D}_k^{\mathcal{Q}}\}_{k\in\mathcal{Q}}$. Let $\mathcal{U}_{i,j}^{\mathcal{Q}}$ denote the set of IVAs for node j computed from the files in $\mathcal{W}_i$, i.e.,

$$\mathcal{U}_{i,j}^{\mathcal{Q}} \triangleq \left\{v_{d,n}:\ d\in\mathcal{D}_j^{\mathcal{Q}},\ w_n\in\mathcal{W}_i\right\},\quad (i,j)\in[\mathsf{F}]\times\mathcal{Q}.$$

2) Shuffle Phase: Node k multicasts the signal

$$X_k^{\mathcal{Q}} = \left\{X_{k,s}^{\mathcal{Q}} : s\in\mathcal{I}_k^{\mathcal{Q}}\cup\mathcal{A}_k^{\mathcal{Q}}\right\},$$

where the signals $X_{k,s}^{\mathcal{Q}}$ are created as described in the following, depending on whether $s\in\mathcal{I}_k$ or $s\in\mathcal{A}_k^{\mathcal{Q}}$. For all $s\in\mathcal{I}_k$, set

$$X_{k,s}^{\mathcal{Q}} \triangleq \mathcal{U}_{i,j}^{\mathcal{Q}},\qquad s\in\mathcal{I}_k, \tag{11}$$

where (i,j) is the unique index in $[\mathsf{F}]\times\mathcal{Q}$ such that $a_{i,j}=s$.

To describe the signal $X_{k,s}^{\mathcal{Q}}$ for $s\in\mathcal{A}_k^{\mathcal{Q}}$, we first describe a partition of the IVA $\mathcal{U}_{i,j}^{\mathcal{Q}}$ for each pair $(i,j)\in[\mathsf{F}]\times\mathcal{Q}$ such that $a_{i,j}\in\mathcal{A}_j^{\mathcal{Q}}$. Let $s'=a_{i,j}$, then $g_{s'}^{\mathcal{Q}}\geq 2$. Let $(l_1,j_1),(l_2,j_2),\ldots,(l_{g_{s'}^{\mathcal{Q}}-1},j_{g_{s'}^{\mathcal{Q}}-1})\in[\mathsf{F}]\times\mathcal{Q}$ indicate all the other $g_{s'}^{\mathcal{Q}}-1$ occurrences of the ordinary symbol s' in $\mathbf{A}^{\mathcal{Q}}$, i.e.,

$$a_{l_1,j_1}=a_{l_2,j_2}=\ldots=a_{l_{g_{s'}^{\mathcal{Q}}-1},j_{g_{s'}^{\mathcal{Q}}-1}}=s'.$$

Partition the set of IVAs $\mathcal{U}_{i,j}^{\mathcal{Q}}$ into $g_{s'}^{\mathcal{Q}}-1$ subsets of equal size and denote these subsets by $\mathcal{U}_{i,j}^{\mathcal{Q},j_1},\mathcal{U}_{i,j}^{\mathcal{Q},j_2},\ldots,\mathcal{U}_{i,j}^{\mathcal{Q},j_{g_{s'}^{\mathcal{Q}}-1}}$:

$$\mathcal{U}_{i,j}^{\mathcal{Q}} = \left\{\mathcal{U}_{i,j}^{\mathcal{Q},j_1},\mathcal{U}_{i,j}^{\mathcal{Q},j_2},\ldots,\mathcal{U}_{i,j}^{\mathcal{Q},j_{g_{s'}^{\mathcal{Q}}-1}}\right\}. \tag{12}$$

For all $s\in\mathcal{A}_k^{\mathcal{Q}}$, set

$$X_{k,s}^{\mathcal{Q}} \triangleq \bigoplus_{\substack{(i,j)\in[\mathsf{F}]\times(\mathcal{Q}\backslash\{k\}):\\ a_{i,j}=s}} \mathcal{U}_{i,j}^{\mathcal{Q},k},\qquad s\in\mathcal{A}_k^{\mathcal{Q}}. \tag{13}$$

3) Reduce Phase: Node k computes all IVAs in

$$\bigcup_{i\in[\mathsf{F}]}\mathcal{U}_{i,k}^{\mathcal{Q}}.$$

In the map phase, node k has already computed all IVAs in $\{\mathcal{U}_{i,k}^{\mathcal{Q}}: a_{i,k}=*\}$. It thus remains to compute all IVAs in

$$\bigcup_{i\in[\mathsf{F}]\,:\,a_{i,k}\neq *}\mathcal{U}_{i,k}^{\mathcal{Q}}.$$

Fix an arbitrary $i\in[\mathsf{F}]$ such that $a_{i,k}\neq *$, and set $s=a_{i,k}$. If $s\in\mathcal{A}_k^{\mathcal{Q}}$, each subblock $\mathcal{U}_{i,k}^{\mathcal{Q},j}$ in (12) can be restored by node k from the signal $X_{j,s}^{\mathcal{Q}}$ sent by node j (see (13)):

$$\mathcal{U}_{i,k}^{\mathcal{Q},j} = \mathcal{U}_{l_1,j_1}^{\mathcal{Q},j}\ \oplus\ \ldots\ \oplus\ \mathcal{U}_{l_{g_s^{\mathcal{Q}}-2},j_{g_s^{\mathcal{Q}}-2}}^{\mathcal{Q},j}\ \oplus X_{j,s}^{\mathcal{Q}}, \tag{14}$$

where (l_t,j_t) $(t\in[g_s^{\mathcal{Q}}-2])$ indicate the other $g_s^{\mathcal{Q}}-2$ occurrences of the symbol s in $\mathbf{A}^{\mathbf{Q}}$, i.e., $a_{l_t,j_t}=s$. Notice that the sub-IVAs on the right-hand side of (14) have been computed by node k during the map phase, because by the PDA properties, $a_{l_t,j_t}=a_{i,k}=s$ and $j_t\neq k$ imply that $l_t\neq i$ and $a_{l_t,k}=*$. Therefore, $\mathcal{U}_{i,k}^{\mathcal{Q},j}$ can be decoded from (14).

If $s\notin\mathcal{A}_k^{\mathcal{Q}}$, then $s\in\mathcal{I}^{\mathcal{Q}}$ by (10). There exists thus an index $j\in\mathcal{Q}\backslash\{k\}$ such that $s\in\mathcal{I}_j$ and therefore, by (11), the subset $\mathcal{U}_{i,k}^{\mathcal{Q}}$ can be recovered from the signal $X_{j,s}^{\mathcal{Q}}$ sent by node j.

Remark 3. It is worth pointing out that the storage design $\{\mathcal{M}_k\}_{k=1}^{\mathsf{K}}$ only depends on the positions of the "$*$" symbols in $\mathbf{A}$, but not on the parameter Q (See (9)). This indicates

that, in practice the map phase can be carried out even without knowing how many nodes will be participating in the shuffle and reduce phases.

B. Performance Analysis

We have analyzed the performances of storage and communication loads in the no-stragglers setup in [8]. For the scheme in the preceding subsection, the analysis of storage load follows the same lines as in [8]. When computing the communication load defined in (3), we have to average over all realizations of the active set $\mathbf{Q}$.

1) Storage Load: Since the Comp-PDA $\mathbf{A}$ contains T "$*$" symbols, and each "$*$" symbol indicates that a batch of $\eta = \frac{\mathsf{N}}{\mathsf{F}}$ files is stored at a given node, see (9), the storage load of the proposed scheme is:

$$\bar{r} = \frac{\sum_{k=1}^{\mathsf{K}} |\mathcal{M}_k|}{\mathsf{N}} = \frac{\mathsf{T} \cdot \eta}{\mathsf{N}} = \frac{\mathsf{T}}{\mathsf{F}}.$$

2) Communication Load: We first analyze the length of the signals sent for a given realization of the active set $\mathbf{Q} = \mathcal{Q}$. For any $s \in [\mathsf{S}]$, let g_s be the occurrence of s in $\mathbf{A}$, and $g_s^{\mathcal{Q}}$ be the occurrence of s in the columns in $\mathcal{Q}$. By (11) and (13), the length of the signals associated to symbol s is

$$l_s^{\mathcal{Q}} = \begin{cases} 0, & \text{if } g_s^{\mathcal{Q}} = 0 \\ \frac{\mathsf{VND}}{\mathsf{FQ}}, & \text{if } g_s^{\mathcal{Q}} = 1 \\ \frac{g_s^{\mathcal{Q}}}{g_s^{\mathcal{Q}}-1} \cdot \frac{\mathsf{VND}}{\mathsf{FQ}}, & \text{if } g_s^{\mathcal{Q}} \geq 2 \end{cases}, \tag{15}$$

when $\mathcal{Q}$ is the active set. The total length of all the signals is thus

$$\begin{aligned} \sum_{k \in \mathcal{Q}} |X_k^{\mathcal{Q}}| &= \sum_{k \in \mathcal{Q}} \sum_{s \in [\mathsf{S}]: g_s^{\mathcal{Q}} > 0} |X_{k,s}^{\mathcal{Q}}| \\ &= \sum_{s \in [\mathsf{S}]: g_s^{\mathcal{Q}} > 0} \sum_{k \in \mathcal{Q}} |X_{k,s}^{\mathcal{Q}}| \\ &= \sum_{s \in [\mathsf{S}]} l_s^{\mathcal{Q}}. \end{aligned} \tag{16}$$

We now compute the communication load as defined in (3) where we have to average over all realizations of the active set $\mathbf{Q}$:

$$\begin{aligned} L_{\mathbf{A}} &= \mathbf{E}\left[\frac{\sum_{k \in \mathbf{Q}} |X_k^{\mathbf{Q}}|}{\mathsf{NDV}}\right] \\ &= \frac{1}{\mathsf{NDV}} \cdot \frac{1}{|\mathbf{\Omega}_{\mathsf{K}}^{\mathsf{Q}}|} \cdot \sum_{\mathcal{Q} \in \mathbf{\Omega}_{\mathsf{K}}^{\mathsf{Q}}} \sum_{k \in \mathcal{Q}} |X_k^{\mathcal{Q}}| \\ &\overset{(a)}{=} \frac{1}{\mathsf{NDV}} \cdot \frac{1}{C_{\mathsf{K}}^{\mathsf{Q}}} \cdot \sum_{\mathcal{Q} \in \mathbf{\Omega}_{\mathsf{K}}^{\mathsf{Q}}} \sum_{s \in [\mathsf{S}]} l_s^{\mathcal{Q}} \\ &\overset{(b)}{=} \frac{1}{\mathsf{NDV}} \cdot \frac{1}{C_{\mathsf{K}}^{\mathsf{Q}}} \\ &\quad \cdot \sum_{s \in [\mathsf{S}]} \sum_{\mathcal{Q} \in \mathbf{\Omega}_{\mathsf{K}}^{\mathsf{Q}}} l_s^{\mathcal{Q}} \cdot \left(\sum_{l=0}^{\mathsf{Q}} \mathbb{1}(g_s^{\mathcal{Q}} = l)\right) \cdot \left(\sum_{g=1}^{\mathsf{K}} \mathbb{1}(g_s = g)\right) \\ &= \frac{1}{\mathsf{NDV}} \cdot \frac{1}{C_{\mathsf{K}}^{\mathsf{Q}}} \cdot \sum_{g=1}^{\mathsf{K}} \sum_{s \in [\mathsf{S}]} \sum_{l=0}^{\mathsf{Q}} \sum_{\mathcal{Q} \in \mathbf{\Omega}_{\mathsf{K}}^{\mathsf{Q}}} l_s^{\mathcal{Q}} \cdot \mathbb{1}(g_s^{\mathcal{Q}} = l) \cdot \mathbb{1}(g_s = g) \\ &\overset{(c)}{=} \frac{1}{\mathsf{NDV}} \cdot \frac{1}{C_{\mathsf{K}}^{\mathsf{Q}}} \cdot \sum_{g=1}^{\mathsf{K}} \sum_{s \in [\mathsf{S}]} \left(\sum_{\mathcal{Q} \in \mathbf{\Omega}_{\mathsf{K}}^{\mathsf{Q}}} \frac{\mathsf{NDV}}{\mathsf{FQ}} \cdot \mathbb{1}(g_s^{\mathcal{Q}} = 1) \right. \\ &\quad \left. + \sum_{l=2}^{\mathsf{Q}} \sum_{\mathcal{Q} \in \mathbf{\Omega}_{\mathsf{K}}^{\mathsf{Q}}} \frac{l\mathsf{NDV}}{(l-1)\mathsf{FQ}} \cdot \mathbb{1}(g_s^{\mathcal{Q}} = l) \right) \cdot \mathbb{1}(g_s = g) \\ &= \frac{1}{\mathsf{FQ}} \cdot \frac{1}{C_{\mathsf{K}}^{\mathsf{Q}}} \cdot \sum_{g=1}^{\mathsf{K}} \sum_{s \in [\mathsf{S}]} \left[\sum_{\mathcal{Q} \in \mathbf{\Omega}_{\mathsf{K}}^{\mathsf{Q}}} \mathbb{1}(g_s^{\mathcal{Q}} = 1) \right. \\ &\quad \left. + \sum_{l=2}^{\mathsf{Q}} \frac{l}{l-1} \cdot \left(\sum_{\mathcal{Q} \in \mathbf{\Omega}_{\mathsf{K}}^{\mathsf{Q}}} \mathbb{1}(g_s^{\mathcal{Q}} = l) \right) \right] \cdot \mathbb{1}(g_s = g) \\ &\overset{(d)}{=} \frac{1}{\mathsf{FK}} \cdot \frac{1}{C_{\mathsf{K}-1}^{\mathsf{Q}-1}} \\ &\quad \cdot \sum_{g=1}^{\mathsf{K}} \sum_{s \in [\mathsf{S}]} \left(C_g^1 C_{\mathsf{K}-g}^{\mathsf{Q}-1} + \sum_{l=2}^{\mathsf{Q}} \frac{l}{l-1} C_g^l C_{\mathsf{K}-g}^{\mathsf{Q}-l} \right) \mathbb{1}(g_s = g) \\ &= \frac{1}{\mathsf{FK}} \cdot \frac{1}{C_{\mathsf{K}-1}^{\mathsf{Q}-1}} \\ &\quad \cdot \sum_{g=1}^{\mathsf{K}} \left(C_g^1 C_{\mathsf{K}-g}^{\mathsf{Q}-1} + \sum_{l=2}^{\mathsf{Q}} \frac{l}{l-1} C_g^l C_{\mathsf{K}-g}^{\mathsf{Q}-l} \right) \sum_{s \in [\mathsf{S}]} \mathbb{1}(g_s = g) \\ &\overset{(e)}{=} \frac{1}{\mathsf{FK}} \cdot \frac{1}{C_{\mathsf{K}-1}^{\mathsf{Q}-1}} \cdot \sum_{g=1}^{\mathsf{K}} \mathsf{S}_g \left(C_g^1 C_{\mathsf{K}-g}^{Q-1} + \sum_{l=2}^{\mathsf{Q}} \frac{l}{l-1} \cdot C_g^l C_{\mathsf{K}-g}^{\mathsf{Q}-l} \right) \\ &\overset{(f)}{=} \frac{1}{\mathsf{FK}} \cdot \frac{1}{C_{\mathsf{K}-1}^{\mathsf{Q}-1}} \cdot \sum_{g=1}^{\mathsf{K}} \mathsf{S}_g \left(g \cdot C_{\mathsf{K}-g}^{\mathsf{Q}-1} + \sum_{l=2}^{\mathsf{Q}} \frac{g}{l-1} \cdot C_{g-1}^{l-1} C_{\mathsf{K}-g}^{\mathsf{Q}-l} \right) \\ &\overset{(g)}{=} \frac{1}{\mathsf{FK}} \cdot \frac{1}{C_{\mathsf{K}-1}^{\mathsf{Q}-1}} \cdot \sum_{g=1}^{\mathsf{K}} \mathsf{S}_g g \left(C_{\mathsf{K}-g}^{\mathsf{Q}-1} + \sum_{l=1}^{\mathsf{Q}-1} \frac{1}{l} \cdot C_{g-1}^l C_{\mathsf{K}-g}^{\mathsf{Q}-l-1} \right) \\ &\overset{(h)}{=} \frac{1}{\mathsf{FK}} \cdot \frac{1}{C_{\mathsf{K}-1}^{\mathsf{Q}-1}} \\ &\quad \cdot \sum_{g=1}^{\mathsf{K}} \mathsf{S}_g g \left(C_{\mathsf{K}-g}^{\mathsf{Q}-1} + \sum_{l=\max\{1,g-\mathsf{K}+\mathsf{Q}-1\}}^{\min\{g,\mathsf{Q}\}-1} \frac{1}{l} \cdot C_{g-1}^l C_{\mathsf{K}-g}^{\mathsf{Q}-l-1} \right) \\ &= \frac{\mathsf{FK}-\mathsf{T}}{\mathsf{FK}} \cdot \frac{1}{C_{\mathsf{K}-1}^{\mathsf{Q}-1}} \cdot \sum_{g=1}^{\mathsf{K}} \frac{\mathsf{S}_g g}{\mathsf{FK}-\mathsf{T}} \\ &\quad \cdot \left(C_{\mathsf{K}-g}^{\mathsf{Q}-1} + \sum_{l=\max\{1,g-\mathsf{K}+\mathsf{Q}-1\}}^{\min\{g,\mathsf{Q}\}-1} \frac{1}{l} \cdot C_{g-1}^l C_{\mathsf{K}-g}^{\mathsf{Q}-l-1} \right) \\ &\overset{(i)}{=} \left(1 - \frac{\mathsf{T}}{\mathsf{FK}}\right) \cdot \frac{1}{C_{\mathsf{K}-1}^{\mathsf{Q}-1}} \\ &\quad \cdot \sum_{g=1}^{\mathsf{K}} \theta_g \left(C_{\mathsf{K}-g}^{\mathsf{Q}-1} + \sum_{l=\max\{1,g-\mathsf{K}+\mathsf{Q}-1\}}^{\min\{g,\mathsf{Q}\}-1} \frac{1}{l} \cdot C_{g-1}^l C_{\mathsf{K}-g}^{\mathsf{Q}-l-1} \right), \end{aligned}$$

where (a) holds by (16); (b) holds since for each $s \in [\mathsf{S}]$, $\sum_{l=0}^{\mathsf{Q}} \mathbb{1}(g_s^{\mathcal{Q}} = l) = 1$ and $\sum_{g=1}^{\mathsf{K}} \mathbb{1}(g_s = g) = 1$; (c) follows from (15); and (d) holds because if a symbol s occurrs exactly g times in a PDA $\mathbf{A}$, then there are $C_g^l \cdot C_{\mathsf{K}-g}^{\mathsf{Q}-l}$ different $F \times \mathsf{Q}$ submatrices $\mathbf{A}^{\mathcal{Q}}$ of $\mathbf{A}$ in which s occurrs exactly l times;

in (e), we defined S_g to be the number of ordinary symbols occurring g times for each $g \in [\mathsf{K}]$; in (f), we used the equality $C_g^l = \frac{g}{l} \cdot C_{g-1}^{l-1}$; in (h), we eliminated the indices of zero terms in the summation of (g); and (i) follows from the definition of symbol frequencies.

C. File Complexity of the Proposed Schemes

The analysis of file complexity is similar to the no-straggler setup in [8]. The files are partitioned into F batches so that each batch contains $\eta = \frac{\mathsf{N}}{\mathsf{F}} > 0$ files. It is assumed that η is a positive integer. The smallest number of files N where this assumption can be met is F. Therefore, the file complexity of the scheme is F.

VII. The Fundamental Storage-Communication Tradeoff (Proof of Theorem 2)

By Corollary 2, the SC pair $\big(r, L^*_{\mathsf{K},\mathsf{Q}}(r)\big)$, $r \in [\mathsf{K}-\mathsf{Q}+1 : \mathsf{K}]$ can be achieved by the MAN-PDA $\mathbf{P}_r$. For a general non-integer $r \in [\mathsf{K}-\mathsf{Q}+1, \mathsf{K}]$, the lower convex envelope of these points can be achieved by memory- and time-sharing. It remains to prove the converse in Theorem 2.

Let $Z^{\mathsf{Q}}_{\mathsf{K}}(x)$ be a piecewise linear function connecting the points $\big(u, Z^{\mathsf{Q}}_{\mathsf{K}}(u)\big)$ sequentially over the interval $[\mathsf{K}-\mathsf{Q}+1, \mathsf{K}]$ with

$$Z^{\mathsf{Q}}_{\mathsf{K}}(u) \triangleq \sum_{l=u+\mathsf{Q}-\mathsf{K}}^{\min\{u,\mathsf{Q}\}} \frac{\mathsf{Q}-l}{\mathsf{Q}l} C_u^l C_{\mathsf{K}-u}^{\mathsf{Q}-l}, \ u \in [\mathsf{K}-\mathsf{Q}+1 : \mathsf{K}]. \quad (17)$$

We shall need the following lemma, proved in Appendix A.

Lemma 1. *The sequence $Z^{\mathsf{Q}}_{\mathsf{K}}(u)$ is strictly convex and decreasing for $u \in [\mathsf{K}-\mathsf{Q}+1 : \mathsf{K}]$. And the function $Z^{\mathsf{Q}}_{\mathsf{K}}(x)$ is convex and decreasing over $[\mathsf{K}-\mathsf{Q}+1, \mathsf{K}]$.*

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

$$a_{\mathcal{M},u} \triangleq \sum_{\mathcal{I}\subseteq\mathcal{K}:|\mathcal{I}|=u} \left| \left(\bigcap_{k\in\mathcal{I}} \mathcal{M}_k \right) \setminus \left(\bigcup_{\bar{k}\in\mathcal{K}\setminus\mathcal{I}} \mathcal{M}_{\bar{k}} \right) \right|, \quad (18)$$

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

$$\begin{aligned} a_{\mathcal{M},u} &\geq 0, \\ \sum_{u=\mathsf{K}-\mathsf{Q}+1}^{\mathsf{K}} a_{\mathcal{M},u} &= \mathsf{N}, \\ \sum_{u=\mathsf{K}-\mathsf{Q}+1}^{\mathsf{K}} u a_{\mathcal{M},u} &= \overline{r}\mathsf{N}. \end{aligned} \quad (19)$$

For any $\mathcal{Q} \in \mathbf{\Omega}^{\mathsf{Q}}_{\mathsf{K}}$ and any $l \in [\mathsf{Q}]$, define

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

i.e., $b^{\mathcal{Q}}_{\mathcal{M},l}$ is the number of files stored exactly l times in the nodes of set $\mathcal{Q}$. Since any file that is stored u times across all the nodes has l occurrences in exactly $C_u^l \cdot C_{\mathsf{K}-u}^{\mathsf{Q}-l}$ subsets $\mathcal{Q}$ of size Q, i.e.,

$$\begin{aligned} &\sum_{\mathcal{Q}\in\mathbf{\Omega}^{\mathsf{Q}}_{\mathsf{K}}} \mathbb{1}(w_n \text{ is stored at exactly } l \text{ nodes of } \mathcal{Q}) \\ &= \sum_{u=\max\{l,\mathsf{K}-\mathsf{Q}+1\}}^{\mathsf{K}-\mathsf{Q}+l} \mathbb{1}(w_n \text{ is stored at exactly } u \text{ nodes of } \mathcal{K}) \\ &\quad \cdot C_u^l \cdot C_{\mathsf{K}-u}^{\mathsf{Q}-l}, \qquad \forall\, n \in [\mathsf{N}]. \end{aligned}$$

Summing over $n \in [\mathsf{N}]$, we obtain

$$\sum_{\mathcal{Q}\in\mathbf{\Omega}^{\mathsf{Q}}_{\mathsf{K}}} b^{\mathcal{Q}}_{\mathcal{M},l} = \sum_{u=\max\{l,\mathsf{K}-\mathsf{Q}+1\}}^{\mathsf{K}-\mathsf{Q}+l} a_{\mathcal{M},u} \cdot C_u^l \cdot C_{\mathsf{K}-u}^{\mathsf{Q}-l}. \quad (20)$$

Now we use the result in [5, Lemma 1] to lower bound the communication load for any realization of the active set $\mathbf{Q} = \mathcal{Q}$:

$$\frac{\sum_{k\in\mathcal{Q}} |X_k^{\mathcal{Q}}|}{\mathsf{NDV}} \geq \sum_{l=1}^{\mathsf{Q}} \frac{b^{\mathcal{Q}}_{\mathcal{M},l}}{\mathsf{N}} \frac{\mathsf{Q}-l}{\mathsf{Q}l}.$$

The average communication load over the random realization of the active set $\mathbf{Q}$ is then obtained as:

$$\begin{aligned} \overline{L} &= \mathbf{E}_{\mathbf{Q}} \left[\frac{\sum_{k\in\mathbf{Q}} |X_k^{\mathbf{Q}}|}{\mathsf{NDV}} \right] \\ &= \sum_{\mathcal{Q}\in\mathbf{\Omega}^{\mathsf{Q}}_{\mathsf{K}}} \frac{\sum_{k\in\mathcal{Q}} |X_k^{\mathcal{Q}}|}{\mathsf{NDV}} \cdot \Pr\{\mathbf{Q} = \mathcal{Q}\} \\ &\geq \frac{1}{C^{\mathsf{Q}}_{\mathsf{K}}} \sum_{\mathcal{Q}\in\mathbf{\Omega}^{\mathsf{Q}}_{\mathsf{K}}} \sum_{l=1}^{\mathsf{Q}} \frac{b^{\mathcal{Q}}_{\mathcal{M},l}}{\mathsf{N}} \frac{\mathsf{Q}-l}{\mathsf{Q}l} \\ &= \frac{1}{C^{\mathsf{Q}}_{\mathsf{K}}} \sum_{l=1}^{\mathsf{Q}} \left(\sum_{\mathcal{Q}\in\mathbf{\Omega}^{\mathsf{Q}}_{\mathsf{K}}} \frac{b^{\mathcal{Q}}_{\mathcal{M},l}}{\mathsf{N}} \right) \frac{\mathsf{Q}-l}{\mathsf{Q}l} \\ &\overset{(a)}{=} \frac{1}{C^{\mathsf{Q}}_{\mathsf{K}}} \sum_{l=1}^{\mathsf{Q}} \left(\sum_{u=\max\{l,\mathsf{K}-\mathsf{Q}+1\}}^{\mathsf{K}-\mathsf{Q}+l} \frac{a_{\mathcal{M},u}}{\mathsf{N}} C_u^l C_{\mathsf{K}-u}^{\mathsf{Q}-l} \right) \frac{\mathsf{Q}-l}{\mathsf{Q}l} \\ &\overset{(b)}{=} \frac{1}{C^{\mathsf{Q}}_{\mathsf{K}}} \sum_{u=\mathsf{K}-\mathsf{Q}+1}^{\mathsf{K}} \frac{a_{\mathcal{M},u}}{\mathsf{N}} \sum_{l=u+\mathsf{Q}-\mathsf{K}}^{\min\{u,\mathsf{Q}\}} C_u^l C_{\mathsf{K}-u}^{\mathsf{Q}-l} \frac{\mathsf{Q}-l}{\mathsf{Q}l} \\ &\overset{(c)}{=} \frac{1}{C^{\mathsf{Q}}_{\mathsf{K}}} \sum_{u=\mathsf{K}-\mathsf{Q}+1}^{\mathsf{K}} \frac{a_{\mathcal{M},u}}{\mathsf{N}} \cdot Z^{\mathsf{Q}}_{\mathsf{K}}(u) \\ &\overset{(d)}{\geq} \frac{1}{C^{\mathsf{Q}}_{\mathsf{K}}} \cdot Z^{\mathsf{Q}}_{\mathsf{K}} \left(\sum_{u=\mathsf{K}-\mathsf{Q}+1}^{\mathsf{K}} \frac{u a_{\mathcal{M},u}}{\mathsf{N}} \right) \\ &\overset{(e)}{=} \frac{Z^{\mathsf{Q}}_{\mathsf{K}}(\overline{r})}{C^{\mathsf{Q}}_{\mathsf{K}}} \\ &\overset{(f)}{\geq} \frac{Z^{\mathsf{Q}}_{\mathsf{K}}(r+\epsilon)}{C^{\mathsf{Q}}_{\mathsf{K}}}, \end{aligned} \quad (21)$$

where (a) follows from (20); (b) holds because the inner summation in (a) only includes summation indices $u \in [\mathsf{K}-\mathsf{Q} : \mathsf{K}]$ and it includes the summation index $u \in \{\mathsf{K}-\mathsf{Q}+1, \ldots, \mathsf{K}\}$ if, and only if, the outer summation index l satisfies $l \leq u$

and $l \geq u+\mathsf{Q}-\mathsf{K}$; (c) follows from (17); (d) follows from Lemma 1; (e) follows from (19); and (f) follows from the fact $\overline{r} \leq r+\epsilon$. Since ϵ can be arbitrarily close to zero, we conclude

$$L \geq \frac{Z_{\mathsf{K}}^{\mathsf{Q}}(r)}{C_{\mathsf{K}}^{\mathsf{Q}}}.$$

In particular, when $r \in [\mathsf{K}-\mathsf{Q}+1:\mathsf{K}]$, by (17),

$$\begin{aligned}
L &\geq \sum_{l=r+\mathsf{Q}-\mathsf{K}}^{\min\{r,\mathsf{Q}\}} \frac{\mathsf{Q}-l}{\mathsf{Q}l}\frac{C_r^l C_{\mathsf{K}-r}^{\mathsf{Q}-l}}{C_{\mathsf{K}}^{\mathsf{Q}}} \\
&\overset{(a)}{=} \sum_{l=r+\mathsf{Q}-\mathsf{K}}^{\min\{r,\mathsf{Q}-1\}} \frac{\mathsf{Q}-l}{\mathsf{Q}l}\frac{C_r^l C_{\mathsf{K}-r}^{\mathsf{Q}-l}}{C_{\mathsf{K}}^{\mathsf{Q}}} \\
&= \sum_{l=r+\mathsf{Q}-\mathsf{K}}^{\min\{r,\mathsf{Q}-1\}} \frac{\mathsf{Q}-l}{\mathsf{Q}l}\cdot\frac{\frac{\mathsf{Q}!}{l!(\mathsf{Q}-l)!}\cdot\frac{(\mathsf{K}-\mathsf{Q})!}{(r-l)!(\mathsf{K}-\mathsf{Q}-r+l)!}}{\frac{\mathsf{K}!}{r!(\mathsf{K}-r)!}} \\
&= \left(1-\frac{r}{\mathsf{K}}\right)\cdot\sum_{l=r+\mathsf{Q}-\mathsf{K}}^{\min\{r,\mathsf{Q}-1\}} \frac{1}{l}\cdot\frac{\frac{r!}{l!(r-l)!}\cdot\frac{(\mathsf{K}-r-1)!}{(\mathsf{Q}-l-1)!(\mathsf{K}-r-\mathsf{Q}+l)!}}{\frac{(\mathsf{K}-1)!}{(\mathsf{Q}-1)!(\mathsf{K}-\mathsf{Q})!}} \\
&\overset{(b)}{=} \left(1-\frac{r}{\mathsf{K}}\right)\sum_{l=r+\mathsf{Q}-\mathsf{K}}^{\min\{r,\mathsf{Q}-1\}} \frac{1}{l}\frac{C_r^l C_{\mathsf{K}-r-1}^{\mathsf{Q}-l-1}}{C_{\mathsf{K}-1}^{\mathsf{Q}-1}},
\end{aligned}$$

where (a) holds since for $l=\mathsf{Q}$, the term in the summation is zero. This establishes the desired converse proof.

VIII. Optimality of File Complexity (Proof of Theorem 3)

In order to prove Theorem 3, we need to first derive several lemmas.

A. Preliminaries

Lemma 2. *If a coded computing scheme achieves the fundamental SC tradeoff pair $\left(r, L_{\mathsf{K},\mathsf{Q}}^*(r)\right)$ for any integer $r \in [\mathsf{K}-\mathsf{Q}+1:\mathsf{K}]$, then each file is stored exactly r times across the nodes.*

Proof: According to Lemma 1, the sequence $\left\{Z_{\mathsf{K}}^{\mathsf{Q}}(u)\right\}_{u=\mathsf{K}-\mathsf{Q}+1}^{\mathsf{K}}$ is strictly convex. Thus for the integer $r=\sum_{u=\mathsf{K}-\mathsf{Q}+1}^{\mathsf{K}}\frac{u a_{\mathcal{M},u}}{\mathsf{N}}$, the equality in (21) holds if, and only if,

$$\begin{aligned}
&\frac{a_{\mathcal{M},r}}{\mathsf{N}} = 1, \\
&\frac{a_{\mathcal{M},u}}{\mathsf{N}} = 0, \quad u \in [\mathsf{K}-\mathsf{Q}+1:\mathsf{K}]\backslash\{r\}.
\end{aligned}$$

Therefore, by definition of $a_{\mathcal{M},u}$ in (18), this indicates that each file is stored exactly r times across the system. ∎

Lemma 3. *Consider a g-regular $(\mathsf{K},\mathsf{F},\mathsf{T},\mathsf{S})$ PDA where $\mathsf{K} \geq g \geq 2$. If there are exactly $g-1$ "$*$"s in each row, then $\mathsf{F} \geq C_{\mathsf{K}}^{g-1}$.*

Proof. With Definition 5 (the definition of PDAs), the conclusion follows directly from [33, Lemma 2]. ∎

For each $u \in [\mathsf{K}]$, define

$$U_{\mathsf{K}}^{\mathsf{Q}}(u) \triangleq C_{\mathsf{K}-u}^{\mathsf{Q}-1} + \sum_{l=\max\{1,u-\mathsf{K}+\mathsf{Q}-1\}}^{\min\{u,\mathsf{Q}\}-1} \frac{C_{u-1}^l \cdot C_{\mathsf{K}-u}^{\mathsf{Q}-l-1}}{l}. \quad (22)$$

Lemma 4. *When $\mathsf{Q} \geq 3$, the subsequence $\{U_{\mathsf{K}}^{\mathsf{Q}}(u)\}_{u=2}^{\mathsf{K}}$ strictly decreases with $u \in [2:\mathsf{K}]$.*

Proof: For each $u \in [2:\mathsf{K}-1]$, by (22),

$$\begin{aligned}
&U_{\mathsf{K}}^{\mathsf{Q}}(u+1) - U_{\mathsf{K}}^{\mathsf{Q}}(u) \\
&= -C_{\mathsf{K}-u-1}^{\mathsf{Q}-2} + \sum_{l=\max\{1,u-\mathsf{K}+\mathsf{Q}\}}^{\min\{u,\mathsf{Q}-1\}} \frac{C_u^l \cdot C_{\mathsf{K}-u-1}^{\mathsf{Q}-l-1}}{l} \\
&\quad - \sum_{l=\max\{1,u-\mathsf{K}+\mathsf{Q}-1\}}^{\min\{u,\mathsf{Q}\}-1} \frac{C_{u-1}^l \cdot C_{\mathsf{K}-u}^{\mathsf{Q}-l-1}}{l} \\
&\overset{(a)}{=} -C_{\mathsf{K}-u-1}^{\mathsf{Q}-2} + \sum_{l=\max\{1,u-\mathsf{K}+\mathsf{Q}\}}^{\min\{u,\mathsf{Q}-1\}} \frac{(C_{u-1}^l + C_{u-1}^{l-1}) \cdot C_{\mathsf{K}-u-1}^{\mathsf{Q}-l-1}}{l} \\
&\quad - \sum_{l=\max\{1,u-\mathsf{K}+\mathsf{Q}-1\}}^{\min\{u,\mathsf{Q}\}-1} \frac{C_{u-1}^l \cdot (C_{\mathsf{K}-u-1}^{\mathsf{Q}-l-1} + C_{\mathsf{K}-u-1}^{\mathsf{Q}-l-2})}{l} \\
&\overset{(b)}{=} -C_{\mathsf{K}-u-1}^{\mathsf{Q}-2} + \sum_{l=\max\{1,u-\mathsf{K}+\mathsf{Q}\}}^{\min\{u,\mathsf{Q}\}-1} \frac{C_{u-1}^l \cdot C_{\mathsf{K}-u-1}^{\mathsf{Q}-l-1}}{l} \\
&\quad + \sum_{l=\max\{1,u-\mathsf{K}+\mathsf{Q}\}}^{\min\{u,\mathsf{Q}-1\}} \frac{C_{u-1}^{l-1} \cdot C_{\mathsf{K}-u-1}^{\mathsf{Q}-l-1}}{l} \\
&\quad - \sum_{l=\max\{1,u-\mathsf{K}+\mathsf{Q}\}}^{\min\{u,\mathsf{Q}\}-1} \frac{C_{u-1}^l \cdot C_{\mathsf{K}-u-1}^{\mathsf{Q}-l-1}}{l} \\
&\quad - \sum_{l=\max\{1,u-\mathsf{K}+\mathsf{Q}-1\}}^{\min\{u-1,\mathsf{Q}-2\}} \frac{C_{u-1}^l \cdot C_{\mathsf{K}-u-1}^{\mathsf{Q}-l-2}}{l} \\
&= -C_{\mathsf{K}-u-1}^{\mathsf{Q}-2} + \sum_{l=\max\{1,u-\mathsf{K}+\mathsf{Q}\}}^{\min\{u,\mathsf{Q}-1\}} \frac{C_{u-1}^{l-1} \cdot C_{\mathsf{K}-u-1}^{\mathsf{Q}-l-1}}{l} \\
&\quad - \sum_{l=\max\{1,u-\mathsf{K}+\mathsf{Q}-1\}}^{\min\{u-1,\mathsf{Q}-2\}} \frac{C_{u-1}^l \cdot C_{\mathsf{K}-u-1}^{\mathsf{Q}-l-2}}{l} \\
&\overset{(c)}{=} -C_{\mathsf{K}-u-1}^{\mathsf{Q}-2} + \sum_{l=\max\{1,u-\mathsf{K}+\mathsf{Q}\}}^{\min\{u,\mathsf{Q}-1\}} \frac{C_{u-1}^{l-1} \cdot C_{\mathsf{K}-u-1}^{\mathsf{Q}-l-1}}{l} \\
&\quad - \sum_{l=\max\{2,u-\mathsf{K}+\mathsf{Q}\}}^{\min\{u,\mathsf{Q}-1\}} \frac{C_{u-1}^{l-1} \cdot C_{\mathsf{K}-u-1}^{\mathsf{Q}-l-1}}{l-1} \\
&= -\sum_{l=\max\{2,u-\mathsf{K}+\mathsf{Q}\}}^{\min\{u,\mathsf{Q}-1\}} \frac{C_{u-1}^{l-1} \cdot C_{\mathsf{K}-u-1}^{\mathsf{Q}-l-1}}{l(l-1)} \\
&\leq 0,
\end{aligned} \quad (23)$$

where in (a), we used (29); in (b), we separated the two summations in (a) and eliminated the indices of zero terms in the separated summations; and in (c), we used the variable change $l'=l+1$. Moreover, if $u \geq 2$ and $\mathsf{Q} \geq 3$, from (23), $U_{\mathsf{K}}^{\mathsf{Q}}(u+1) - U_{\mathsf{K}}^{\mathsf{Q}}(u) \leq -\frac{C_{u-1}^l C_{\mathsf{K}-u-1}^{\mathsf{Q}-2}}{2} < 0$, i.e., $U_{\mathsf{K}}^{\mathsf{Q}}(u)$

is strictly decreasing when $u \geq 2$. ■

B. Proof of Theorem 3

Define the set

$$\mathcal{E} \triangleq \{(\mathsf{Q}, r) : \mathsf{Q} \in [\mathsf{K}], r \in [\mathsf{K}-\mathsf{Q}+1 : \mathsf{K}]\},$$

and partition it into three subsets

$$\mathcal{E}_1 \triangleq \{(\mathsf{Q}, r) : \mathsf{Q} \in [\mathsf{K}], r = \mathsf{K}\},$$
$$\mathcal{E}_2 \triangleq \{(\mathsf{Q}, r) : \mathsf{Q} \in \{2, \mathsf{K}\}, r = \mathsf{K}-\mathsf{Q}+1\},$$
$$\mathcal{E}_3 \triangleq \{(\mathsf{Q}, r) : \mathsf{Q} \in [3 : \mathsf{K}], r \in [\max\{\mathsf{K}-\mathsf{Q}+1, 2\} : \mathsf{K}-1]\}.$$

Notice that, if $(\mathsf{Q}, r) \in \mathcal{E}_1$, the bound $\mathsf{F} \geq C_{\mathsf{K}}^{\mathsf{K}} = 1$ is trivial. The case $(\mathsf{Q}, r) \in \mathcal{E}_2$ is excluded. Therefore, in the rest of the proof, we assume $(\mathsf{Q}, r) \in \mathcal{E}_3$, i.e., $\mathsf{Q} \in [3 : \mathsf{K}]$ and $r \in [\max\{\mathsf{K}-\mathsf{Q}+1, 2\} : \mathsf{K}-1]$.

Let $\mathbf{A}$ be a $(\mathsf{K}, \mathsf{F}, \mathsf{T}, \mathsf{S})$ Comp-PDA that achieves the optimal tradeoff point $(r, L^*_{\mathsf{K},\mathsf{Q}}(r))$. Recall that each row in a Comp-PDA is associated to a file batch, and a "$*$" symbol in that row and column k indicates that the file batch is stored at node k. According to Lemma 2, each file is stored exactly r times across the nodes, i.e., there are exactly r "$*$" symbols in each row of $\mathbf{A}$.

Let $\theta_{g'}$ be the fraction of ordinary entries occurring g' times in the Comp-PDA $\mathbf{A}$, for all $g' \in [\mathsf{K}]$. Since there are r "$*$" symbols in each row, from the PDA properties a) and b) in Definition 5, any ordinary symbol cannot appear more than $r+1$ times, i.e.,

$$\theta_{g'} = 0, \quad \forall\, g' \in [r+2 : \mathsf{K}]. \tag{24}$$

Therefore,

$$\sum_{g'=1}^{r+1} \theta_{g'} = 1. \tag{25}$$

From (5), (22), and (24), the communication load of $\mathbf{A}$ has the form

$$\begin{aligned} L_{\mathbf{A}} &= \left(1-\frac{r}{\mathsf{K}}\right) \cdot \frac{1}{C_{\mathsf{K}-1}^{\mathsf{Q}-1}} \cdot \sum_{g'=1}^{r+1} \theta_{g'} \cdot U_{\mathsf{K}}^{\mathsf{Q}}(g') \\ &\overset{(a)}{\geq} \left(1-\frac{r}{\mathsf{K}}\right) \cdot \frac{1}{C_{\mathsf{K}-1}^{\mathsf{Q}-1}} \cdot \left(\sum_{g'=1}^{r+1} \theta_{g'}\right) \cdot U_{\mathsf{K}}^{\mathsf{Q}}(r+1) \\ &\overset{(b)}{=} \left(1-\frac{r}{\mathsf{K}}\right) \cdot \frac{1}{C_{\mathsf{K}-1}^{\mathsf{Q}-1}} \cdot U_{\mathsf{K}}^{\mathsf{Q}}(r+1) \\ &\overset{(c)}{=} L^*_{\mathsf{K},\mathsf{Q}}(r), \end{aligned} \tag{26}$$

where (a) follows since by Lemma 4 the sequence $\{U_{\mathsf{K}}^{\mathsf{Q}}(u)\}_{u=2}^{\mathsf{K}}$ is decreasing and because $U_{\mathsf{K}}^{\mathsf{Q}}(1) = U_{\mathsf{K}}^{\mathsf{Q}}(2) = C_{\mathsf{K}-1}^{\mathsf{Q}-1}$; (b) follows from (25); and (c) follows from Theorem 2 and (22). By our assumption $L_{\mathbf{A}} = L^*_{\mathsf{K},\mathsf{Q}}(r)$, the equality in (26) must hold. Since $r+1 \geq 3$ and the sequence $\{U_{\mathsf{K}}^{\mathsf{Q}}(u)\}_{u=2}^{\mathsf{K}}$ strictly decreases, equality in (26) implies that

$$\theta_{g'} = 0, \quad \forall\, g' \in [r]. \tag{27}$$

Combining (24) and (27), we conclude that $\mathbf{A}$ is a $(r+1)$-regular PDA, and each row has exactly r "$*$" symbols. Applying Lemma 3, we complete the proof.

IX. Conclusion

In this work, we have explained how to convert any Comp-PDA with at least $\mathsf{K}-\mathsf{Q}+1$ "$*$" symbols in each row into a coded computing scheme for a MapReduce system with Q non-straggling nodes. We have further characterized the optimal storage-communication (SC) tradeoff for this system. The Comp-PDA framework allows us to design universal coded computing schemes with small file complexities compared to the ones (the MAN-PDAs) achieving the fundamental SC tradeoff. We further model the total execution time of coded computing systems and numerically find the choice of the active set size Q that reduces the total execution time in this model. It is obtained by balancing the duration of the map phase with the durations of the shuffle and reduce phases.

In our setup, for a given integer storage load r, the size of active set Q has to be no less than $\mathsf{K}-r+1$, since we exclude outage events (See Footnote 1). With a given Comp-PDA, the key to obtaining a coded computing scheme for a given active set is that the subarray formed by the columns corresponding to the active set is still a Comp-PDA. In fact, for the constructions in P1) and P2), it allows to construct coded computing schemes for some (but not all) active sets if the active set size Q satisfies $\lceil \frac{\mathsf{K}}{r} \rceil \leq \mathsf{Q} \leq \mathsf{K}-\mathsf{r}$.

Appendix A
Proof of Lemma 1

We shall prove the first statement of the lemma that the sequence $\left\{Z_{\mathsf{K}}^{\mathsf{Q}}(u)\right\}_{u=\mathsf{K}-\mathsf{Q}+1}^{\mathsf{K}}$ is strictly convex and decreasing, i.e.,

$$\begin{aligned} &Z_{\mathsf{K}}^{\mathsf{Q}}(u+1) - Z_{\mathsf{K}}^{\mathsf{Q}}(u) < 0, \quad \forall\, u \in [\mathsf{K}-\mathsf{Q}+1 : \mathsf{K}-1], \\ &Z_{\mathsf{K}}^{\mathsf{Q}}(u+1) - Z_{\mathsf{K}}^{\mathsf{Q}}(u) > Z_{\mathsf{K}}^{\mathsf{Q}}(u) - Z_{\mathsf{K}}^{\mathsf{Q}}(u-1), \\ &\qquad\qquad \forall\, u \in [\mathsf{K}-\mathsf{Q}+2 : \mathsf{K}-1]. \end{aligned}$$

The second statement of the lemma on the piecewise linear function is an immediate consequence of the first one.

By (17),

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

where in (a), we used the identity $\sum_{l=s+\mathsf{Q}-\mathsf{K}}^{\min\{u,\mathsf{Q}\}} C_u^l \cdot C_{\mathsf{K}-u}^{\mathsf{Q}-l} = C_{\mathsf{K}}^{\mathsf{Q}}$. Then,

$$\begin{aligned} &Z_{\mathsf{K}}^{\mathsf{Q}}(u+1) - Z_{\mathsf{K}}^{\mathsf{Q}}(u) \\ &= \sum_{l=u+1+\mathsf{Q}-\mathsf{K}}^{\min\{u+1,\mathsf{Q}\}} \frac{C_{u+1}^l C_{\mathsf{K}-u-1}^{\mathsf{Q}-l}}{l} - \sum_{l=u+\mathsf{Q}-\mathsf{K}}^{\min\{u,\mathsf{Q}\}} \frac{C_u^l C_{\mathsf{K}-u}^{\mathsf{Q}-l}}{l} \\ &\overset{(a)}{=} \sum_{l=u+1+\mathsf{Q}-\mathsf{K}}^{\min\{u+1,\mathsf{Q}\}} \frac{\left(C_u^l + C_u^{l-1}\right) \cdot C_{\mathsf{K}-u-1}^{\mathsf{Q}-l}}{l} \end{aligned}$$

$$
\begin{aligned}
&- \sum_{l=u+\mathsf{Q}-\mathsf{K}}^{\min\{u,\mathsf{Q}\}} \frac{C_u^l \cdot \left(C_{\mathsf{K}-u-1}^{\mathsf{Q}-l} + C_{\mathsf{K}-u-1}^{\mathsf{Q}-l-1}\right)}{l} \\
&\overset{(b)}{=} \sum_{l=\mathsf{Q}+u+1-\mathsf{K}}^{\min\{u,\mathsf{Q}\}} \frac{C_u^l C_{\mathsf{K}-u-1}^{\mathsf{Q}-l}}{l} + \sum_{l=\mathsf{Q}+u+1-\mathsf{K}}^{\min\{u+1,\mathsf{Q}\}} \frac{C_u^{l-1} C_{\mathsf{K}-u-1}^{\mathsf{Q}-l}}{l} \\
&\quad - \sum_{l=\mathsf{Q}+u+1-\mathsf{K}}^{\min\{u,\mathsf{Q}\}} \frac{C_u^l C_{\mathsf{K}-u-1}^{\mathsf{Q}-l}}{l} - \sum_{l=\mathsf{Q}+u-\mathsf{K}}^{\min\{u,\mathsf{Q}-1\}} \frac{C_u^l C_{\mathsf{K}-u-1}^{\mathsf{Q}-l-1}}{r} \\
&= \sum_{l=u+1+\mathsf{Q}-\mathsf{K}}^{\min\{u+1,\mathsf{Q}\}} \frac{C_u^{l-1} C_{\mathsf{K}-u-1}^{\mathsf{Q}-l}}{l} - \sum_{l=u+\mathsf{Q}-\mathsf{K}}^{\min\{u,\mathsf{Q}-1\}} \frac{C_u^l C_{\mathsf{K}-u-1}^{\mathsf{Q}-l-1}}{l} \\
&\overset{(c)}{=} \sum_{l=u+\mathsf{Q}-\mathsf{K}}^{\min\{u,\mathsf{Q}-1\}} \frac{C_u^l C_{\mathsf{K}-u-1}^{\mathsf{Q}-l-1}}{l+1} - \sum_{l=u+\mathsf{Q}-\mathsf{K}}^{\min\{u,\mathsf{Q}-1\}} \frac{C_u^l C_{\mathsf{K}-u-1}^{\mathsf{Q}-l-1}}{l} \\
&= - \sum_{l=u+\mathsf{Q}-\mathsf{K}}^{\min\{u,\mathsf{Q}-1\}} \frac{C_u^l C_{\mathsf{K}-u-1}^{\mathsf{Q}-l-1}}{l(l+1)} \qquad (28) \\
&< 0,
\end{aligned}
$$

where in (a), we applied the identity

$$C_{n+1}^{m+1} = C_n^{m+1} + C_n^m; \qquad (29)$$

in (b), we separated the two summations of (a) into four summations and eliminated indices of zero terms in the separated summations; and in (c), we used the change of variable $l' = l - 1$ in the first summation. Finally, from (28), for $u \in [\mathsf{K}-\mathsf{Q}+2 : \mathsf{K}-1]$, we have

$$
\begin{aligned}
&\left(Z_{\mathsf{K}}^{\mathsf{Q}}(u+1) - Z_{\mathsf{K}}^{\mathsf{Q}}(u)\right) - \left(Z_{\mathsf{K}}^{\mathsf{Q}}(u) - Z_{\mathsf{K}}^{\mathsf{Q}}(u-1)\right) \\
&\quad = \sum_{l=\mathsf{Q}+u-1-\mathsf{K}}^{\min\{u,\mathsf{Q}\}-1} \frac{C_{u-1}^l C_{\mathsf{K}-u}^{\mathsf{Q}-l-1}}{l(l+1)} - \sum_{l=\mathsf{Q}+u-\mathsf{K}}^{\min\{u,\mathsf{Q}-1\}} \frac{C_u^l C_{\mathsf{K}-u-1}^{\mathsf{Q}-l-1}}{l(l+1)} \\
&\quad \overset{(a)}{=} \sum_{l=\mathsf{Q}+u-1-\mathsf{K}}^{\min\{u,\mathsf{Q}\}-1} \frac{C_{u-1}^l \cdot \left(C_{\mathsf{K}-u-1}^{\mathsf{Q}-l-1} + C_{\mathsf{K}-u-1}^{\mathsf{Q}-l-2}\right)}{l(l+1)} \\
&\qquad - \sum_{l=\mathsf{Q}+u-\mathsf{K}}^{\min\{u,\mathsf{Q}-1\}} \frac{\left(C_{u-1}^l + C_{u-1}^{l-1}\right) \cdot C_{\mathsf{K}-u-1}^{\mathsf{Q}-l-1}}{l(l+1)} \\
&\quad \overset{(b)}{=} \sum_{l=\mathsf{Q}+u-\mathsf{K}}^{\min\{u,\mathsf{Q}\}-1} \frac{C_{u-1}^l C_{\mathsf{K}-u-1}^{\mathsf{Q}-l-1}}{l(l+1)} + \sum_{l=\mathsf{Q}+u-1-\mathsf{K}}^{\min\{u-1,\mathsf{Q}-2\}} \frac{C_{u-1}^l C_{\mathsf{K}-u-1}^{\mathsf{Q}-l-2}}{l(l+1)} \\
&\qquad - \sum_{l=\mathsf{Q}+u-\mathsf{K}}^{\min\{u,\mathsf{Q}\}-1} \frac{C_{u-1}^l C_{\mathsf{K}-u-1}^{\mathsf{Q}-l-1}}{l(l+1)} - \sum_{l=\mathsf{Q}+u-\mathsf{K}}^{\min\{u,\mathsf{Q}-1\}} \frac{C_{u-1}^{l-1} C_{\mathsf{K}-u-1}^{\mathsf{Q}-l-1}}{l(l+1)} \\
&\quad = \sum_{l=\mathsf{Q}+u-1-\mathsf{K}}^{\min\{u-1,\mathsf{Q}-2\}} \frac{C_{u-1}^l C_{\mathsf{K}-u-1}^{\mathsf{Q}-l-2}}{l(l+1)} - \sum_{l=\mathsf{Q}+u-\mathsf{K}}^{\min\{u,\mathsf{Q}-1\}} \frac{C_{u-1}^{l-1} C_{\mathsf{K}-u-1}^{\mathsf{Q}-l-1}}{l(l+1)} \\
&\quad \overset{(c)}{=} \sum_{l=\mathsf{Q}+u-\mathsf{K}}^{\min\{u,\mathsf{Q}-1\}} \frac{C_{u-1}^{l-1} C_{\mathsf{K}-u-1}^{\mathsf{Q}-l-1}}{(l-1)l} - \sum_{l=\mathsf{Q}+u-\mathsf{K}}^{\min\{u,\mathsf{Q}-1\}} \frac{C_{u-1}^{l-1} C_{\mathsf{K}-u-1}^{\mathsf{Q}-l-1}}{l(l+1)} \\
&\quad = \sum_{l=u+\mathsf{Q}-\mathsf{K}}^{\min\{u,\mathsf{Q}-1\}} \frac{2C_{u-1}^{l-1} C_{\mathsf{K}-u-1}^{\mathsf{Q}-l-1}}{(l-1)l(l+1)} \\
&\quad > 0,
\end{aligned}
$$

where in (a) we applied the identity (29); in (b), we separated the two summations in (a) and eliminated the indices of zero terms in the separated summations; and in (c), we used the change of variable $l' = l + 1$.

Appendix B
Proof of Proposition 1

By (6), (7) and (22), we have

$$
\begin{aligned}
L_{\mathsf{K},\mathsf{Q}}(r) &= \left(1 - \frac{r}{\mathsf{K}}\right) \cdot \frac{1}{C_{\mathsf{K}-1}^{\mathsf{Q}-1}} \cdot U_{\mathsf{K}}^{\mathsf{Q}}(r), \\
L_{\mathsf{K},\mathsf{Q}}^*(r) &= \left(1 - \frac{r}{\mathsf{K}}\right) \cdot \frac{1}{C_{\mathsf{K}-1}^{\mathsf{Q}-1}} \cdot U_{\mathsf{K}}^{\mathsf{Q}}(r+1).
\end{aligned}
$$

Combining these equalities with (23), we obtain

$$
\begin{aligned}
&L_{\mathsf{K},\mathsf{Q}}(r) - L_{\mathsf{K},\mathsf{Q}}^*(r) \\
&\quad = -\left(1 - \frac{r}{\mathsf{K}}\right) \cdot \frac{1}{C_{\mathsf{K}-1}^{\mathsf{Q}-1}} \cdot \left(U_{\mathsf{K}}^{\mathsf{Q}}(r+1) - U_{\mathsf{K}}^{\mathsf{Q}}(r)\right) \\
&\quad = \left(1 - \frac{r}{\mathsf{K}}\right) \cdot \frac{1}{C_{\mathsf{K}-1}^{\mathsf{Q}-1}} \cdot \sum_{l=\max\{2,r+\mathsf{Q}-\mathsf{K}\}}^{\min\{r,\mathsf{Q}-1\}} \frac{C_{r-1}^{l-1} C_{\mathsf{K}-r-1}^{\mathsf{Q}-l-1}}{l(l-1)} \\
&\quad \overset{(a)}{=} \left(1 - \frac{r}{\mathsf{K}}\right) \cdot \frac{1}{C_{\mathsf{K}-1}^{\mathsf{Q}-1}} \cdot \frac{1}{r} \cdot \sum_{l=\max\{2,r+\mathsf{Q}-\mathsf{K}\}}^{\min\{r,\mathsf{Q}-1\}} \frac{C_r^l C_{\mathsf{K}-r-1}^{\mathsf{Q}-l-1}}{l-1}, \qquad (30)
\end{aligned}
$$

where in (a), we used the identity $C_{r-1}^{l-1} = \frac{l}{r} \cdot C_r^l$. Therefore, with (6) and (30),

$$
\begin{aligned}
&\frac{L_{\mathsf{K},\mathsf{Q}}(r) - L_{\mathsf{K},\mathsf{Q}}^*(r)}{L_{\mathsf{K},\mathsf{Q}}^*(r)} \\
&\quad = \frac{1}{r} \cdot \frac{\sum_{l=\max\{2,r+\mathsf{Q}-\mathsf{K}\}}^{\min\{r,\mathsf{Q}-1\}} \frac{1}{l-1} \cdot C_r^l \cdot C_{\mathsf{K}-r-1}^{\mathsf{Q}-l-1}}{\sum_{l=r+\mathsf{Q}-\mathsf{K}}^{\min\{r,\mathsf{Q}-1\}} \frac{1}{l} \cdot C_r^l \cdot C_{\mathsf{K}-r-1}^{\mathsf{Q}-l-1}} \\
&\quad \le \frac{1}{r} \cdot \frac{\sum_{l=\max\{2,r+\mathsf{Q}-\mathsf{K}\}}^{\min\{r,\mathsf{Q}-1\}} \frac{1}{l-1} \cdot C_r^l \cdot C_{\mathsf{K}-r-1}^{\mathsf{Q}-l-1}}{\sum_{l=\max\{2,r+\mathsf{Q}-\mathsf{K}\}}^{\min\{r,\mathsf{Q}-1\}} \frac{1}{l} \cdot C_r^l \cdot C_{\mathsf{K}-r-1}^{\mathsf{Q}-l-1}} \\
&\quad = \frac{1}{r} \cdot \frac{\sum_{l=\max\{2,r+\mathsf{Q}-\mathsf{K}\}}^{\min\{r,\mathsf{Q}-1\}} \frac{l}{l-1} \cdot \frac{1}{l} \cdot C_r^l \cdot C_{\mathsf{K}-r-1}^{\mathsf{Q}-l-1}}{\sum_{l=\max\{2,r+\mathsf{Q}-\mathsf{K}\}}^{\min\{r,\mathsf{Q}-1\}} \frac{1}{l} \cdot C_r^l \cdot C_{\mathsf{K}-r-1}^{\mathsf{Q}-l-1}} \\
&\quad \overset{(a)}{\le} \frac{2}{r},
\end{aligned}
$$

where in (a), we used the fact $\frac{l}{l-1} \le 2$ for any $l \ge 2$.

To prove the second part, we first note that, by Corollary 3, the number of batches required by constructions P1) and P2) is

$$\mathsf{F} = \frac{1}{c} \cdot q^{\frac{\mathsf{K}}{q}}. \qquad (31)$$

On the other hand, to achieve the fundamental SC tradeoff, the number of required batches is

$$
\begin{aligned}
\mathsf{F}^* &= C_{\mathsf{K}}^r \\
&= \frac{\mathsf{K}!}{r!(\mathsf{K}-r)!} \\
&\overset{(a)}{\ge} \frac{\sqrt{2\pi}\mathsf{K}^{\mathsf{K}+\frac{1}{2}} e^{-\mathsf{K}}}{e^{\frac{1}{12}}\sqrt{2\pi} r^{r+\frac{1}{2}} e^{-r} \cdot e^{\frac{1}{12}}\sqrt{2\pi}(\mathsf{K}-r)^{\mathsf{K}-r+\frac{1}{2}} e^{-(\mathsf{K}-r)}} \\
&= \frac{1}{e^{\frac{1}{6}}} \cdot \sqrt{\frac{\mathsf{K}}{2\pi r(\mathsf{K}-r)}} \cdot \left(\frac{\mathsf{K}}{r}\right)^r \left(\frac{\mathsf{K}}{\mathsf{K}-r}\right)^{\mathsf{K}-r}
\end{aligned}
$$

$$= \frac{1}{e^{\frac{1}{6}}} \cdot \frac{q}{\sqrt{2\pi(q-1)\mathsf{K}}} \cdot q^{\frac{\mathsf{K}}{q}} \cdot \left(\frac{q}{q-1}\right)^{\mathsf{K}(1-\frac{1}{q})}, \qquad (32)$$

where (a) follows by applying Stirling's approximation $\sqrt{2\pi}n^{n+\frac{1}{2}}e^{-n} \leq n! \leq e^{\frac{1}{12}}\sqrt{2\pi}n^{n+\frac{1}{2}}e^{-n}$ to both the numerator and the denominator. Taking the ratio $\frac{\mathsf{F}}{\mathsf{F}^*}$ using (31) and (32), we complete the proof of the second part.

REFERENCES

[1] 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,* Paris, France, pp. 2803–2807, Jul. 2019.

[2] J. Dean and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," *Sixth USENIX OSDI*, Dec. 2004.

[3] 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.

[4] C. T. Chu, S. K. Kim, Y. A. Lin, Y. Y. Yu, G. Bradski, A. Y. Ng, and K. Olukotun,"Map-reduce for machine learning on multicore." In *Proc. 20th Ann. Conf. Neural Information Processing Systems (NIPS),* Vancouver, British Columbia, Canada, pp. 281–288, Dec. 2006.

[5] 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.

[6] 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, Nov. 2017.

[7] Q. Yan, S. Yang, and M. Wigger, "A storage-computation-communication tradeoff for distributed computing," in *Proc. IEEE Int. Symp. Wire. Commun. System (ISWCS),* Lisbon, Portugal, Aug. 2018.

[8] Q. Yan, S. Yang, and M. Wigger, "Storage-computation-communication tradeoff in distributed computing: Fundamental tradeoff and complexity," arXiv:1806.07565.

[9] Q. Yu, S. Li, M. A. Maddah-Ali, and A. S. Avestimehr, "How to optimally allocate resources for coded distributed computing," in *Proc. IEEE Int. Conf. Commun. (ICC), 2017,* Paris, France, 21–25, May. 2017.

[10] S. Li, Q. Yu, M. A. Maddah-Ali, A. S. Avestimehr,"A scalable framework for wireless distributed computing," *IEEE/ACM Trans. Netw.,*vol. 25, no. 5, pp. 2643–2653, Oct. 2017.

[11] S. Li, Q. Yu, M. A. Maddah-Ali, and A. S. Avestimehr, "Edge-facilitated wireless distributed computing," in *Proc. IEEE Glob. Commun. Conf. (Globlcom),* Washington, DC, USA, Dec. 2016.

[12] F. Li, J. Chen, and Z. Wang, "Wireless MapReduce distributed computing," in *Proc. IEEE Int. Symp. Inf. Theory,* Vail, CO, USA, pp. 1286–1290, Jun. 2018.

[13] E. Parrinello, E. Lampiris, and P. Elia, "Coded distributed computing with node cooperation substantially increases speedup factors," in *Proc. IEEE Int. Symp. Inf. Theory,* Vail, CO, USA, pp. 1291–1295, Jun. 2018.

[14] S. R. Srinivasavaradhan, L. Song, and C. Fragouli, "Distributed computing trade-offs with random connectivity," in *Proc. IEEE Int. Symp. Inf. Theory,* Vail, CO, USA, pp. 1281–1285, Jun. 2018.

[15] 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.

[16] 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.

[17] 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.

[18] K. Lee, C. Suh, and K. Ramchandran, "High-dimensional coded matrix multiplication," in *Proc. IEEE Int. Symp. Inf. Theory,* Aachen, Germany, pp. 2418–2422, Jun. 2017.

[19] H. Park, K. Lee, J. Sohn, C. Suh, and J. Moon, "Hierarchical coding for distributed computing," in *Proc. IEEE Int. Symp. Inf. Theory,* Vail, CO, USA, pp. 1630–1634, Jun. 2018.

[20] F. Haddadpour, and V. R. Cadambe, "Codes for distributed finite alphabet matrix-vector multiplication," in *Proc. IEEE Int. Symp. Inf. Theory,* Vail, CO, USA, pp. 1625–1629, Jun. 2018.

[21] S. Kiani, N. Ferdinand and S. C. Draper, "Exploitation of stragglers in coded computation," in *Proc. IEEE Int. Symp. Inf. Theory,* Vail, CO, USA, pp. 1988–1992, Jun. 2018.

[22] 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.

[23] N. Ferdinand and S. C. Draper, "Hierachical coded computation," in *Proc. IEEE Int. Symp. Inf. Theory,* Vail, CO, USA, pp. 1620–1624, Jun. 2018.

[24] A. Reisizadeh, S. Prakash, R. Pedarsani, and S. Avestimehr, "Coded computation over heterogeneous clusters," in *Proc. IEEE Int. Symp. Inf. Theory,* Aachen, Germany, pp. 2408–2412, Jun. 2017.

[25] R. Bitar, P. Parag, and S. E. Rouayheb, "Minimizing latency for secure coded computing using secret sharing via staircase codes", *IEEE Trans. Commun.*, early access, 2020.

[26] R. Tandon, Q. Lei, A. G. Dimakis, and N. Karampatziakis, "Gradient coding: avoiding stragglers in synchronous gradient desent," in *Proc. 34th Int. Conf. Machine Learning (ICML),* Sydney, Australia, Aug. 2017.

[27] N. Raviv, I. Tamo, R. Tandon, and A. G. Dimakis, "Gradient coding from cyclic MDS codes and expander graphs," in *Proc. 35th Int. Conf. Machine Learning, (ICML),* Stockholm, Sweden, Jul. 2018.

[28] Z. Charles, and D. Papailiopoulos,"Gradient coding using the stochastic block model," in *Proc. IEEE Int. Symp. Inf. Theory,* Vail, CO, USA, pp. 1998–2002, Jun. 2018.

[29] W. Halbawi, N. Azizan, F. Salehi, and B. Hassibi, "Improving distributed gradient descent using Reed-Solomon codes," in *Proc. IEEE Int. Symp. Inf. Theory,* Vail, CO, USA, pp. 2027–2031, Jun. 2018.

[30] E. Ozfatura, D. Gündüz and S. Ulukus, "Speeding up distributed gradient descent by utilizing non-persistent stragglers," in *Proc. IEEE Int. Symp. Inf. Theory,* Paris, France, pp. 2729–2733, Jul. 2019.

[31] S. Li, M. A. Maddah-Ali, and A. S. Avestimehr, "A unified coding framework for distributed computing with straggling servers," in *Proc. IEEE Globecom Workshop,* Washington, DC, USA, pp. 1–6, 2016.

[32] J. Zhang and O. Simeone, "Improved latency-communication trade-off for map-shuffle-reduce systems with stragglers," in *Proc. IEEE Int. Conf. Acoust., Speech & Signal Processing (ICASSP)*, Brighton, UK, May, 2019.

[33] 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.

[34] M. A. Maddah-Ali and U. Niesen, "Fundamental limits of caching," *IEEE Trans. Inf. Theory,* vol. 60, no. 5, pp. 2856–2867, May 2014.

[35] J. Wang, M. Cheng, Q. Yan, and X. Tang, "Placement delivery array design for coded caching scheme in D2D networks," *IEEE Trans. Commun,* vol. 67, no. 5, May 2019.

[36] Q. Yan, M. Wigger, and S. Yang, "Placement delivery array design for combination networks with edge caching," in *Proc. IEEE Int. Symp. Inf. Theory,* Vail, CO, USA, pp. 1555–1559, Jun. 2018.

[37] V. R. Aravind, P. Sarvepalli, and A. Thangaraj, "Subpacketization in coded caching with demand privacy," in *Proc. 26th National Conf. Commun. (NNC),*Kharagpur, India, Feb. 2020.

[38] R. Sun, H. Zheng, J. Liu, X. Du, and M. Guizani, "Placement delivery array design for the coded caching scheme in medical data sharing," *Neural Comput. & Applic. 32*, pp. 867–878, 2020.

[39] M Cheng, J Jiang, Q Yan, X Tang, "Constructions of coded caching schemes with flexible memory sizes," *IEEE Trans. Commun.*, vol. 67, no. 6, Jun. 2019.

[40] M. Cheng, J. Jiang, X. Tang, and Q. Yan, "Some variant of known coded caching schemes with good performance," *IEEE Trans. Commun.*, vol. 68, no. 3, pp. 1370–1377, Mar. 2020.

[41] M. Cheng, J. Jiang, Q. Wang, and Y. Yao, "A generalized grouping scheme in coded caching." *IEEE Trans. Commun.* vol. 67, no. 5, pp. 3422–3430, May. 2019.

[42] C. Shangguan, Y. Zhang, and G. Ge, "Centralized coded caching schemes: A hypergraph theoretical approach," *IEEE Trans. Inf. Theory,* vol. 64, no. 8, pp. 5755–5766, Aug. 2018.

[43] Q. Yan, X. Tang, Q. Chen, and M. Cheng, "Placement delivery array design through strong edge coloring of bipartite graphs," *IEEE Commun. Lett.*, vol. 22, no. 2, pp. 236–239, Feb. 2018.

[44] Q. Yan, X. H. Tang, and Q. Chen, "Placement delivery array and its appllications," in *Proc. IEEE Inf. Theory Workshop (ITW)*, Guangzhou, China, Nov. 2018.

[45] L. Tang and A. Ramamoorthy, "Coded caching schemes with reduced subpacketization from linear block codes," *IEEE Trans. Inf. Theory,* vol. 64, no. 4, pp. 3099–3120, Apr. 2018.

[46] K. Shanmugam, A. G. Dimakis, J. Llorca and A. M. Tulino, "A unified Ruzsa-Szemerédi framework for finite-length coded caching," In proc. *51st Asilomar Conference on Signals, Systems, and Computers,* Pacific Grove, CA, USA, Oct. 2017.

[47] H. A. David and H. N. Nagaraja, "Order Statistics," *John Wiley & Sons, Inc*, 2003.

Qifa Yan received the B.S. degree in mathematics and applied mathematics from Shanxi University, Taiyuan, China, in 2010, and the Ph.D. degree in communication and information system from the School of Information Science and Technology, Southwest Jiaotong University, Chengdu, China, in 2017.

From 2017 to 2019, he was a joint post-doctoral researcher at Télécom Paris, Institut Politechnique de Paris, and CentraleSupélec, Paris-Saclay University in France. He is currently a postdoctoral research fellow at the Department of Electrical and Computer Engineering of the University of Illinois at Chicago in U.S.. His research interests include caching networks, distributed computing, and other fields related to wireless networks, information theory, and coding theory.

Michèle Wigger (S'05–M'09–SM'14) received the M.Sc. degree in electrical engineering, with distinction, and the Ph.D. degree in electrical engineering both from ETH Zurich in 2003 and 2008, respectively. In 2009, she was first a post-doctoral fellow at the University of California, San Diego, USA, and then joined Telecom Paris, France, where she is currently a full professor. Dr. Wigger has held visiting professor appointments at the Technion–Israel Institute of Technology and ETH Zurich. Dr. Wigger has previously served as an Associate Editor of the IEEE Communication Letters and as an Associate Editor for Shannon Theory for the IEEE Transactions on Information Theory. During 2016–2019 she also served on the Board of Governors of the IEEE Information Theory Society. Dr. Wigger's research interests are in multi-terminal information theory.

PLACE PHOTO HERE

Sheng Yang

Xiaohu Tang (M'04-SM'18) received the B.S. degree in applied mathematics from the Northwest Polytechnic University, Xi'an, China, the M.S. degree in applied mathematics from the Sichuan University, Chengdu, China, and the Ph.D. degree in electronic engineering from the Southwest Jiaotong University, Chengdu, China, in 1992, 1995, and 2001 respectively.

From 2003 to 2004, he was a research associate in the Department of Electrical and Electronic Engineering, Hong Kong University of Science and Technology. From 2007 to 2008, he was a visiting professor at University of Ulm, Germany. Since 2001, he has been in the School of Information Science and Technology, Southwest Jiaotong University, where he is currently a professor. His research interests include coding theory, network security, distributed storage and information processing for big data.

Dr. Tang was the recipient of the National excellent Doctoral Dissertation award in 2003 (China), the Humboldt Research Fellowship in 2007 (Germany), and the Outstanding Young Scientist Award by NSFC in 2013 (China). He served as Associate Editors for several journals including *IEEE Transactions on Information Theory* and *IEICE Transactions on Fundamentals*, and served on a number of technical program committees of conferences.