Multi-User Linearly Separable Computation: A Coding Theoretic Approach

Ali Khalesi, Petros Elia

Multi-User Linearly Separable Computation: A Coding Theoretic Approach

Ali Khalesi Petros Elia
Eurecom
06410 Sophia Antipolis, France
Email: {ali.khalesi;petros.elia}@eurecom.fr

***Abstract*—In this work, we investigate the problem of multi-user linearly separable function computation, where N servers help compute the desired functions (jobs) of K users. In this setting each desired function can be written as a linear combination of up to L (generally non-linear) sub-functions. Each server computes some of the sub-tasks, and communicates a linear combination of its computed outputs (files) in a single-shot to some of the users, then each user linearly combines its received data in order to recover its desired function. We explore the range of the optimal computation cost via establishing a novel relationship between our problem, syndrome decoding and covering codes. The work reveals that in the limit of large N, the optimal computation cost — in the form of the maximum fraction of all servers that must compute any subfunction — is lower bounded as $\gamma \geq H_q^{-1}(\frac{\log_q(L)}{N})$, for any fixed $\log_q(L)/N$. The result reveals the role of the computational rate $\log_q(L)/N$, which cannot exceed what one might call the computational capacity $H_q(\gamma)$ of the system.**

***Index Terms*— Distributed computation, Linearly separable function, Coding theory**

I. Introduction

There is a recent surge in the use of distributed computing system such as MapReduce [1], Spark [2] for processing non-linear and computationally hard functions. This surge has been further intensified by recent developments relating to training large-scale machine learning algorithms such as deep neural networks with high data complexity (cf. [3]). In many such applications, the main goal is to utilize distributed parallel processing techniques to offload computations to a group of distributed servers in order to reduce the computation time. Such approaches can be found in various settings such as in [4].

This parallelization among distributed workers and servers presents new challenges in terms of accuracy, scalability, latency and straggler mitigation, privacy and security, communication and computation complexity [3], [5]. To this end, various information- and coding-theoretic works, have sought to design algorithms that alleviate the above problems, as well as establish the fundamental performance limits in various scenarios. Such works can be found in the context of computational accuracy [6]–[9], latency and straggler mitigation [10]–[13], scalability [14], [15], security and privacy [16]–[20], as well as in the context of communication and computation complexity [21]–[27]. For a detailed survey of various such works, the reader is encouraged to see [28], [29].

This work was supported by the European Research Council (ERC) through the EU Horizon 2020 Research and Innovation Program under Grant 725929 (Project DUALITY)

In this paper we adapted a data parallel, centralized topology [3], where there exist a master or dealer node who acts as a total trusted authority and manages N worker\server nodes, in the presence of L datasets\sub-tasks. Our setting also considers K agents\users, where each such user sends its demand to the master node. In particular, our setting here considers a master node that manages N server nodes that must contribute in a distributed manner to the computation of the desired function by each of the K different users. Under the linearly-separable assumption (cf. [30]), we consider that each user $k \in \{1, 2, \dots, K\}$ asks for a linearly-separable function that takes as input up to L sub-functions.

To derive bounds to the computational costs required (at the servers) to complete the computation for all users, we here establish a novel relationship between our computing problem, linear codes, syndrome decoding, and the here-introduced class of *partial-covering codes*.

In this paper, Section II introduces the system model, Section III formulates the problem, Section IV makes the connection to coding theory, Section V presents the main algebraic converse, and finally Section VI concludes the paper.

Notations: For some $n \in \mathbb{N}$, we define $[n] \triangleq \{1, 2, \dots, n\}$. For some matrix (or vector) $\mathbf{X}$, we use $\omega(\mathbf{X})$ to denote the corresponding hamming weight. We will denote the finite field $\mathbf{GF}(q)$ as $\mathbb{F}$. For some vector $\mathbf{x} \in \mathbb{F}^n$ and some code $\mathcal{C} \subseteq \mathbb{F}^n$, we will use $d(\mathbf{x}, \mathcal{C})$ to represent the hamming distance of $\mathbf{x}$ to the nearest codeword in $\mathcal{C}$. We will dedicate the use of letter ρ to correspond to a code's normalized covering radius, while we will use $\rho(\mathcal{C})$ when we want to emphasize on the normalized covering radius of a specific code $\mathcal{C} \in \mathbb{F}^n$. For some matrix $\mathbf{H}$, we will use $\mathcal{C}_{\mathbf{H}}$ to represent the linear code whose parity-check matrix is $\mathbf{H}$. Similarly, when we write $\mathbf{H}_{\mathcal{C}}$, we refer to the parity-check matrix of a linear code $\mathcal{C}$. For some $k \leq n$, $k, n \in \mathbb{N}$, we will also use $\mathcal{C}(k, n)$ to represent a linear code of message length k and codeword length n. We will use $V_q(n, \rho)$ to represent the volume of a Hamming ball in $\mathbb{F}^n$ with radius ρn. For $0 \leq x \leq 1 - \frac{1}{q}, x \in \mathbb{R}$, we will use the notation $H_q(x) \triangleq x \log_q(q-1) - x \log_q(x) - (1-x) \log_q(1-x)$

to represent the q-ary entropy function. Finally we will use $\sup(\mathbf{x}^\intercal)$ to represent the support of some vector $\mathbf{x}^\intercal \in \mathbb{F}^n$, i.e., to represent the set of indices of non-zero elements of that vector.

II. Problem Statement

In this section, we explain the Multi-user Linearly Separable Computation setting (cf. Fig. 1) which consists of K users, N servers, and a master node that coordinates servers and users[1].

In this problem, user $k \in [K]$ demands a function $F_k(D_1, \ldots, D_L)$ of L independent datasets $D_\ell, \ell \in [L]$, where this function takes the general linearly-separable form

$$F_k(D_1, D_2, \ldots, D_L) \tag{1}$$
$$\triangleq f_{k,1} f_1(D_1) + f_{k,2} f_2(D_2) + \ldots + f_{k,L} f_L(D_L) \tag{2}$$
$$= f_{k,1} W_1 + f_{k,2} W_2 + \ldots + f_{k,L} W_L \;\; k \in [K], \tag{3}$$

where in the above, $W_\ell = f_\ell(D_\ell) \in \mathbb{F}$, $\ell \in [L]$ is a so-called 'file' output, and where $f_{k,\ell} \in \mathbb{F}$, $k \in [K], \ell \in [L]$ are the linear combination coefficients which belong, together with the entries of W_ℓ, in some finite field $\mathbb{F}$.

A. Phases

The model involves three phases, with the first being the *demand phase*, then the *assignment and computation phase* and then the *transmission and decoding phase*.

In the demand phase, each user $k \in [K]$ sends the information of its desired function $F_k(.)$ to the master node, who then deduces the linear decomposition of this function according to (3). Then based on these K desired functions, during the assignment and computation phase, the master assigns some of the datasets to each server, who then proceeds to calculate the corresponding files $W_\ell = f_\ell(D_\ell)$ for their locally available datasets. Based on this assignment, each dataset D_ℓ will be placed at all the servers in some chosen server set $\mathcal{W}_\ell$.

In the assignment and computation phase, the master in accordance with the desired task functions assigns some datasets to each server, and then each server $n \in [N]$ computes all the computationally hard files from the locally-available datasets. To this end, we define the set of indices of the servers that dataset $\ell \in [L]$ has been assigned to as $\mathcal{W}_\ell \subset [N]$.

During the transmission phase, a linear combination of the locally available output files at the server is transmitted, where each server $n \in [N]$ transmits

$$z_n \triangleq \sum_{\ell \in [L]} e_{n,\ell} W_\ell \;\; n \in [N], \tag{4}$$

in a single shot (slot) over a broadcast channel to a subset of users in $\mathcal{T}_n \in [K]$. In the above, we can easily see that the encoding coefficients $e_{n,\ell}$, which are indeed determined by the master node, satisfy $e_{n,\ell} = 0, \forall (n,\ell) \in [N] \times [L]$, $n \notin \mathcal{W}_\ell$.

[1]Our setting incorporates the underlying assumption that the tasks performed at the servers substantially outweigh in computational complexity the basic linear operations that are performed at the different users and also we have assumed that each server node is connected to all of the users through a broadcast channel.

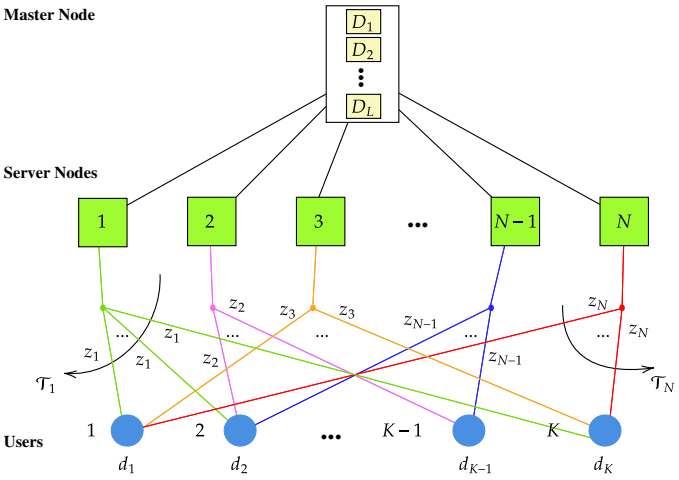

Fig. 1. The K-user, N-server Linearly Separable Computation setting. After each user informs the master of its desired function $F_k(.)$, each component subfunction $W_\ell = f_\ell(D_\ell)$ is evaluated at each server in $\mathcal{W}_\ell$. Each server n transmits a linear combination z_n (of the locally available files) to all users in $\mathcal{T}_n$. This combination is defined by the coefficients $e_{n,\ell}$. Finally, to decode, each user $k \in [K]$ linearly combines (based on decoding vectors $\mathbf{d}_k$), its received signal from all the servers it is connected to. Decoding must produce for each user its desired function $F_k(D_1, \ldots, D_L)$.

Finally during the decoding part, each user k linearly combines the received signals as follows

$$F'_k \triangleq \sum_{n \in [N]} d_{k,n} z_n, \tag{5}$$

for some decoding coefficients $d_{k,n} \in \mathbb{F}, n \in [N]$ determined again by the master node. Naturally $d_{k,n} = 0, \forall k \notin \mathcal{T}_n$. Decoding is successful when $F'_k = F_k$ for all $k \in [K]$.

B. Computation Cost

Remembering that $|\mathcal{W}_\ell|$ indicates the number of servers that compute a subfunction $W_\ell = f_\ell(D_\ell)$, $\ell \in [L]$, our *normalized computational cost* metric takes the form

$$\gamma \triangleq \frac{\max_{l \in [L]} |\mathcal{W}_l|}{N}, \tag{6}$$

to represent the maximum fraction of all servers that must compute any subfunction. We wish to establish a lower bound on this computational cost.

III. Problem Formulation

To formulate the problem we first define the below vectors,

$$\mathbf{w} \triangleq [W_1, W_2, \ldots, W_L]^\intercal, \tag{7}$$
$$\mathbf{e}_n \triangleq [e_{n,1}, e_{n,2}, \ldots, e_{n,L}]^\intercal, \; n \in [N], \tag{8}$$
$$\mathbf{E} \triangleq [\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_N]^\intercal, \tag{9}$$
$$\mathbf{d}_k \triangleq [d_{k,1}, d_{k,2}, \ldots, d_{k,N}]^\intercal, \; k \in [K], \tag{10}$$
$$\mathbf{z} \triangleq [z_1, z_2, \ldots, z_N]^\intercal, \tag{11}$$
$$\mathbf{f}_k \triangleq [f_{k,1}, f_{k,2}, \ldots, f_{k,L}]^\intercal, \; k \in [K], \tag{12}$$
$$\mathbf{f} \triangleq [F_1, F_2, \ldots, F_K]^\intercal, \tag{13}$$
$$\mathbf{f}' \triangleq [F'_1, F'_2, \ldots, F'_K]^\intercal, \tag{14}$$

$$\mathbf{F} \triangleq [\mathbf{f}_1, \mathbf{f}_2, \dots, \mathbf{f}_K]^\intercal. \tag{15}$$

Note that from (3), we have that

$$\mathbf{f} = [\mathbf{f}_1, \mathbf{f}_2, \dots, \mathbf{f}_K]^\intercal \mathbf{w}, \tag{16}$$

and then from (4) we conclude that

$$\mathbf{z} = \mathbf{E}\mathbf{w} = [\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_N]^\intercal \mathbf{w}, \tag{17}$$

which indicates what each server transmits. This transmission is defined by the so-called *encoding matrix* $\mathbf{E}$. Now from (5) we see that each user, after decoding, receives

$$F'_k = \mathbf{d}_k^T \mathbf{z}, \tag{18}$$

which in turn defines the set of all decoded signals

$$\mathbf{f}' = [\mathbf{d}_1, \mathbf{d}_2, \dots, \mathbf{d}_K]^\intercal \mathbf{z} \tag{19}$$

across all the users. This decoding is itself fully defined by the so-called*decoding matrix*

$$\mathbf{D} \triangleq [\mathbf{d}_1, \mathbf{d}_2, \dots, \mathbf{d}_K]^\intercal \in \mathbb{F}^{K \times N}. \tag{20}$$

Decoding is successful and correct if and only if

$$\mathbf{f} = \mathbf{f}' \tag{21}$$

for any $D_\ell, \ell \in [L]$. Directly from (16), (17), (19) into (21), we see that correctness holds — for any $D_\ell, \ell \in [L]$, i.e., for any $\mathbf{w} \in \mathbb{F}^{L \times 1}$ — if and only if [2]

$$\mathbf{D}\mathbf{E} = \mathbf{F}. \tag{22}$$

Noting that $\mathcal{W}_l = \sup(\mathbf{E}(:, \{l\})^\intercal)$ and that $|\mathcal{W}_l| = \omega(\mathbf{E}(:, \{l\}))$, we have that

$$\max_{l \in [L]} \omega(\mathbf{E}(:, l)) = \max_{l \in [L]} |\mathcal{W}_l| \tag{23}$$

which simply tells us that our computational cost γ from (6) takes the form

$$\gamma = \frac{1}{N} \max_{l \in [L]} \omega(\mathbf{E}(:, l)). \tag{24}$$

We here provide a simple example to help clarify the setting and the notations.

A. Simple Example

As described in Figure 2, we consider the example of a system with a master node, $N = 8$ servers, $K = 4$ users, $L = 6$ datasets, and a field of size $q = 7$.

Let us assume that the users ask the following functions:

$$\begin{aligned}
F_1 &= 2f_1(D_1) + 4f_2(D_2) + 4f_3(D_3) + 5f_4(D_4) + 5f_5(D_5),\\
F_2 &= 3f_1(D_1) + 4f_2(D_2) + 5f_3(D_3) + 2f_4(D_4) + 6f_5(D_5)\\
&\quad + 6f_6(D_6),\\
F_3 &= 2f_1(D_1) + 4f_2(D_2) + 6f_3(D_3) + 5f_4(D_4) + 2f_5(D_5),\\
F_4 &= 3f_1(D_1) + 5f_2(D_2) + 2f_4(D_4) + 3f_5(D_5) + f_6(D_6)
\end{aligned}$$

[2]Since the master node does not know about $W_\ell, \ell \in$ where designing the scheme, to make sure that (21) holds for all values of $W_\ell \in \mathbb{F}^n$, we reach $\mathbf{D}\mathbf{E} = \mathbf{F}$ is both necessary and sufficient condition. more is available on https://arxiv.org/abs/2206.11119.

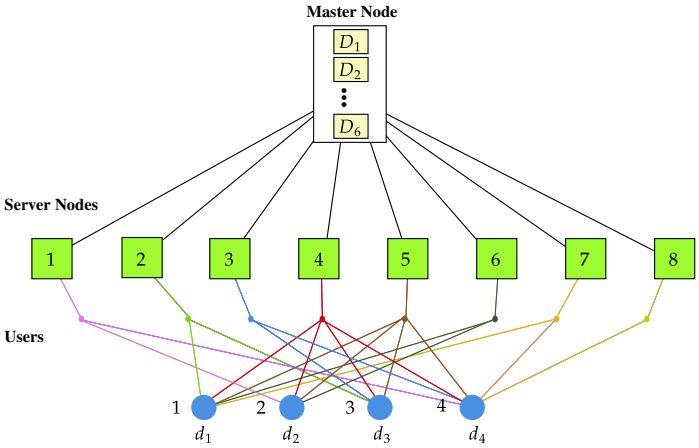

Fig. 2. Multi-user linearly separable setting with 8 servers, 4 users and 6 datasets.

where $F_k, \ k \in [4]$ is defined in (3) (after which, the corresponding $\mathbf{f}_k, \mathbf{w}$ are respectively defined in (7) and (12)). Consequently from (15), our demand matrix takes the form

$$\mathbf{F} = \begin{bmatrix} 2 & 4 & 4 & 5 & 5 & 0 \\ 3 & 4 & 5 & 2 & 6 & 6 \\ 2 & 4 & 6 & 5 & 2 & 0 \\ 3 & 5 & 0 & 2 & 3 & 1 \end{bmatrix}. \tag{25}$$

In the assignment phase, the master allocates $D_1, D_2, \dots, D_6$ to the 8 servers according to

$$\mathcal{W}_1 = \{1, 2, 3, 5, 8\}, \ \mathcal{W}_2 = \{1, 2, 3, 4, 6, 7\}, \tag{26}$$
$$\mathcal{W}_3 = \{1, 2, 3\}, \ \mathcal{W}_4 = \{1, 4, 5, 7\} \tag{27}$$
$$\mathcal{W}_5 = \{1, 2, 4, 5, 6, 8\}, \ \mathcal{W}_6 = \{3, 4, 5, 6, 7, 8\} \tag{28}$$

so that for example dataset 3 resides at servers $\{1, 2, 3\}$, and where server 2 is assigned datasets D_1, D_2, D_3, D_5 and thus has to compute $W_1 = f_1(D_1), W_2 = f_2(D_2), W_3 = f_3(D_3), W_5 = f_5(D_5)$. A quick inspection shows that the normalized computation cost (cf. (6)) is equal to

$$\gamma = \frac{\max_{l \in [6]} |\mathcal{W}_l|}{8} = 6/8. \tag{29}$$

After computing their designated output files, each server n transmits z_n as follows

$$z_1 = 2W_1 + 6W_2 + 3W_3 + W_4 + 2W_5, \tag{30}$$
$$z_2 = 4W_1 + 5W_2 + 2W_3 + 3W_5, \tag{31}$$
$$z_3 = W_1 + 2W_2 + W_3 + 2W_6, \tag{32}$$
$$z_4 = W_2 + 2W_4 + 4W_5 + W_6, \tag{33}$$
$$z_5 = 2W_1 + W_4 + 3W_5 + 2W_6, \tag{34}$$
$$z_6 = 2W_2 + 5W_5 + 3W_6, \tag{35}$$
$$z_7 = W_2 + 2W_4 + 4W_6, \tag{36}$$
$$z_8 = 2W_1 + 4W_5 + 5W_6, \tag{37}$$

corresponding to an encoding matrix (cf. (17)) of the form

$$\mathbf{E} = \begin{bmatrix} 2 & 6 & 3 & 1 & 2 & 0 \\ 4 & 5 & 2 & 0 & 3 & 0 \\ 1 & 2 & 1 & 0 & 0 & 2 \\ 0 & 1 & 0 & 2 & 4 & 1 \\ 2 & 0 & 0 & 1 & 3 & 2 \\ 0 & 2 & 0 & 0 & 5 & 3 \\ 0 & 1 & 0 & 2 & 0 & 4 \\ 2 & 0 & 0 & 0 & 4 & 5 \end{bmatrix}. \tag{38}$$

We can quickly verify (cf. (29)) that indeed $\max_{l\in[6]} \omega(\mathbf{E}(:,l))/8 = 6/8 = \gamma$.

Subsequently, the master asks each server n to send its generated z_n to its designated receiving users, such that for each server, these user sets are:

$$\mathcal{T}_1 = \{2,4\},\ \mathcal{T}_2 = \{1,3\},\ \mathcal{T}_3 = \{3\},\ \mathcal{T}_4 = \{1,2,3,4\}, \tag{39}$$

$$\mathcal{T}_5 = \{1,2,3,4\},\ \mathcal{T}_6 = \{1,2\}, \mathcal{T}_7 = \{1,4\}, \mathcal{T}_8 = \{4\}, \tag{40}$$

where for example server 2 will transmit z_2 to users 1 and 3. To decode, each user $k \in [4]$ computes the linear combination F'_k as

$$F'_1 = 2z_2 + 3z_4 + 4z_5 + 2z_6 + z_7, \tag{41}$$

$$F'_2 = 4z_1 + 2z_4 + z_5 + 3z_6, \tag{42}$$

$$F'_3 = 4z_2 + 5z_3 + 2z_4 + z_5, \tag{43}$$

$$F'_4 = 4z_1 + 2z_3 + z_4 + 2z_5 + 4z_7 + 5z_8, \tag{44}$$

corresponding to a decoding matrix of the form

$$\mathbf{D} = \begin{bmatrix} 0 & 2 & 0 & 3 & 4 & 2 & 1 & 0 \\ 4 & 0 & 0 & 2 & 1 & 3 & 0 & 0 \\ 0 & 4 & 5 & 2 & 1 & 0 & 0 & 0 \\ 4 & 0 & 2 & 1 & 2 & 0 & 4 & 5 \end{bmatrix}. \tag{45}$$

A quick verification[3] reveals the correctness of decoding and that indeed $F'_k = F_k$ for all $k = 1,2,3,4$. For example, for the first user, we see that $F'_1 = 2z_2 + 3z_4 + 4z_5 + 2z_6 + z_7 = 2(4W_1 + 5W_2 + 2W_3 + 3W_5) + 3(W_2 + 2W_4 + 4W_5 + W_6) + 4(2W_1 + W_4 + 3W_5 + 2W_6) + 2(2W_2 + 5W_5 + 3W_6) + (W_2 + 2W_4 + 4W_6) = 2W_1 + 4W_2 + 4W_3 + 5W_4 + 5W_5 + 0W_6$ which indeed matches F_1. In this example, each user recovers their desired function, with a corresponding normalized computational cost $\gamma = 3/4$. This has just been an example to illustrate the setting. The effort to find a solution with reduced computation cost, follows in the section below.

IV. A Coding Theoretic Approach

This section establishes a conceptual bridge between our problem and coding theory.

Our first aim is to decompose $\mathbf{F}$ into $\mathbf{F} = \mathbf{DE}$ under a constrained computation cost, i.e., under a sparsity constraint on $\mathbf{E}$. For $\mathbf{E}_\ell \triangleq \mathbf{E}(:,l)$ and $\mathbf{F}_\ell \triangleq \mathbf{F}(:,l)$ denoting the lth column of $\mathbf{D}$ and $\mathbf{E}$ respectively, we can rewrite our decomposition as

$$\mathbf{DE}_\ell = \mathbf{F}_\ell, \quad \ell \in [L]. \tag{46}$$

[3]Let us recall that each decoded symbol takes the form $F'_k = \mathbf{d}_k^\intercal \mathbf{z}$ where $\mathbf{d}_k^\intercal$ is the kth row of $\mathbf{D}$, and where $\mathbf{z} = [z_1\ z_2\ \cdots\ z_N]^T$.

If we viewed $\mathbf{D} \in \mathbb{F}^{K\times N}$ as a parity check matrix of a code $\mathcal{C} \subset \mathbb{F}^N$, referred to as $\mathbf{H}_\mathcal{C}$, then we could view $\mathbf{E}_\ell \in \mathbb{F}^N$ as an arbitrary error pattern, and $\mathbf{F}_\ell \in \mathbb{F}^K$ as the corresponding syndrome. Since we wish to sparsify $\mathbf{E}_\ell$, we are interested in $\mathbf{E}_\ell$ being the minimum-weight coset leader whose syndrome is $\mathbf{F}_\ell$. This is simply the output of the minimum-distance syndrome decoder. To get a first handle on the weights of $\mathbf{E}_\ell$, we can refer to the theory of covering codes which bounds the weights of coset leaders.

To derive our results for any $L \leq q^K$[4], we will first need to explore certain coding-theoretic properties, and for this reason we will transition to the traditional coding-theoretic notation which speaks of an n-length code $\mathcal{C}$ of rate k/n, where for us $n = N$ and $k = N - K$. The parity check matrix $\mathbf{H}_\mathcal{C} \in \mathbb{F}^{(n-k)\times n}$ will generally be associated to our decoding matrix $\mathbf{D} \in \mathbb{F}^{K\times N}$, the received (or error) vectors $\mathbf{x} \in \mathbb{F}^n$ will be associated to the encoding vectors $\mathbf{E}_\ell \in \mathbb{F}^N$, and its syndrome $\mathbf{s}_\mathbf{x} \in \mathbb{F}^{n-k}$ (or just $\mathbf{s}$ depending on the occasion) will be associated to $\mathbf{F}_\ell \in \mathbb{F}^K$. As suggested before, when we write $\mathcal{C}_\mathbf{D}$ (or $\mathcal{C}_\mathbf{H}$) we will refer to the code whose parity check matrix is $\mathbf{D}$ (or $\mathbf{H}$).

V. Results

As a first step, we extend the concept of covering codes to the following class of codes,

Definition 1. Let $\rho \in (0,1]$. We say that a set $\mathcal{X} \subseteq \mathbb{F}^n$ is ρ-covered by a code $\mathcal{C} \subseteq \mathbb{F}^n$ iff

$$d(\mathbf{x},\mathcal{C}) \leq \rho n, \ \ \forall \mathbf{x} \in \mathcal{X} \tag{47}$$

in which case, we say that $\mathcal{C}$ is a $(\rho,\mathcal{X})$-partial covering code.

Naturally when $\mathcal{X} = \mathbb{F}^n$, the code is simply the traditional covering code.

We are now able to link partial covering codes to our distributed computing problem via the following thheorem,

Theorem 1. *A solution to the multi-user linearly separable problem* $\mathbf{DE} = \mathbf{F}$ *with complexity* $\gamma = \frac{1}{N}\max_{\ell\in[L]} \omega(\mathbf{E}(:,l))$ *exists if and only if* $\mathbf{D}$ *is the parity check matrix to a* $(\gamma,\mathcal{X})$*-partial covering code* $\mathcal{C}_\mathbf{D}$ *for some existing set* $\mathcal{X} \supseteq \mathcal{X}_{\mathbf{F},\mathbf{D}}$*, where* $\mathcal{X}_{\mathbf{F},\mathbf{D}}$ *is defined as,*

$$\mathcal{X}_{\mathbf{F},\mathbf{D}} \triangleq \{\mathbf{x} \in \mathbb{F}^N | \mathbf{Dx} = \mathbf{F}(:,l), \text{ for some } \ell \in [L]\}. \tag{48}$$

With such $\mathbf{D}$ *in place, each* $\mathbf{E}(:,l)$ *is the output of the minimum-distance syndrome decoder of* $\mathcal{C}_\mathbf{D}$ *for syndrome* $\mathbf{F}(:,l)$.

Proof. To first prove that the complexity constraint indeed requires $\mathbf{D}$ to correspond to a partial covering code that covers $\mathcal{X}$, let us assume that $\mathbf{D}$ does not have this property, and thus there exists an $\mathbf{x} \in \mathcal{X}$ such that $d(\mathbf{x},\mathcal{C}_\mathbf{D}) > \rho n$. Let $\mathbf{c}_{\min}$ be the closest codeword to $\mathbf{x}$ in the sense that $d(\mathbf{x},\mathbf{c}_{\min}) = d(\mathbf{x},\mathcal{C}_\mathbf{D})$. Now let $\mathbf{e}_{\min} = \mathbf{x} - \mathbf{c}_{\min}$ and note, directly from the above assumption, that $\omega(\mathbf{e}_{\min}) > \rho n$.

[4]There are at most $L = q^K$ possible distinct columns for $\mathbf{F}$. We naturally assume the worst case where the columns of $\mathbf{F}$ are different.

Naturally $\mathbf{Dx} = \mathbf{D}(\mathbf{e}_{\min} + \mathbf{c}_{min}) = \mathbf{De}_{\min}$ by virtue of the fact that $\mathbf{D}$ is the parity check matrix of $\mathcal{C}_{\mathbf{D}}$. Since $\mathbf{x} \in \mathcal{X}$, we know that $\exists\, \ell \in [L] :\ \mathbf{Dx} = \mathbf{F}(:, l)$ which directly means that $\exists\, \ell \in [L] :\ \mathbf{De}_{\min} = \mathbf{F}(:, l)$. This $\mathbf{e}_{\min}$ is the coset leader associated to syndrome $\mathbf{F}(:, l)$.

Since though $\mathbf{DE} = \mathbf{F}$, we also have that $\mathbf{DE}(:, l) = \mathbf{F}(:, l)$. Since $\mathbf{E}(:, l)$ and $\mathbf{e}_{\min}$ are in the same coset (of the same syndrome $\mathbf{F}(:, l)$), and $\mathbf{e}_{\min}$ is the minimum-weight coset leader, we can conclude that $\omega(\mathbf{E}(:, l)) \geq \mathbf{e}_{\min}$. Thus the assumption that $\omega(\mathbf{e}_{\min}) > \rho n$ implies that $\omega(\mathbf{E}(:, l)) > \rho n$ which contradicts the complexity requirement that $\omega(\mathbf{E}(:, l)) \leq \rho n$ from (23) and (6) and. Thus if $\mathbf{D}$ does not correspond to a partial covering code that covers $\mathcal{X}_{\mathbf{F},\mathbf{D}}$, the complexity constraint is violated.

On the other hand, recalling that $\mathcal{C}_{\mathbf{D}}$ is a partial covering code for $\mathcal{X}$, means that for any $\mathbf{x} \in \mathcal{X}$ then $d(\mathbf{x}, \mathcal{C}_{\mathbf{D}}) \leq \rho n$. For the same $\mathbf{x} \in \mathcal{X}$, let $\mathbf{c}_{\min}$ be again its closest codeword, and let $\mathbf{e}_{\min} = \mathbf{x} - \mathbf{c}_{\min}$, where again by definition (of the partial covering code), $\omega(\mathbf{e}_{\min}) \leq \rho n$. Since, like before, $\mathbf{De}_{\min} = \mathbf{F}(:, l)$ for some $l \in [L]$, then we simply set $\mathbf{E}(:, l) = \mathbf{e}_{\min}$ whose weight is sufficiently low to guarantee the complexity constraint. We recall that for each $\mathbf{F}(:, l)$, this coset leader $\mathbf{E}(:, l) = \mathbf{e}_{\min}$ can be found using the minimum-distance syndrome decoder. □

We can now proceed with the characterisation of the converse.

Theorem 2. *For the distributed linearly separable problem with K users, N servers and any number L of sub-functions, the optimal computation cost is lower bounded as*

$$\gamma \geq H_q^{-1}(\frac{\log_q(L)}{N}). \tag{49}$$

Proof. Let $\gamma = \rho$. To prove our theorem, we first need to prove that a code $\mathcal{C}(k, n) \subseteq \mathbb{F}_q^n$ can ρn-cover a set $\mathcal{X} \subseteq \mathbb{F}_q^n$ of size $|\mathcal{X}| = q^k L$, only if

$$\log_q(L) \leq \log_q(V_q(n, \rho)). \tag{50}$$

To prove (50) we simply apply the union bound after noting that each of the q^k codewords can only ρn-cover $V_q(n, \rho)$ vectors. This implies that

$$Lq^k \leq V_q(n, \rho) q^k, \tag{51}$$

which in turn directly yields (50) after applying the logarithm on both sides of the inequality.

To complete the proof, we first assign the above set $\mathcal{X}$ to be the $\mathcal{X}$ in Theorem 1, where now $\mathcal{X} \supseteq \mathcal{X}_{\mathbf{F},\mathbf{D}}$ (cf. (48)). We know from above that $|\mathcal{X}| = Lq^k$. We can also see that $|\mathcal{X}_{\mathbf{F},\mathbf{D}}| = Lq^k$ because, by definition (cf. (48)), this set $\mathcal{X}_{\mathbf{F},\mathbf{D}}$ is the reverse image – to the ambient space $\mathbb{F}^n$ – of the all the syndromes corresponding to $\mathbf{F}$ (i.e., the reverse image of all the columns of $\mathbf{F}$). Thus now we know that $\mathcal{X} = \mathcal{X}_{\mathbf{F},\mathbf{D}}$. Then by substituting $N = n, K = n - k$, we see that $\log_q(L) \leq \log_q(V_q(N, \rho))$. By applying the well known (asymptotically tight) bound $q^{NH_q(\rho) - o(N)} \leq V_q(N, \rho) \leq q^{NH_q(\rho)}$, we can conclude that, in the limit of large L, $\log_q(L) \leq NH_q(\rho)$, thus proving that $H_q^{-1}(\frac{\log_q(L)}{N}) \leq \rho$. □

Remark 1. We see that the lower bound is only dependent on $\frac{\log_q(L)}{N}$ and if $L = q^K$, the result reduces to sphere-covering bound, which has been already known in the literature.

VI. CONCLUSION

We have explored the computational cost of the multi-user linearly separable function computation setting, which is a broad setting that captures several distributed computing problems such as the distributed gradient coding problem [13], the distributed linear transform problem [31], and the distributed matrix multiplication and the distributed multivariate polynomial computation problems [32], [33], among others. The work established a novel relationship between our problem and coding theory, and provided an algebraic converse that reveals that the normalized computational cost — in the form of the maximum fraction of all servers that must compute any subfunction — is lower bounded as $\gamma \geq H_q^{-1}(\frac{\log_q(L)}{N})$ in the limit of large L and fixed $\log_q(L)/N$.

In the journal version of this work which is available online now on https://arxiv.org/abs/2206.11119, we have provided achievable schemes for the same setting and in detail insights on this problem.

REFERENCES

[1] J. Dean and S. Ghemawat, "Mapreduce: simplified data processing on large clusters," *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[2] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica, "Spark: Cluster computing with working sets," in *2nd USENIX Workshop on Hot Topics in Cloud Computing (HotCloud 10)*, 2010.

[3] J. Verbraeken, M. Wolting, J. Katzy, J. Kloppenburg, T. Verbelen, and J. S. Rellermeyer, "A survey on distributed machine learning," *ACM Computing Surveys (CSUR)*, vol. 53, no. 2, pp. 1–33, 2020.

[4] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, *et al.*, "{TensorFlow}: A system for {Large-Scale} machine learning," in *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, pp. 265–283, 2016.

[5] S. Ulukus, S. Avestimehr, M. Gastpar, S. Jafar, R. Tandon, and C. Tian, "Private retrieval, computing and learning: Recent progress and future challenges," *IEEE Journal on Selected Areas in Communications*, 2022.

[6] T. Jahani-Nezhad and M. A. Maddah-Ali, "Codedsketch: A coding scheme for distributed computation of approximated matrix multiplication," *IEEE Transactions on Information Theory*, vol. 67, no. 6, pp. 4185–4196, 2021.

[7] J. Wang, Z. Jia, and S. A. Jafar, "Price of precision in coded distributed matrix multiplication: A dimensional analysis," in *2021 IEEE Information Theory Workshop (ITW)*, pp. 1–6, IEEE, 2021.

[8] E. Ozfatura, S. Ulukus, and D. Gündüz, "Coded distributed computing with partial recovery," *IEEE Transactions on Information Theory*, 2021.

[9] K. Wan, H. Sun, M. Ji, D. Tuninetti, and G. Caire, "Cache-aided matrix multiplication retrieval," *IEEE Transactions on Information Theory*, 2022.

[10] N. Raviv, I. Tamo, R. Tandon, and A. G. Dimakis, "Gradient coding from cyclic mds codes and expander graphs," *IEEE Transactions on Information Theory*, vol. 66, no. 12, pp. 7475–7489, 2020.

[11] A. Reisizadeh, S. Prakash, R. Pedarsani, and A. S. Avestimehr, "Tree gradient coding," in *2019 IEEE International Symposium on Information Theory (ISIT)*, pp. 2808–2812, IEEE, 2019.

[12] K. Wan, H. Sun, M. Ji, and G. Caire, "Distributed linearly separable computation," *IEEE Transactions on Information Theory*, 2021.

[13] R. Tandon, Q. Lei, A. G. Dimakis, and N. Karampatziakis, "Gradient coding: Avoiding stragglers in distributed learning," in *International Conference on Machine Learning*, pp. 3368–3376, PMLR, 2017.

[14] S. Li, S. M. M. Kalan, A. S. Avestimehr, and M. Soltanolkotabi, "Near-optimal straggler mitigation for distributed gradient methods," in *2018 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, pp. 857–866, IEEE, 2018.

[15] M. Soleymani, H. Mahdavifar, and A. S. Avestimehr, "Analog lagrange coded computing," *IEEE Journal on Selected Areas in Information Theory*, vol. 2, no. 1, pp. 283–295, 2021.

[16] M. Soleymani and H. Mahdavifar, "Distributed multi-user secret sharing," *IEEE Transactions on Information Theory*, vol. 67, no. 1, pp. 164–178, 2020.

[17] A. Khalesi, M. Mirmohseni, and M. A. Maddah-Ali, "The capacity region of distributed multi-user secret sharing," *IEEE Journal on Selected Areas in Information Theory*, vol. 2, no. 3, pp. 1057–1071, 2021.

[18] H. Akbari-Nodehi and M. A. Maddah-Ali, "Secure coded multi-party computation for massive matrix operations," *IEEE Transactions on Information Theory*, vol. 67, no. 4, pp. 2379–2398, 2021.

[19] H. A. Nodehi and M. A. Maddah-Ali, "Limited-sharing multi-party computation for massive matrix operations," in *2018 IEEE International Symposium on Information Theory (ISIT)*, pp. 1231–1235, IEEE, 2018.

[20] K. Wan, H. Sun, M. Ji, and G. Caire, "On secure distributed linearly separable computation," *IEEE Journal on Selected Areas in Communications*, 2022.

[21] S. Li, M. A. Maddah-Ali, and A. S. Avestimehr, "Compressed coded distributed computing," in *2018 IEEE International Symposium on Information Theory (ISIT)*, pp. 2032–2036, IEEE, 2018.

[22] Q. Yu, S. Li, M. A. Maddah-Ali, and A. S. Avestimehr, "How to optimally allocate resources for coded distributed computing?," in *2017 IEEE International Conference on Communications (ICC)*, pp. 1–7, IEEE, 2017.

[23] Q. Yu, M. Maddah-Ali, and S. Avestimehr, "Polynomial codes: an optimal design for high-dimensional coded matrix multiplication," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[24] W. Li, Z. Chen, Z. Wang, S. A. Jafar, and H. Jafarkhani, "Flexible distributed matrix multiplication," *arXiv preprint arXiv:2107.10448*, 2021.

[25] M. V. Jamali, M. Soleymani, and H. Mahdavifar, "Coded distributed computing: Performance limits and code designs," in *2019 IEEE Information Theory Workshop (ITW)*, pp. 1–5, IEEE, 2019.

[26] G. Suh, K. Lee, and C. Suh, "Matrix sparsification for coded matrix multiplication," in *2017 55th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pp. 1271–1278, IEEE, 2017.

[27] K. Wan, H. Sun, M. Ji, and G. Caire, "On the tradeoff between computation and communication costs for distributed linearly separable computation," *IEEE Transactions on Communications*, vol. 69, no. 11, pp. 7390–7405, 2021.

[28] J. S. Ng, W. Y. B. Lim, N. C. Luong, Z. Xiong, A. Asheralieva, D. Niyato, C. Leung, and C. Miao, "A survey of coded distributed computing," *arXiv preprint arXiv:2008.09048*, 2020.

[29] S. Li and S. Avestimehr, "Coded computing: Mitigating fundamental bottlenecks in large-scale distributed computing and machine learning," 2020.

[30] K. Wan, H. Sun, M. Ji, and G. Caire, "Distributed linearly separable computation," *IEEE Transactions on Information Theory*, pp. 1–1, 2021.

[31] S. Dutta, V. Cadambe, and P. Grover, "Short-dot: Computing large linear transforms distributedly using coded short dot products," *Advances In Neural Information Processing Systems*, vol. 29, 2016.

[32] Q. Yu, M. A. Maddah-Ali, and A. S. Avestimehr, "Straggler mitigation in distributed matrix multiplication: Fundamental limits and optimal coding," *IEEE Transactions on Information Theory*, vol. 66, no. 3, pp. 1920–1933, 2020.

[33] Z. Jia and S. A. Jafar, "Cross subspace alignment codes for coded distributed batch computation," *IEEE Transactions on Information Theory*, vol. 67, no. 5, pp. 2821–2846, 2021.