Tessellated Distributed Computing of Non-Linearly Separable Functions

Ali Khalesi, Ahmad Tanha, Derya Malak, and Petros Elia

***Abstract*—The work considers the N-server distributed computing scenario with K users requesting functions that are arbitrary multi-variable polynomial evaluations of L real (potentially non-linear) basis subfunctions of a certain degree. Our aim is to reduce both the computational cost at the servers, as well as the load of communication between the servers and the users. To do so, we take a novel approach, which involves transforming our distributed computing problem into a sparse tensor factorization problem $\bar{\mathcal{F}} = \bar{\mathcal{E}} \times_1 \mathbf{D}$, where tensor $\bar{\mathcal{F}}$ represents the requested non-linearly-decomposable jobs expressed as the mode-1 product between tensor $\bar{\mathcal{E}}$ and matrix $\mathbf{D}$, where $\mathbf{D}$ and $\bar{\mathcal{E}}$ respectively define the communication and computational assignment, and where their sparsity respectively allows for reduced communication and computational costs. We here design an achievable scheme, designing $\bar{\mathcal{E}}, \mathbf{D}$ by utilizing novel fixed-support SVD-based tensor factorization methods that first split $\bar{\mathcal{F}}$ into properly sized and carefully positioned subtensors, and then decompose them into properly designed subtensors of $\bar{\mathcal{E}}$ and submatrices of $\mathbf{D}$. For the zero-error case and under basic dimensionality assumptions, this work reveals a lower bound on the optimal rate K/N with a given communication and computational load.**

***Keywords*—Distributed computing, MapReduce, sparse tensor factorization, low-rank tensor approximation, tessellations, tensor decomposition, random matrix theory, capacity, bulk synchronous parallel.**

I. Introduction

There is a growing demand for massive parallel computing approaches that effectively distribute computations across distributed servers [1], [2]. To address this pressing need, numerous works have introduced novel methods tackling various aspects of distributed computing, including scalability [3]–[6], privacy and security [7]–[13], completion time [14], as well as latency and straggler mitigation [15]–[17], among others. For a comprehensive overview of such related works, the reader is encouraged to consult [18], [19].

Beyond these considerations, the renowned computation-versus-communication tradeoff lies at the heart of distributed computing as a foundational principle with far-reaching implications. This tradeoff emerges as a critical limiting factor in numerous distributed computing scenarios [20], [21], including the practical settings of *multi-user, multi-server distributed computation of linearly-separable functions*, addressing several classes of computing problems, e.g., gradient coding [22]–[25], linear-transform computation [26]–[28], matrix multiplication, or multivariate polynomial computation [15], [29]–[31], as well as training of large-scale machine learning algorithms and deep neural networks with massive data [20], where communication and computation are often the two interdependent bottlenecks that significantly shape overall performance.

While multi-user linearly-separable settings have been extensively studied from various perspectives, such as coding theory [32]–[34], compressed sensing [35], and tessellation-based schemes for fixed-support matrix factorization [36], to our best knowledge, for the broader scenario of non-linearly separable functions, which nicely captures several classes of computing problems in approximation theory [37], [38], neural networks [39], [40], non-linear optimization [41], the prior work has not explored the coding gains and the achievable bounds on the computation cost. This work is the first to address non-linearly-separable functions.

A. Multi-User Non-Linearly Separable Distributed Computing

We focus on the very broad and arguably practical N-*server distributed computing framework* with K users requesting from distributed servers arbitrary multi-variable polynomial evaluations of L real (potentially non-linear) basis subfunctions of a certain degree, as we will detail below.

Our setting, as depicted in Figure 1, initially considers a *master node* that coordinates, in three phases, a set of N distributed *servers* that compute functions requested by the K *users*. During the initial *demand* phase, each user $k \in [K]$ independently requests the computed output of a single real function $F_k(.)$. Under the real-valued non-linear separability assumption[1], these functions take the basic form

$$F_k(.) = \sum_{\underline{\mathbf{p}} \in [P_1] \times \ldots \times [P_L]} f_{k,\underline{\mathbf{p}}} \, W_1^{p_1 - 1} \ldots W_L^{p_L - 1} \qquad (1)$$

where $W_\ell = f_\ell(x), x \in \mathcal{D}$, denotes the real-valued *output file* of $f_\ell(\cdot)$ for a multi-variate input x from any domain set $\mathcal{D}$, $f_\ell(\cdot)$ denotes a *(basis or component) subfunction*, and

This work was partially supported by the Huawei France-funded Chair towards Future Wireless Networks, and the program "PEPR Networks of the Future" of France 2030.

Co-funded by the European Union (ERC, SENSIBILITÉ, 101077361). Views and opinions expressed are however those of the authors only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them.

The authors are with the Communication Systems Department at EURECOM, 450 Route des Chappes, 06410 Sophia Antipolis, France (Emails: {khalesi, tanha, malak, elia}@eurecom.fr).

[1]This nicely captures non-linearly separable functions, where each $F_k(.)$, taking L subfunctions as input, can be written as a non-linear combination of L *univariate* basis subfunctions.

$f_{k,\underline{\mathbf{p}}}$ denotes a real-valued basis coefficient. Moreover, the range of degrees for each basis subfunction W_ℓ in each user's demand is equal to $[0 : P_\ell - 1], \forall \ell \in [L]$. In (1), each basis subfunction $f_\ell(\cdot)$ has a degree $p_\ell - 1$ that introduces a *multiplication cost*[1] *to computation*. Hence, the cost of computing the p_ℓ-th power of each $f_\ell(\cdot)$ for each server is considered a multiplication cost of p_ℓ versus the computation time of $f_\ell(\cdot)$. Capturing these multiplication costs enables us to analyze the critical communication-computation tradeoff for distributed computing of general multi-variable polynomials.

Subsequently, during the *computing phase*, the master assigns to each server $n \in [N]$, a set of subfunctions $\mathcal{S}_n \subseteq [L]$ to compute locally[2] and also assigns to each server $n \in [N]$, a set of functions[3] $\mathcal{M} \in 2^{\prod_{\ell \in \mathcal{S}_n}[P_\ell]}$ to generate the corresponding $\prod_{\ell \in \mathcal{S}_n} W_\ell^{p_\ell - 1}$ from the computed output functions. $\mathcal{M}$ determines how to compute the powers of different subfunctions, and how to multiply the output subfunctions computed locally on each server. Then during the *communication phase*, each server n forms signals

$$z_{n,t} \triangleq \sum_{\underline{\mathbf{p}} \in \mathcal{P}_n} e_{n,\underline{\mathbf{p}},t} \prod_{\ell \in \mathcal{S}_n} W_\ell^{p_\ell - 1}, \quad n \in [N], \ t \in [T] \tag{2}$$

as dictated by the *encoding coefficients* $e_{n,\underline{\mathbf{p}},t} \in \mathbb{R}, n \in [N], t \in [T]$, which we will design later on, where

$$\mathcal{P}_n \triangleq \{\underline{\mathbf{p}} \mid \prod_{\ell \in \mathcal{S}_n} W_\ell^{p_\ell - 1} \text{ is computed in server } n\} \ .$$

We note that $p_\ell = 1, \forall \ell \notin \mathcal{S}_n$, reflecting the cases where given the absence of certain basis subfunctions, it does not apply to consider their powers. The n^{th} server subsequently proceeds to transmit $z_{n,t}$ during time-slot $t = 1, 2, \ldots, T$, to a subset of users $\mathcal{T}_n \subseteq [K]$, via an error-free shared link. Finally, during the decoding part of the last phase, each user k linearly combines its received signals to get

$$F'_k \triangleq \sum_{n \in [N], t \in [T]} d_{k,n,t} z_{n,t} \tag{3}$$

as dictated by the *decoding coefficients* $d_{k,n,t} \in \mathbb{R}, n \in [N], t \in [T], k \in [K]$. Naturally $d_{k,n,t} = 0, \forall k \notin \mathcal{T}_n$, simply because user $k \in [K]$ does not receive any symbols from server $n \in [N]$ during any time $t \in [T]$. Note that both encoding and decoding coefficients are determined by the master node after the demand phase, and thus are dependent on the functions requested, but remain independent of the instance of the *input* to the requested functions.

Furthermore, we consider computation, communication, and multiplication costs

$$\Gamma \triangleq \max_{n \in [N]} |\mathcal{S}_n| \ , \ \Delta \triangleq \max_{n \in [N]} |\mathcal{T}_n| \ , \tag{4}$$

[1]The cost of computing powers of each basis subfunction for each server is considered as a multiplication cost.

[2]We will interchangeably use $\mathcal{S}_n$ to describe sets of indices of subfunctions, as well as the subfunctions themselves.

[3]2^S means the power set of S, where we here consider the cardinality of the range of degrees for basis subfunctions belonging to sets $\mathcal{S}_n$.

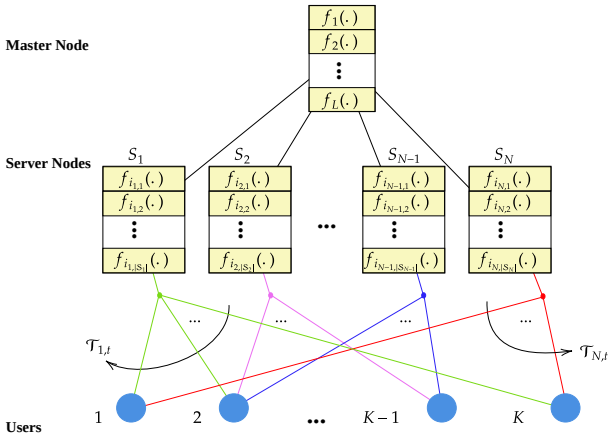

Fig. 1. The K-user, N-server, T-shot setting. Each server n computes the subfunctions in $\mathcal{S}_n = \{f_{i_{n,1}}(.), f_{i_{n,2}}(.), \ldots, f_{i_{n,|\mathcal{S}_n|}}(.)\}$ and multiplicative terms $f_{i_{n,1}}^{p_{i_{n,1}}-1}(.) f_{i_{n,2}}(.)^{p_{i_{n,2}}-1} \ldots f_{i_{n,|\mathcal{S}_n|}}(.)^{p_{i_{n,|\mathcal{S}_n|}}-1}$ where $(1, \ldots, 1, p_{i_{n,1}}, 1, \ldots, 1, p_{i_{n,2}}, 1, \ldots 1, p_{i_{n,|\mathcal{S}_n|}}, 1)$ communicates to users in $\mathcal{T}_{n,t}$, under computational constraint $|\mathcal{S}_n| \leq \Gamma \leq L$, communication constraint $|\mathcal{T}_n| \leq \Delta \leq K$ and multiplication constraint $\Lambda_\ell \leq P_\ell, \forall \ell \in \mathcal{Q}_i$, yielding a system with normalized constraints $\gamma = \frac{\Gamma}{L}, \delta = \frac{\Delta}{K}, \lambda_\ell = \frac{\Lambda_\ell}{P_\ell}$, where $\gamma, \delta, \lambda_\ell \in [0, 1]$.

$$\Lambda_\ell \triangleq \max_{\underline{\mathbf{p}} \in \mathcal{P}_n} p_\ell - \min_{\underline{\mathbf{p}} \in \mathcal{P}_n} p_\ell + 1 \tag{5}$$

respectively representing the maximum number of basis subfunctions to be locally computed at any server[4], the maximum number of users that a server can communicate to, and finally, the range of powers of basis subfunctions that must be computed locally at each server among all multiplicative statements in (1). After normalization, we here consider the normalized costs

$$\gamma \triangleq \frac{\Gamma}{L} \ , \ \delta \triangleq \frac{\Delta}{K} \ , \ \lambda_\ell \triangleq \frac{\Lambda_\ell}{P_\ell} \ . \tag{6}$$

The above three parameters are bounded. In brief, γ is the fraction of subfunctions that must be computed locally, δ is the fraction of available links to be activated, and $\lambda_\ell, \ell \in [L]$ is the normalized range of degrees of each basis subfunction to be computed by each server. Having $\gamma = 1$ corresponds to the centralized scenario of having to locally calculate all subfunctions, while $\delta = 1$ matches an extreme parallelized scenario that activates all available communication links. From (5), when $\Lambda_\ell = P_\ell = 2$, leading to $\lambda_\ell = 1, \forall \ell \in [L]$, we capture the special case of multi-user linearly separable settings of [36]. In particular, it has been shown in Proposition 2 that the proposed scheme in this paper surpasses the performance of the initial Tessellated Distributed computing scheme due to better computational resource allocations.

In a system defined by K, N, and L, our goal is to find schemes that can recover any set of desired functions without error, with the smallest possible computation, communication,

[4]We highlight that the constraints Γ, Δ, and Λ_ℓ are strict, meaning they must be satisfied for every instance of the problem.

and multiplication loads γ, δ, and λ_ℓ. To do so, we must carefully decide which subfunctions each server computes, and which combinations of computed outputs each server sends to which users. Having to serve many users with fewer servers naturally places a burden on the system (suggesting higher γ, δ, and λ_ℓ), bringing to the fore the concept of the *system rate*

$$R \triangleq \frac{K}{N} \tag{7}$$

and the corresponding *system capacity* C representing the supremum of all rates.

B. Connection to sparse tensor factorization

Toward analysing our distributed computing problem, we can see from (1) that the desired functions are fully represented by a tensor $\bar{\mathcal{F}} \in \mathbb{R}^{K \times P_1 \times \ldots \times P_L}$ of the aforementioned coefficients $f_{k,\underline{\mathbf{p}}}$. With $\bar{\mathcal{F}}$ in place, we must decide on the computation-assignment and communication (encoding and decoding) protocol.

As we have seen in [42], for the error-free case, this task is equivalent — directly from (2),(3)— to solving a (sparse) tensor factorization problem of the form

$$\bar{\mathcal{E}} \times_1 \mathbf{D} = \bar{\mathcal{F}} \tag{8}$$

where, as we will specify later on, the $NT \times P_1 \times \ldots \times P_L$ *computing-and-encoding tensor* $\bar{\mathcal{E}}$ holds the coefficients $e_{n,\underline{\mathbf{p}},t}$ from (2), while the $K \times NT$ *communication matrix* $\mathbf{D}$ holds the decoding coefficients $d_{k,n,t}$ from (3). Furthermore, we aim to decompose a given tensor $\bar{\mathcal{F}}$ as a mode-1 product of a tensor $\bar{\mathcal{E}}$ and a matrix $\mathbf{D}$, satisfying sparsity constraints γ, δ, and λ_ℓ. Particularly, we leverage tensors as high-dimensional arrays to precisely capture the multiplicative terms associated with each basis subfunction in each user's demands.

Summary of our contributions: Having made the connection between tensor factorization and distributed computing, we here identify the fixed support tensor factorization problem to be key in the resolution of our real-valued multi-user distributed computing problem.

In this paper, we only focus on the lossless case. By employing an achievable scheme using novel concepts and algorithms introduced in [42], [43], Theorems 1 and 2 establish the single-shot system capacity for both cases $P_\ell > \Lambda_\ell$ and $P_\ell = \Lambda_\ell$, where these general cases are interesting because the tessellation patterns we design must accommodate tiles of various sizes and shapes.

C. Paper Organization

The paper is organized as follows. Section II formulates the system model for multi-user distributed computing of non-linearly-separable functions. Section III addresses the error-free case, presenting schemes for single-shot scenarios, leading to Theorems 1 and 2. We compare our tensor-based solutions with the matrix approach in [36], highlighting significant gains in computation cost and server requirements for both cases where $P_\ell > \Lambda_\ell$ and $P_\ell = \Lambda_\ell$ (cf. Propositions 1 and 2). Finally, Section IV concludes the paper.

Notations. We define $[n] \triangleq \{1, 2, \ldots, n\}$ and $\underline{\mathbf{p}} \triangleq (p_1, \ldots, p_\ell, \ldots, p_L)$. $[[a : b]]$ demonstrates the ordered set of integers, ranging from a to b. For matrices $\mathbf{A}$ and $\mathbf{B}$, $[\mathbf{A}, \mathbf{B}]$ indicates the horizontal concatenation of the two matrices. For any matrix $\mathbf{X} \in \mathbb{R}^{m \times n}$, then $\mathbf{X}(i, j)$, $i \in [m]$, $j \in [n]$, represents the entry in the ith row and jth column, while $\mathbf{X}(i, :), i \in [m]$, represents the ith row, and $\mathbf{X}(:, j)$, $j \in [n]$ represents the jth column of $\mathbf{X}$. For two index sets $\mathcal{I} \subset [m], \mathcal{J} \in [n]$, then $\mathbf{X}(\mathcal{I}, \mathcal{J})$ represents the submatrix comprised of the rows in $\mathcal{I}$ and columns in $\mathcal{J}$. We will use $\text{supp}(\mathbf{x}^\intercal)$ to represent the support of some vector $\mathbf{x}^\intercal \in \mathbb{R}^n$, describing the set of indices of non-zero elements, and $\text{supp}(\mathbf{X})$ to represent the support of some matrix $\mathbf{X} \in \mathbb{R}^{m \times n}$, describing the set of two-dimensional indices $(i, j) \in [m] \times [n]$ of non-zero elements. $\mathbb{I}(\mathcal{X}, \mathcal{Y}) \in \mathbb{R}^{m \times n}$ represents a matrix that all of its elements are zero except the elements, $\mathbb{I}(i, j), i \in \mathcal{X}, j \in \mathcal{Y}$. $\text{Supp}(\mathbf{A}) \in \{0, 1\}^{m \times n}, \mathbf{A} \in \mathbb{R}^{m \times n}$, is a binary matrix representing the locations of non-zero elements of $\mathbf{A}$. All the above matrix notations are extended to tensors. For tensors $\bar{\mathcal{E}}_1$ and $\bar{\mathcal{E}}_2$, $[\bar{\mathcal{E}}_1, \bar{\mathcal{E}}_2]_1$ represents the mode-1 concatenation of the two tensors. The operation $\odot$ represents a Hadamard product. For a real number $x \in \mathbb{R}$, $\lceil x \rceil, \lfloor x \rfloor$, represents respectively the ceiling and floor function of x. $\bar{\mathcal{X}}$ represents a tensor. $\times_n, n \in \mathbb{N}$ is the mode-n tensor product and $\times_{[[n]]}^{[[m]]}$ represents the generalized tensor contraction product operation. $\mathbb{1}$ is the indicator function and $\mathbb{1}(\ell \in \mathcal{Q})$ then returns 1 if $\ell \in \mathcal{Q}$. Subset$([L], \Gamma)$ denotes the subsets of L subfunctions with cardinality Γ. *stack*$_1$ stands for joining multiple tensors together along dimension 1, creating a higher-dimensional tensor.

II. Problem Formulation

We here describe in detail the main parameters of our model, defining matrix $\mathbf{D}$ and tensors $\bar{\mathcal{E}}, \bar{\mathcal{F}}$ and the various metrics, rigorously linking the distributed computing problem and the factorization in (8) for our error-free case. Let us consider

$$\mathbf{f} = [F_1, F_2, \ldots, F_K]^\intercal \in \mathbb{R}^K \ , \tag{9}$$

$$\bar{\mathcal{F}}_k(\underline{\mathbf{p}}) = f_{k,\underline{\mathbf{p}}} \in \mathbb{R}^{P_1 \times P_2 \times \ldots \times P_L}, \ k \in [K] \ , \tag{10}$$

$$\bar{\mathcal{W}}(\underline{\mathbf{p}}) \triangleq W_1^{p_1 - 1} W_2^{p_2 - 1} \ldots W_L^{p_L - 1} \in \mathbb{R}^{P_1 \times P_2 \times \ldots \times P_L} \ , \quad \forall \underline{\mathbf{p}} \in [P_1] \times [P_2] \times \ldots \times [P_L] \tag{11}$$

where $\mathbf{f}$ represents the vector of desired function outputs F_k from (1), $\bar{\mathcal{F}}_k$ represents the tensor of function coefficients $f_{k,\underline{\mathbf{p}}}$ from (1) for the function requested by user k, and $\bar{\mathcal{W}}$ denotes the tensor of multiplicative terms of the output files $W_\ell = f_\ell(\cdot)$ again from (1). Then recalling the encoding coefficients $e_{n,\underline{\mathbf{p}},t}$ and transmitted signals $z_{n,t}$ from (2), as well as the decoding coefficients $d_{k,n,t}$ and decoded functions F'_k from (3), we have

$$\bar{e}_{n,t}(\underline{\mathbf{p}}) \triangleq e_{n,\underline{\mathbf{p}},t} \prod_{\ell \in \mathcal{S}_n} W_\ell^{p_\ell - 1}, \ \forall \underline{\mathbf{p}} \in \mathcal{P}_n \ , \quad \bar{e}_{n,t} \in \mathbb{R}^{P_1 \times \ldots \times P_L}, \ n \in [N], \ t \in [T] \tag{12}$$

and

$$\mathbf{z}_n \triangleq [z_{n,1}, \ldots, z_{n,T}]^\intercal \in \mathbb{R}^T, \ n \in [N] \ ,$$

$$\bar{\mathcal{E}}_n \triangleq \mathit{stack}_1(\bar{e}_{n,1},\ldots,\bar{e}_{n,T}) \in \mathbb{R}^{T\times P_1\times\ldots\times P_L} \ , \tag{13}$$

$$\mathbf{d}_{k,n} \triangleq [d_{k,n,1},\ldots,d_{k,n,T}]^\intercal \in \mathbb{R}^T,\ k\in[K], n\in[N] \ , \tag{14}$$

$$\mathbf{d}_k \triangleq [\mathbf{d}_{k,1}^\intercal,\ldots,\mathbf{d}_{k,N}^\intercal]^\intercal \in \mathbb{R}^{NT},\ k\in[K] \tag{15}$$

and thus from (9), we have the output vector taking the form

$$\mathbf{f} = \mathit{stack}_1(\bar{\mathcal{F}}_1,\bar{\mathcal{F}}_2,\ldots,\bar{\mathcal{F}}_K) \times_{[[2:L+1]]}^{[[1:L]]} \bar{\mathcal{W}} \tag{16}$$

as well as the transmitted vector by server n taking the form

$$\mathbf{z}_n = \bar{\mathcal{E}}_n \times_{[[2:L+1]]}^{[[1:L]]} \bar{\mathcal{W}} \ . \tag{17}$$

This allows us to form the tensors

$$\bar{\mathcal{F}} \triangleq \mathit{stack}_1(\bar{\mathcal{F}}_1,\ldots,\bar{\mathcal{F}}_K) \in \mathbb{R}^{K\times P_1\times P_2\times\ldots\times P_L}, \tag{18}$$

$$\bar{\mathcal{E}} \triangleq [\bar{\mathcal{E}}_1,\ldots,\bar{\mathcal{E}}_N]_1 \in \mathbb{R}^{NT\times P_1\times\ldots\times P_L}, \tag{19}$$

$$\mathbf{D} \triangleq [\mathbf{d}_1,\ldots,\mathbf{d}_K]^\intercal \in \mathbb{R}^{K\times NT} \tag{20}$$

where $\bar{\mathcal{F}}$ represents the $K\times P_1\times\ldots\times P_L$ tensor of all function coefficients across all the users, $\bar{\mathcal{E}}$ represents the aforementioned $NT\times P_1\times\ldots\times P_L$ *computing tensor* capturing the computing and linear encoding tasks of servers in each shot, and $\mathbf{D}$ represents the $K\times NT$ *communication matrix* capturing the communication protocol and the linear decoding task done by each user. All presented results and schemes assume our multi-user non-linearly-separable distributed computing problem and assume, in their entirety, linear encoding at the servers and linear decoding at the users.

To see the transition to the tensor factorization problem, we first note that from (2) and (17), the overall transmitted vector $\mathbf{z} \triangleq [\mathbf{z}_1^\intercal,\mathbf{z}_2^\intercal,\ldots,\mathbf{z}_N^\intercal]^\intercal \in \mathbb{R}^{NT}$ takes the form

$$\mathbf{z} = [\bar{\mathcal{E}}_1,\ldots,\bar{\mathcal{E}}_N]_1 \times_{[[2:L+1]]}^{[[1:L]]} \bar{\mathcal{W}} = \bar{\mathcal{E}} \times_{[[2:L+1]]}^{[[1:L]]} \bar{\mathcal{W}} \tag{21}$$

and then that given the decoding phase from (3), each retrieved function takes the form

$$F_k' = \mathbf{d}_k^\intercal \mathbf{z} \tag{22}$$

thus resulting in the vector of all retrieved functions taking the form $\mathbf{f}' = [\mathbf{d}_1,\mathbf{d}_2,\ldots,\mathbf{d}_K]^\intercal\mathbf{z}$. We aim to make sure that

$$\mathbf{f}' = \mathbf{f} \ . \tag{23}$$

Directly from the above and as in (8), we see that for the error-free case, we must solve the tensor factorization problem

$$\bar{\mathcal{F}} = \bar{\mathcal{E}} \times_1 \mathbf{D} \ . \tag{24}$$

In terms of the corresponding connection to the sparsity of $\mathbf{D}$ and $\bar{\mathcal{E}}$, we recall from (4) our metrics $\Gamma = \max_{n\in[N]} |\mathcal{S}_n|$, $\Delta = \max_{n\in[N]} |\mathcal{T}_n|$, and $\Lambda_\ell = \max\limits_{\underline{\mathbf{p}}\in\mathcal{P}_n} p_\ell - \min\limits_{\underline{\mathbf{p}}\in\mathcal{P}_n} p_\ell + 1$, which directly from (15)–(17) and from (18)–(20), imply the communication constraint as

$$\max_{n\in[N]} |\cup_{t=1}^T \mathrm{supp}(\mathbf{D}(:,(n-1)T+t))| \le \Delta \ , \tag{25}$$

and the computation constraint as

$$\max_{n\in[N]} \sum_{\ell\in\mathcal{Q}_i} |\mathbb{1}(\cup_{t=1}^T \mathrm{supp}(\bar{\mathcal{E}}((n-1)T+t, \underbrace{:,\ldots,:}_{\ell-1 \text{ times}}, [2:P_\ell], \underbrace{:,\ldots,:}_{L-\ell \text{ times}}) \neq \emptyset)| \le \Gamma \tag{26}$$

and the multiplication constraint as

$$\|\bar{\mathcal{E}}(p_1,p_2,\ldots,p_{\ell-1},:,p_{\ell+1},\ldots,p_L)\|_0 \le \Lambda_\ell,\ \forall \ell\in\mathcal{Q}_i \ ,$$
$$\forall \underline{\mathbf{p}} \in [P_1]\times[P_2]\times\ldots\times[P_L]$$

and thus we see how the normalized costs $\delta,\gamma,\lambda_\ell$ from (6) form the upper bounds on the fraction of non-zero elements of the columns of $\mathbf{D}$, special non-zero support of subtensors of $\bar{\mathcal{E}}$, and non-zero elements of each 1-dimensional subtensors of $\bar{\mathcal{E}}$, respectively.

Finally, from (7), we recall the system rate $R = \frac{K}{N}$, the corresponding system capacity C representing the supremum of all rates for error-free function reconstruction.

III. Main Results

We present the achievable rate results for the proposed distributed computing settings for the simplified single-shot cases, assuming $\delta,\lambda_\ell \in \mathbb{N}$ in Theorem 1 and $\delta\in\mathbb{N}$, $\lambda_\ell = 1$ in Theorem 2, respectively. For the general multi-shot cases, we refer the reader to Theorem 3 for $\delta,\lambda_\ell \notin \mathbb{N}$ and Theorem 4 for $\delta\notin\mathbb{N}$, $\lambda_\ell = 1$ both in Appendix C available online in [44]. All the results hold without any restriction on the dimensions and under the assumption that each subtensor of $\bar{\mathcal{F}}$ has full-rank, a condition that is easily justified in our real-function settings. Particularly, we consider a mode-1 matrix unfolding[1] of tensor $\bar{\mathcal{F}}$ and use its rank properties to elaborate the number of required servers as the sum of the total number of tiles with different maximum representative rank-one supports. Since in general we have $\Gamma\le L$, our tiles have some combinatorial intersections, where we introduce a combinatorial optimization based on a bipartite matching approach to assign the intersecting subtitles to one of the tiles for each $\mathcal{Q}_i \in \text{Subset}([L],\Gamma)$, $\forall i\in[\binom{L}{\Gamma}]$. This procedure results in tiles with dimensions $|X_{i,\text{opt}}|$ and $|X''_{i,\text{opt}}|$ for the cases $\Lambda_\ell | P_\ell$ and $\Lambda_\ell \nmid P_\ell$, respectively. All the results hold under the basic disjoint support assumption on matrix $\mathbf{D}$ and tensor $\bar{\mathcal{E}}$ (cf. Definition 9).

Theorem 1. *The optimal achievable rate of the lossless $K,N,T = 1, L, P_\ell > \Lambda_\ell, \Gamma, \Delta, \Lambda_\ell, \ell \in \mathcal{Q}_i$ distributed computing system for the broad settings in which $\Delta | K$, $\Lambda_\ell | P_\ell$ takes the form $C = K/N_{opt}$, where*

$$N_{opt} \le \frac{K}{\Delta}\sum_{i=1}^{\binom{L}{\Gamma}} \Big(\min\big(\Delta, \prod_{\ell\in\mathcal{Q}_i}\Lambda_\ell\big) \prod_{\ell\in\mathcal{Q}_i}\big(\frac{P_\ell}{\Lambda_\ell}-1\big) + \min\big(\Delta, |X_{i,opt}|\big) \prod_{\ell\in\mathcal{Q}_i\setminus\mathcal{L}_i}\big(\frac{P_\ell}{\Lambda_\ell}-1\big)\Big) \tag{27}$$

where $\mathcal{L}_i \triangleq \{\ell\in\mathcal{Q}_i \mid j_\ell = 1\}$, and j_ℓ is the index in tensor $\bar{\mathcal{F}}$ corresponding to the ℓ^{th} basis subfunction.

[1] Unfolding, also referred to as matricization or flattening, is the process of rearranging the elements of a multi-dimensional array into a matrix format.

Proof. Due to lack of space (partly because our proofs require a sizable set of definitions), we choose to present the proof in the supplementary document (cf. Appendix C available online in [44].), which describes the achievability of the corresponding decomposition $\bar{\mathcal{E}} \times_1 \mathbf{D} = \bar{\mathcal{F}}$, and how this is translated into our distributed computing setting while abiding by the constraints communication cost δ, computation cost γ, and multiplication cost λ_ℓ. To provide some insight, we present a proof sketch.

Sketch of the proof: The proof consists of an achievable scheme, consisting of three main steps. In the first step, the master node divides $\bar{\mathcal{F}}$ into properly sized hypercubic subtensors (corresponding to 'tiles') where the tiles must remain disjoint, cover all elements of $\bar{\mathcal{F}}$, and maintain a width limit Δ and modal constraint Λ_ℓ, $\forall \ell \in \mathcal{Q}_i$ for Γ modes in subset $\mathcal{Q}_i$ of L modes. In the second step, the master node performs a multilinear singular value decomposition (SVD) for each tile (for each of the aforementioned subtensors of $\bar{\mathcal{F}}$), so that this SVD defines the way each tile is the mode-1 product of a subtensor by a submatrix (the so-called left factor and right factor of the multilinear SVD). Finally, in the third step, the left and right factors are carefully positioned to form the desired $\bar{\mathcal{E}}$ and $\mathbf{D}$ that in turn define the computation and communication protocol of the distributed computing problem. □

We next argue in Proposition 1 that considering a potentially large dimension L in multi-user linearly separable settings [36] to accommodate a greater number of basis subfunctions results in a higher total computational load per server. In contrast, our proposed model achieves improved efficiency in total computation cost by fewer basis subfunctions, each with a bounded degree, captured by the multiplication cost. Our comparison assumes that the time of computing a basis subfunction is the same as the time of computing one power of its output.

Proposition 1. *In a distributed computing system with $K, N, \Delta, L, P_\ell = P > 2, \Lambda_\ell = \Lambda$, $\forall \ell \in [L]$, considering the tensor solution in this work with a computation cost $\Gamma = L$ and the multiplication costs Λ, the total computation cost is upper bounded by $NL(\Lambda + 1)$, which linearly scales with L. While the matrix approach of [36] with a computation cost Γ' results in an upper bound $N\Big(\Gamma' + \big(\frac{(P-2)(P-1)}{2}\big)^L\Big)$ on the total computation cost, which exponentially grows with L.*

Proof. See Appendix E available online in [44]. □

Theorem 2. *The optimal achievable rate of the lossless $K, N, T = 1, L, P_\ell = \Lambda_\ell, \Gamma, \Delta, \Lambda_\ell, \ell \in \mathcal{Q}_i$ distributed computing system for the broad settings in which $\Delta | K$ takes the form $C = K/N_{opt}$, where*

$$N_{opt} \leq \frac{K}{\Delta} \sum_{i=1}^{\binom{L}{\Gamma}} \min\big(\Delta, |X_{i,opt}|\big) \ . \tag{28}$$

We next discuss under which conditions our proposed tensor approach's performance outperforms the matrix approach's result in [36] for the scenario where $P_\ell = \Lambda_\ell = 2, \forall l \in [L]$.

Proposition 2. *In a distributed computing system with $K, \Delta > 1, L, P_\ell = \Lambda_\ell = 2$, $\forall \ell \in [L]$, letting N_M and N_T denote the required number of servers using the matrix approach in [36] and the proposed tensor approach in this work, respectively, we have $N_M > N_T$ in the following regimes:*

- *Asymptotic case (large L):*
  In this case, $N_M > N_T$ happens only if Δ grows rapidly relative to L and Γ, i.e.,

$$\begin{cases} \Delta > \frac{L}{\Gamma} & \text{when} \quad 1 < \Delta \leq \Gamma \ , \\ \Gamma^2 > L & \text{when} \quad \Delta > \Gamma \ . \end{cases} \tag{29}$$

- *Non-asymptotic case (small L):*
  In this case, $N_M > N_T$ holds if and only if:

$$\Delta > \frac{\Gamma \cdot \binom{L}{\Gamma}}{\sum_{d=1}^{\Gamma-1} \binom{L}{d}} \quad \text{when} \quad 1 < \Delta \leq \Gamma \ .$$

In the case of large L, if we set $\Gamma = O(\sqrt{L})$, our scheme can achieve $\gamma = O(1/\sqrt{L})$, which becomes arbitrarily small as L increases. This corresponds to a broad asymptotic regime where $\kappa \triangleq K/L$ remains fixed, and $L \to \infty$. In this scenario, to meet (29), the parameter δ only needs to satisfy the condition $\delta > O(\kappa/\sqrt{L})$, which also captures an arbitrarily small normalized communication cost.

Proof. See Appendix F. □

IV. Conclusion

In this work, we investigated the fundamental limits of lossless multi-user distributed computing of real-valued multivariable polynomials by making clear connections to the problem of fixed support tensor factorization and tessellation theory. The error-free system capacity $C = \frac{K}{N_{\text{opt}}}$, as presented in Theorems 1 and 2, identifies the minimal computational and communication resources γ, δ needed to support a specific number of users and subfunctions, considering a normalized multiplication cost λ_ℓ. The general results for arbitrary parameters $T, K, L, \Delta, \Gamma, \Lambda_\ell, \ell \in [L]$ are detailed in Theorem 3 for the case $P_\ell > \Lambda_\ell$, and in Theorem 4 for $P_\ell = \Lambda_\ell$. Both theorems are elaborated in Appendix C available online in [44].

Our approach achieves an exponential reduction in computation cost compared to the Tessellated Distributed Computing approach when $P_\ell > \Lambda_\ell$ while having equal computation times. Additionally, it outperforms the previous scheme for large L in the case where $P = \Lambda = 2$, leveraging a bipartite matching algorithm for efficient resource allocation.

Our future directions include exploring the lossy case, where the users can recover the requested functions with a bounded error, as well as the converse bounds on the optimal number of servers, revealing a fundamental tradeoff between communication, computation, and multiplication costs.

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] S. Li, Q. Yu, M. A. Maddah-Ali, and A. S. Avestimehr, "A scalable framework for wireless distributed computing," *IEEE/ACM Transactions on Networking*, vol. 25, no. 5, pp. 2643–2654, 2017.

[4] 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.

[5] F. Brunero, K. Wan, G. Caire, and P. Elia, "Coded distributed computing for sparse functions with structured support," in *2023 IEEE Information Theory Workshop (ITW)*, 2023, pp. 474–479.

[6] F. Brunero and P. Elia, "Multi-access distributed computing," *IEEE Transactions on Information Theory*, vol. Early Access, pp. 1–1, 2024.

[7] T. Jahani-Nezhad, M. A. Maddah-Ali, S. Li, and G. Caire, "Swiftagg+: Achieving asymptotically optimal communication loads in secure aggregation for federated learning," *IEEE Journal on Selected Areas in Communications*, vol. 41, no. 4, pp. 977–989, 2023.

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

[9] 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.

[10] M. Soleymani, H. Mahdavifar, and A. S. Avestimehr, "Privacy-preserving distributed learning in the analog domain," *arXiv preprint arXiv:2007.08803*, 2020.

[11] M. Soleymani, R. E. Ali, H. Mahdavifar, and A. S. Avestimehr, "List-decodable coded computing: Breaking the adversarial toleration barrier," *IEEE Journal on Selected Areas in Information Theory*, vol. 2, no. 3, pp. 867–878, 2021.

[12] J. So, R. E. Ali, B. Güler, J. Jiao, and A. S. Avestimehr, "Securing secure aggregation: Mitigating multi-round privacy leakage in federated learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, 2023, pp. 9864–9873.

[13] F. Engelmann and P. Elia, "A content-delivery protocol, exploiting the privacy benefits of coded caching," in *15th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*, Paris, France, 2017, pp. 1–6.

[14] E. Lampiris and P. Elia, "Full coded caching gains for cache-less users," *IEEE Transactions on Information Theory*, vol. 66, no. 12, pp. 7635–7651, Dec 2020.

[15] 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.

[16] E. Parrinello, E. Lampiris, and P. Elia, "Coded distributed computing with node cooperation substantially increases speedup factors," in *IEEE International Symposium on Information Theory (ISIT)*, Vail, CO, USA, 2018, pp. 1291–1295.

[17] T. Jahani-Nezhad and M. A. Maddah-Ali, "Berrut approximated coded computing: Straggler resistance beyond polynomial computing," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 1, pp. 111–122, Feb. 2022.

[18] J. S. Ng, W. Y. B. Lim, N. C. Luong, Z. Xiong, A. Asheralieva, D. Niyato, C. Leung, and C. Miao, "A comprehensive survey on coded distributed computing: Fundamentals, challenges, and networking applications," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 3, pp. 1800–1837, 2021.

[19] S. Li and S. Avestimehr, *Coded Computing: Mitigating Fundamental Bottlenecks in Large-Scale Distributed Computing and Machine Learning*. now Publishers Inc, 2020, vol. 17. [Online]. Available: http://dx.doi.org/10.1561/0100000103

[20] 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.

[21] 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.

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

[23] M. Ye and E. Abbe, "Communication-computation efficient gradient coding," in *International Conference on Machine Learning*. PMLR, 2018, pp. 5610–5619.

[24] 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.

[25] N. Charalambides, H. Mahdavifar, and A. O. Hero, "Generalized fractional repetition codes for binary coded computations," *IEEE Transactions on Information Theory*, pp. 1–1, January 2025.

[26] 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.

[27] S. Wang, J. Liu, N. Shroff, and P. Yang, "Fundamental limits of coded linear transform," *arXiv preprint arXiv:1804.09791*, 2018.

[28] S. Vithana and S. Ulukus, "Private read-update-write with controllable information leakage for storage-efficient federated learning with top r sparsification," *IEEE Transactions on Information Theory*, vol. 70, no. 5, pp. 3669–3692, December 2023.

[29] A. Ramamoorthy, L. Tang, and P. O. Vontobel, "Universally decodable matrices for distributed matrix-vector multiplication," in *2019 IEEE International Symposium on Information Theory (ISIT)*, 2019, pp. 1777–1781.

[30] A. Tanha and D. Malak, "The influence of placement on transmission in distributed computing of boolean functions," in *25th IEEE International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, Lucca, Italy, Sep. 2024.

[31] D. Malak, M. R. Deylam Salehi, B. Serbetci, and P. Elia, "Multi-server multi-function distributed computation," *Entropy*, vol. 26, no. 6, 2024. [Online]. Available: https://www.mdpi.com/1099-4300/26/6/448

[32] A. Khalesi and P. Elia, "Multi-user linearly separable computation: A coding theoretic approach," in *2022 IEEE Information Theory Workshop (ITW)*, 2022, pp. 428–433.

[33] ——, "Multi-user linearly-separable distributed computing," *IEEE Transactions on Information Theory*, vol. 69, no. 10, pp. 6314–6339, 2023.

[34] ——, "Perfect multi-user distributed computing," in *2024 IEEE International Symposium on Information Theory (ISIT)*, Athens, Greece, 2024, pp. 1349–1354.

[35] A. Khalesi, S. Daei, M. Kountouris, and P. Elia, "Multi-user distributed computing via compressed sensing," in *2023 IEEE Information Theory Workshop (ITW)*, 2023, pp. 509–514.

[36] A. Khalesi and P. Elia, "Tessellated distributed computing," *To appear in* IEEE Trans. Inf. Theory, Nov. 2024.

[37] M. J. D. Powell, *Approximation theory and methods*. Cambridge university press, 1981.

[38] B. C. Csáji *et al.*, "Approximation with artificial neural networks," *Faculty of Sciences, Etvs Lornd University, Hungary*, vol. 24, no. 48, p. 7, Jun. 2001.

[39] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural networks*, vol. 2, no. 5, pp. 359–366, Jan. 1989.

[40] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?" in *International Conference on Learning Representations*, 2019.

[41] D. P. Bertsekas, *Nonlinear Programming*. Athena Scientific, 1999.

[42] L. De Lathauwer, B. De Moor, and J. Vandewalle, "A multilinear singular value decomposition," *SIAM Journal on Matrix Analysis and Applications*, vol. 21, no. 4, pp. 1253–1278, 2000. [Online]. Available: https://doi.org/10.1137/S0895479896305696

[43] Q.-T. Le, E. Riccietti, and R. Gribonval, "Spurious valleys, np-hardness, and tractability of sparse matrix factorization with fixed support," *SIAM Journal on Matrix Analysis and Applications*, vol. 44, no. 2, pp. 503–529, 2023.

[44] *Tessellated Distributed Computing of Non-Linearly Separable Functions*. Zenodo, Jan. 2025. [Online]. Available: https://doi.org/10.5281/zenodo.14721265

[45] S. V. Dolgov and D. V. Savostyanov, "Alternating minimal energy methods for linear systems in higher dimensions," *SIAM Journal on Scientific Computing*, vol. 36, no. 5, pp. A2248–A2271, 2014. [Online]. Available: https://doi.org/10.1137/140953289

[46] P. J. Cameron, *Combinatorics: topics, techniques, algorithms*. Cambridge University Press, 1994.

[47] J. E. Hopcroft and R. M. Karp, "An n^5/2 algorithm for maximum matchings in bipartite graphs," *SIAM Journal on computing*, vol. 2, no. 4, pp. 225–231, 1973.

APPENDIX

A. Basic Tensor Definitions

Definition 1. *An order-N tensor*, $\bar{\mathcal{X}} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$, is a multi-way array with N modes, with the n-th mode of dimensionality I_n, for $n \in [N]$. Special cases of tensors include matrices as order-2 tensors (e.g., $\bar{\mathcal{X}} \in \mathbb{R}^{I_1 \times I_2}$), vectors as order-1 tensors (e.g., $\mathbf{x} \in \mathbb{R}^{I_1}$), and scalars as order-0 tensors (e.g., $x \in \mathbb{R}$).

Definition 2. A *mode-n unfolding* of a tensor is the procedure of mapping the elements from a multidimensional array to a two-dimensional array (matrix). Conventionally, such a procedure is associated with stacking mode-n fibers (modal vectors) as column vectors of the resulting matrix. For instance, mode-1 unfolding of $\bar{\mathcal{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is represented as $\bar{\mathcal{X}}_{(1)} \in \mathbb{R}^{I_1 \times I_2 I_3 \cdots I_N}$, and given by

$$\bar{\mathcal{X}}_{(1)} \left[i_1, \overline{i_2 i_3 \dots i_N} \right] = \bar{\mathcal{X}}[i_1, i_2, \dots, i_N] \ . \tag{30}$$

Note that the overlined subscripts refer to linear indexing (or Little-Endian) [45], given by

$$\begin{aligned} \overline{i_1 i_2 \dots i_N} &= 1 + \sum_{n=1}^{N} \left[(i_n - 1) \prod_{n'=1}^{n-1} I_{n'} \right] \\ &= 1 + i_1 + (i_2 - 1) I_1 + \cdots + (i_N - 1) I_1 \dots I_{N-1}. \end{aligned} \tag{31}$$

Definition 3. Any given vector $x \in \mathbb{R}^{I_1 I_2 \dots I_N}$ can be *folded* into an N-th order tensor, $\bar{\mathcal{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, with the relation between their entries defined by

$$\bar{\mathcal{X}}[i_1, i_2, \dots, i_N] = x_i, \ \forall i_n \in [I_n] \tag{32}$$

where $i = 1 + \sum_{n=1}^{N} (i_n - 1) \prod_{k=1}^{n-1} I_k$.

Definition 4. Consider grouping J order-N tensor samples, $\bar{\mathcal{X}} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$, so as to form an order-$(N+1)$ data tensor, $\bar{\mathcal{Y}} \in \mathbb{R}^{I_1 \times \cdots \times I_N \times J}$. This *stacking* operation is denoted by

$$\bar{\mathcal{Y}} = stack_{N+1} \left(\bar{\mathcal{X}}[1], \dots, \bar{\mathcal{X}}[J] \right) \ . \tag{33}$$

In other words, the so combined tensor samples introduce another dimension, the $(N+1)$-th mode of $\bar{\mathcal{Y}}$, such that the mode-$(N+1)$ unfolding of Y becomes

$$\bar{\mathcal{Y}}_{(N+1)} = \left[vec(\bar{\mathcal{X}}[1]), \dots, vec(\bar{\mathcal{X}}[J]) \right] \ . \tag{34}$$

Definition 5. The *mode-n product* takes as input an order-N tensor, $\bar{\mathcal{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, and a matrix, $\mathcal{A} \in \mathbb{R}^{J \times I_n}$, to produce another tensor, $\bar{\mathcal{Y}}$, of the same order as the original tensor $\bar{\mathcal{X}}$. The operation is denoted by

$$\bar{\mathcal{Y}} = \bar{\mathcal{X}} \times_n \mathcal{A} \tag{35}$$

where $\bar{\mathcal{Y}} \in \mathbb{R}^{I_1 \times \cdots \times I_{n-1} \times J \times I_{n+1} \times \cdots \times I_N}$. The mode-$n$ product is comprised of 3 consecutive steps:

$$\begin{aligned} \bar{\mathcal{X}} &\rightarrow \bar{\mathcal{X}}_{(n)} \ , \\ \bar{\mathcal{Y}}_{(n)} &= \mathcal{A} \bar{\mathcal{X}}_{(n)} \ , \\ \bar{\mathcal{Y}}_{(n)} &\rightarrow \bar{\mathcal{Y}} \ . \end{aligned} \tag{36}$$

Definition 6. *Tensor Contraction Product* (TCP) is at the core of tensor decompositions, an operation similar to the mode-n product, but the arguments of which are multidimensional arrays that can be of a different order. For instance, given an N-th order tensor $\bar{\mathcal{X}} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ and another M-th order tensor $\bar{\mathcal{Y}} \in \mathbb{R}^{J_1 \times \cdots \times J_M}$, with common modes $I_n = J_m$, then their (n,m)-contraction denoted by $\times_n^m$, yields a third tensor $\bar{\mathcal{Z}} \in \mathbb{R}^{I_1 \times \cdots \times I_{n-1} \times I_{n+1} \times \cdots \times I_N \times J_1 \times \cdots \times J_{m-1} \times J_{m+1} \times \cdots \times J_M}$ of order $(N+M-2)$, $\bar{\mathcal{Z}} = \bar{\mathcal{X}} \times_n^m \bar{\mathcal{Y}}$ with entries

$$\begin{aligned} &z_{i_1,\dots,i_{n-1},i_{n+1},\dots,i_N,j_1,\dots,j_{m-1},j_{m+1},\dots,j_M} \\ &= \sum_{i_n \in [I_n]} x_{i_1,\dots,i_{n-1},i_n,i_{n+1},\dots,i_N} \ y_{j_1,\dots,j_{m-1},i_n,j_{m+1},\dots,j_M} \ . \end{aligned} \tag{37}$$

The overwhelming indexing associated with the TCP operation in (37) quickly becomes unmanageable for larger tensor networks, whereby multiple TCPs are carried out across a large number of tensors. Manipulation of such expressions is prone to errors and prohibitive to manipulation of higher-order tensors.

Definition 7. *Generalized Tensor Contraction Product* is an operation similar to TCP, but the common modes of two tensors are considered as the ordered sets of integers. For instance, given an N-th order tensor $\bar{\mathcal{X}} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ and another M-th order tensor $\bar{\mathcal{Y}} \in \mathbb{R}^{J_1 \times \cdots \times J_M}$, with common modes $I_p = J_q, \forall p \in [[n:N]], q \in [[m:M]]$, then their $([[n:N]],[[m:M]])$-contraction denoted by $\times_{[[n:N]]}^{[[m:M]]}$, yields a third tensor $\bar{\mathcal{Z}} \in \mathbb{R}^{I_1 \times \cdots \times I_{n-1} \times J_1 \times \cdots \times J_{m-1}}$ of order $(n+m-2)$, $\bar{\mathcal{Z}} = \bar{\mathcal{X}} \times_{[[n:N]]}^{[[m:M]]} \bar{\mathcal{Y}}$ with entries

$$z_{i_1,\dots,i_{n-1},j_1,\dots,j_{m-1}} = \sum_{i_p \in [I_p]} x_{i_1,\dots,i_{n-1},i_p} y_{j_1,\dots,j_{m-1},i_p}$$

where $i_p = [[i_n : i_N]]$ and $I_p = [[I_n : I_N]]$.

Quickly recall that for a $L+1$ dimensional tensor $\bar{\mathcal{A}}$, then $\bar{\mathcal{A}}(:,\dots,n)$ represents a L dimensional subtensor and $\mathbf{B}(:,n)$ for a matrix $\mathbf{B}$ is its nth column, Supp$(\mathbf{A})$(Supp$(\bar{\mathcal{A}})$) is the binary matrix (tensor) indicating the support of $\mathbf{A}$ ($\bar{\mathcal{A}}$). Also, when we refer to a support constraint, this will be in the form of a binary matrix (tensor) that indicates the support (the position of the allowed non-zero elements) of a matrix (tensor) of interest [36].

Before proceeding with the scheme, we here also give a very brief reminder on the basic concepts regarding SVD decompositions.

B. A Primer on Multilinear SVD

The multilinear singular value decomposition (MLSVD) extends the concept of the matrix singular value decomposition (SVD) into the multilinear domain [42]. This decomposition

provides a powerful tool for analyzing tensors and obtaining low-rank multilinear approximations.

For matrices, the SVD is well-known and expressed as

$$\mathbf{M} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\intercal} \ , \tag{38}$$

where $\mathbf{M} \in \mathbb{R}^{J_1 \times J_2}$ is an arbitrary real-valued matrix, $\mathbf{\Sigma} \in \mathbb{R}^{I_1 \times I_2}$ is a diagonal matrix with the entries $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_{\min(I_1,I_2)} \geq 0$ in descending order, and $\mathbf{U} \in \mathbb{R}^{J_1 \times I_1}$ and $\mathbf{V} \in \mathbb{R}^{J_2 \times I_2}$ are orthogonal matrices.

Using mode-n tensor-matrix products, we have:

$$\mathbf{M} = \mathbf{\Sigma} \times_1 \mathbf{U} \times_2 \mathbf{V}^{\intercal} \ . \tag{39}$$

The MLSVD generalizes this decomposition to higher-order tensors. In the literature [42], it is also referred to as the higher-order SVD or Tucker decomposition, though "Tucker decomposition" has evolved into a broader term. The MLSVD of a N^{th} order tensor is represented as

$$\bar{\mathcal{T}} = \bar{\mathcal{S}} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \ldots \times_N \mathbf{U}^{(N)} \ , \tag{40}$$

where $\bar{\mathcal{T}} \in \mathbb{R}^{J_1 \times J_2 \times \ldots \times J_N}$, $\bar{\mathcal{S}} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$, and $\mathbf{U}^{(n)} \in \mathbb{R}^{J_n \times I_n}$, $\forall n \in [N]$.

Similar to the matrix case, where $\mathbf{U}$ and $\mathbf{V}$ serve as orthonormal bases for the column and row spaces, the MLSVD computes N orthonormal bases $\mathbf{U}^{(n)} \in \mathbb{R}^{J_n \times I_n}$, $\forall n \in [N]$ for the N subspaces of mode-n vectors. These bases are analogous to those in the matrix SVD. The MLSVD essentially interprets a tensor as a collection of mode-n vectors and performs the matrix SVD for each mode. The subtensors $\bar{\mathcal{S}}_{i_n=\alpha}$ of $\bar{\mathcal{S}}$ are obtained by fixing the n^{th} index to α with the following properties:

- *All-orthogonality*[1]: Subtensors $\bar{\mathcal{S}}_{i_n=\alpha}$ and $\bar{\mathcal{S}}_{i_n=\beta}$ are orthogonal for all possible n, α, β if

$$\langle \bar{\mathcal{S}}_{i_n=\alpha}, \bar{\mathcal{S}}_{i_n=\beta} \rangle = 0 \ \text{ when } \ \alpha \neq \beta \ , \tag{41}$$

  where the scalar product of two tensors $\bar{\mathcal{T}}, \bar{\mathcal{S}} \in \mathbb{R}^{I_1 \times \ldots \times I_N}$ is defined as

$$\langle \bar{\mathcal{T}}, \bar{\mathcal{S}} \rangle \triangleq \sum_{i_1} \ldots \sum_{i_N} t_{i_1 \ldots i_N} s_{i_1 \ldots i_N} \ , \tag{42}$$

  where t and s represent the elements of tensors $\bar{\mathcal{T}}$ and $\bar{\mathcal{S}}$, respectively.
- *Ordering*:

$$\sigma_1^{(n)} \geq \sigma_2^{(n)} \geq \ldots \geq \sigma_{I_n}^{(n)} \geq 0 \ , \tag{43}$$

  where symbols $\sigma_i^{(n)}$ represent the n-mode singular values of $\bar{\mathcal{S}}$, and are equal to the Frobenius-norms $\|\bar{\mathcal{S}}_{i_n=i}\|$, $\forall i \in [I_n]$, where the Frobenius-norm of tensor $\bar{\mathcal{T}}$ is described by

$$\|\bar{\mathcal{T}}\| \triangleq \langle \bar{\mathcal{T}}, \bar{\mathcal{T}} \rangle \ . \tag{44}$$

The n-rank of a tensor can be examined using matrix-based methods. The rank of a higher-order tensor follows the fact that a rank-n matrix can be decomposed as a sum of n rank-1 elements. The n-mode vectors of $\bar{\mathcal{T}}$ are the column vectors of the matrix unfolding $\bar{\mathcal{T}}_{(n)}$. We thus have:

$$\text{rank}_n(\bar{\mathcal{T}}) = \text{rank}(\bar{\mathcal{T}}_{(n)}) \ . \tag{45}$$

[1]Arrays with a scalar product of zero are considered orthogonal.

C. Scheme for Lossless Reconstruction and Achievability Proof for Theorem 3

We proceed to describe the design of the communication matrix $\mathbf{D}$ and the computing tensor $\bar{\mathcal{E}}$ that yield $\bar{\mathcal{E}} \times_1 \mathbf{D} = \bar{\mathcal{F}}$, while maintaining N, T as well as a maximum of Δ non-zero elements in any column of $\mathbf{D}$ and a maximum of Λ_ℓ non-zero elements in any row of subtensors of $\bar{\mathcal{E}}$ for any subset of L subfunctions with cardinality Γ. These constraints guarantee that each of the N servers can locally calculate up to a range Λ_ℓ of Γ subfunctions and can engage in communication with at most Δ users. The design borrows concepts and definitions from the blockwise SVD approach of [43], generalized to the tensor scenario in [42].

In this proof we first in Part C1 begin with defining the n-th rank-one contribution support, representative rank-one support, or tile and their related parameters. These tiles are actually classes of rank-one contribution supports which shows how any support constraint on $\mathbf{D}, \bar{\mathcal{E}}$ contributes to the supports on $\bar{\mathcal{E}} \times_1 \mathbf{D}$. The aforementioned correspondence is shown via Lemma 1. Then in Part C2, we show how to design the tiles for the product matrix $\bar{\mathcal{E}} \times_1 \mathbf{D}$, create and fill the non-zero tiles in $\mathbf{D}$ and $\bar{\mathcal{E}}$ and place the filled tiles in $\mathbf{D}$ and in $\bar{\mathcal{E}}$ for both single-shot. The number of servers required for the single-shot required the rank of each tile to the servers and then summing the ranks over all tiles. Example 1 illustrates the performance of our scheme in the cases where $\Delta | K, \Lambda_\ell | P_\ell$, and $T = 1$.

1) Basic Concepts and Definitions: We first present the following definitions[2].

Definition 8 ([43])**.** Given two support constraints $\mathbf{I} \in \{0,1\}^{K \times NT}$ and $\bar{\mathcal{J}} \in \{0,1\}^{NT \times P_1 \times P_2 \times \ldots \times P_L}$, then for any $n \in [NT]$, we refer to $\bar{\mathcal{S}}_n(\mathbf{I}, \bar{\mathcal{J}}) \triangleq \bar{\mathcal{J}}(n, :, \ldots, :) \times_1 \mathbf{I}(:, n)$ as the n-th *rank-one contribution support*.

We note that when the supports are implied, we may shorten $\bar{\mathcal{S}}_n(\mathbf{I}, \bar{\mathcal{J}})$ to just $\bar{\mathcal{S}}_n$. The aforementioned $\mathbf{I}$ and $\bar{\mathcal{J}}$ will generally represent the support of $\mathbf{D}$ and $\bar{\mathcal{E}}$ respectively, while $\bar{\mathcal{S}}_n(\mathbf{I}, \bar{\mathcal{J}})$ will generally capture some of the support of $\bar{\mathcal{E}} \times_1 \mathbf{D}$ and thus of $\bar{\mathcal{F}}$. We have the following lemma for this.

Lemma 1. For $\mathbf{I} \triangleq \text{supp}(\mathbf{D})$ and $\bar{\mathcal{J}} \triangleq \text{supp}(\bar{\mathcal{E}})$, then

$$\begin{aligned} \cup_{n=1}^{NT} \bar{\mathcal{S}}_n(\mathbf{I}, \bar{\mathcal{J}}) &= \cup_{n=1}^{NT} \bar{\mathcal{J}}(n, :, \ldots, :) \times_1 \mathbf{I}(:, n) \\ &\supseteq \text{supp}(\bar{\mathcal{E}} \times_1 \mathbf{D}) \ . \end{aligned} \tag{46}$$

Proof. The above follows from Definition 8 and from the fact that $\bar{\mathcal{E}} \times_1 \mathbf{D} = \sum_{n=1}^{NT} \bar{\mathcal{E}}(n, :, \ldots, :) \times_1 \mathbf{D}(:, n)$. □

We also need the following definitions.

[2]We quickly recall that for a matrix $\mathbf{A}$, $\mathbf{A}(:, n)$ represents its n-th column, and for a Tensor $\bar{\mathcal{T}}$, $\bar{\mathcal{T}}(n, :, \ldots, :)$ demonstrates its n-th row, $\text{supp}(\mathbf{A})$ the binary matrix indicating the support of $\mathbf{A}$, while for a vector $\mathbf{a}$, then $\text{supp}(\mathbf{a})$ represents the set of indices of $\mathbf{a}$ with non-zero elements.

Definition 9. (*Disjoint Support Assumption.*) We say that matrix $\mathbf{D} \in \mathbb{R}^{K\times NT}$ and tensor $\bar{\mathcal{E}} \in \mathbb{R}^{NT\times P_1\ldots\times P_L}$, accept the *disjoint support assumption* if and only if for any two columns $\mathbf{D}(:,i), \mathbf{D}(:,i'), i, i' \in [NT]$ of $\mathbf{D}$ and the respective two subtensors $\bar{\mathcal{E}}(i,\underbrace{:,\ldots,:}_{L \text{ times}}), \bar{\mathcal{E}}(i',\underbrace{:,\ldots,:}_{L \text{ times}})$ of $\bar{\mathcal{E}}$, then $\text{supp}(\bar{\mathcal{E}}(i,:,\ldots,:)\times_1\mathbf{D}(:,i)) = \text{supp}(\bar{\mathcal{E}}(i',:,\ldots,:)\times_1\mathbf{D}(:,i'))$ or $\text{supp}(\bar{\mathcal{E}}(i,:,\ldots,:) \times_1 \mathbf{D}(:,i)) \cap \text{supp}(\bar{\mathcal{E}}(i',:,\ldots,:) \times_1 \mathbf{D}(:,i')) = \emptyset$.

Definition 10. Given two supports $\mathbf{I} \in \{0,1\}^{K\times NT}$ and $\bar{\mathcal{J}} \in \{0,1\}^{NT\times P_1\times P_2\times\ldots\times P_L}$, the *equivalence classes of rank-one supports* are defined by the equivalence relation $i \sim j$ on $[NT]$, which holds if and only if $\bar{\mathcal{S}}_i = \bar{\mathcal{S}}_j$, as represented in Figure 2.

The above splits the columns of $\mathbf{D}$ (and correspondingly the rows of $\bar{\mathcal{E}}$) into equivalence classes such that $i \sim j$ holds if and only if $\bar{\mathcal{J}}(i,:,\ldots,:)\times_1\mathbf{I}(:,i) = \bar{\mathcal{J}}(j,:,\ldots,:)\times_1\mathbf{I}(:,j)$.

Definition 11. For two supports $\mathbf{I} \in \{0,1\}^{K\times NT}, \bar{\mathcal{J}} \in \{0,1\}^{NT\times P_1\times P_2\times\ldots\times P_L}$, and for $\mathcal{C}$ being the collection of equivalence classes as in Definition 10, then for each class $\mathcal{P} \in \mathcal{C}$, we call $\bar{\mathcal{S}}_\mathcal{P}$ to be the *representative rank-one support* of class $\mathcal{P}$, which will also be called the *tile* labeled[1] by $\mathcal{P}$. Furthermore for each tile $\bar{\mathcal{S}}_\mathcal{P}$, let $\mathbf{r}_\mathcal{P} \triangleq \mathbf{I}(:,n), n \in \mathcal{P}$ (resp. $\mathbf{c}_\mathcal{P} \triangleq \bar{\mathcal{J}}(n,:,\ldots,:), n \in \mathcal{P}$) be the corresponding *component column* (resp. *component row*) of the representative rank-one support. Finally, $\mathcal{R}_\mathcal{P} \triangleq \text{supp}(\mathbf{r}_\mathcal{P}) \subset [K]$ describes the set of the indices of the non-zero elements in $\mathbf{r}_\mathcal{P}$, while $\mathcal{C}_\mathcal{P} \triangleq \text{supp}(\mathbf{c}_\mathcal{P}) \subset [P_1]\times\ldots\times[P_L]$ describes the set of indices of the non-zero elements in $\mathbf{c}_\mathcal{P}$. This is illustrated in Figure 3.

Definition 12. For every set $\mathcal{C}' \subseteq \mathcal{C}$ of equivalence classes, let us define the *union of the representative rank-one supports* $\mathcal{S}_{\mathcal{C}'} \triangleq \cup_{\mathcal{P}\in\mathcal{C}'}\bar{\mathcal{S}}_\mathcal{P}$ to be the point-wise logical OR of the corresponding $\bar{\mathcal{S}}_\mathcal{P}$.

In the above, $\mathcal{S}_{\mathcal{C}'}$ is simply the area of the product tensor covered by all the tiles $\mathcal{P}$ in $\mathcal{C}'$. Furthermore, we have the following definition.

Definition 13 ([43])**.** The *maximum rank of a representative rank-one support of class* $\mathcal{P} \subset \mathcal{C}$ is

$$r_\mathcal{P} \triangleq \min(|\mathcal{C}_\mathcal{P}|, |\mathcal{R}_\mathcal{P}|) \ . \tag{47}$$

The above simply says that for the case of $\mathbf{I} = \text{supp}(\mathbf{D})$ and $\bar{\mathcal{J}} = \text{supp}(\bar{\mathcal{E}})$, then the part of tensor $\bar{\mathcal{E}} \times_1 \mathbf{D}$ covered by tile $\bar{\mathcal{S}}_\mathcal{P}$ can have rank which is at most $r_\mathcal{P}$.

With the above in place, we proceed to describe the method used to design the matrix $\mathbf{D}$ and the tensor $\bar{\mathcal{E}}$.

2) Construction of $\mathbf{D}, \bar{\mathcal{E}}$*:* The steps will involve

- Designing the sizes of the tiles for the product tensor $\bar{\mathcal{E}} \times_1 \mathbf{D}$, and placing them.
- Creating and filling the non-zero tiles in $\mathbf{D}$ and $\bar{\mathcal{E}}$.
- Placing the filled tiles in $\mathbf{D}$ and in $\bar{\mathcal{E}}$

[1]Note that $\bar{\mathcal{S}}_n = \bar{\mathcal{S}}_\mathcal{P}$ for any $n \in \mathcal{P}$. Furthermore, the term *tile* will be used interchangeably to represent both $\bar{\mathcal{S}}_\mathcal{P}$ and $\mathcal{P}$.

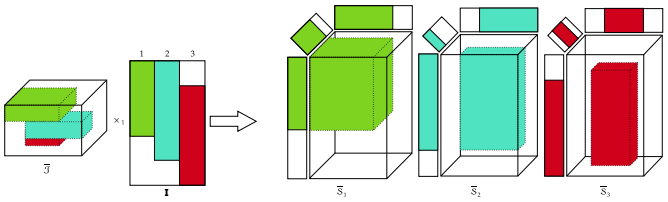

Fig. 2. The figure on the left illustrates the support constraints $\mathbf{I}$ and $\bar{\mathcal{J}}$ on $\mathbf{D}$ and $\bar{\mathcal{E}}$ respectively. The constraints $\mathbf{I}(:,1)$ and $\bar{\mathcal{J}}(1,:,:)$ on the columns and rows of $\mathbf{D}$ and $\bar{\mathcal{E}}$ respectively are colored green, $\mathbf{I}(:,2)$ and $\bar{\mathcal{J}}(2,:,:)$ are colored cyan and $\mathbf{I}(:,3)$ and $\bar{\mathcal{J}}(3,:,:)$ are colored red. The product of a column with a row of the same color, yields the corresponding rank-one contribution support $\bar{\mathcal{S}}_n(\mathbf{I},\bar{\mathcal{J}}), n = 1,2,3$, as described in Definition 8, and as illustrated on the right side of the figure.

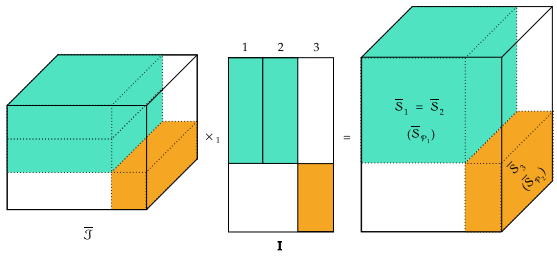

Fig. 3. This figure illustrates three different rank-one contribution supports $\bar{\mathcal{S}}_1, \bar{\mathcal{S}}_2, \bar{\mathcal{S}}_3$, where the first two fall into the same equivalence class $\bar{\mathcal{S}}_{\mathcal{P}_1} = \bar{\mathcal{S}}_1 = \bar{\mathcal{S}}_2$, while $\bar{\mathcal{S}}_{\mathcal{P}_2} = \bar{\mathcal{S}}_3$.

We first address the case of $T = 1$, which we then generalize to larger T. There will also be a clarifying example that follows the description of the design.

3) Designing $\mathbf{D}, \bar{\mathcal{E}}$ *for the case of* $T = 1$*:*

a) Step 1: Designing the sizes of the tiles for the $K \times P_1 \times \ldots \times P_L$ *product tensor* $\bar{\mathcal{E}} \times_1 \mathbf{D}$*, and placing the tiles:* We initially partition the set of equivalent classes $\mathcal{C}$ (cf. Definition 10) into the following set of equivalence classes

$$\begin{aligned}
&\mathcal{C} \triangleq \{\mathcal{P}_{\mathcal{Q}_i} | \mathcal{R}_{\mathcal{P}_{\mathcal{Q}_i}} = [1 + (k-1)\Delta : \min(k\Delta, K)],\\
&\mathcal{C}_{\mathcal{P}_{\mathcal{Q}_i}} = \prod_{\ell\in[L]} [1] \cup (\mathbb{1}(\ell \in \mathcal{Q}_i) \times [1 + (j_\ell - 1)\Lambda_\ell :\\
&\min(j_\ell\Lambda_\ell, P_\ell)]),\\
&\forall (k, j_1, \ldots, j_L) \in [\lceil\frac{K}{\Delta}\rceil] \times [\lceil\frac{P_1}{\Lambda_1}\rceil] \times \ldots \times [\lceil\frac{P_L}{\Lambda_L}\rceil],\\
&\mathcal{Q}_i = \{i_1, \ldots, i_\Gamma\} \subseteq [L], \ |\mathcal{Q}_i| = \Gamma, \ \forall i \in [\binom{L}{\Gamma}]\}
\end{aligned} \tag{48}$$

where $\mathcal{R}_{\mathcal{P}_{\mathcal{Q}_i}}$ are the indices of the corresponding columns of $\mathbf{I}$, $\mathcal{C}_{\mathcal{P}_{\mathcal{Q}_i}}$ are the indices of the corresponding rows of $\bar{\mathcal{J}}$, which are defined in Definition 11, and j_ℓ is the corresponding index to the ℓ^{th} subfunction in tensor $\bar{\mathcal{F}}$. To cover all the possibilities for our tensor-based tiling problem, we first consider the broader scenario where $P_\ell > \Lambda_\ell, \forall \ell \in [L]$ and find the general form for the required number of servers N_{opt} as the sum of the total number of tiles with different maximum representative rank-one supports. We then consider the special case where $P_\ell = \Lambda_\ell$ and evaluate the required number of servers similarly.

To decompose the high-dimensional tiles in this problem, we consider the multilinear SVD approach, detailed in Appendix B, which is based on the matrix SVD on mode-1 unfolding of tensor $\bar{\mathcal{F}} = \bar{\mathcal{E}} \times_1 \mathbf{D}$, described as

$$\bar{\mathcal{F}}_{(1)} \in \mathbb{R}^{K \times (P_1 \dots P_L)} \ .$$

We then use the rank properties of the matrix unfolding of a tensor, where we have

$$\text{rank}(\bar{\mathcal{F}}) = \text{rank}(\bar{\mathcal{F}}_{(1)}) \ .$$

Despite having L subfunctions, the computation cost Γ restricts the servers to compute up to Γ subfunctions, which is equivalent to having Γ-dimensional tiles with some combinatorial intersections, described as subtiles. We, therefore, need to find such subtiles and assign them to only one of the existing tiles with the corresponding indices $\mathcal{C}_{\mathcal{P}_{\mathcal{Q}_i}}$ in different hyperplanes corresponding to each $\mathcal{Q}_i$ to ensure disjoint support assumption for SVD-decomposition. To that end, we consider two disjoint sets of tiles $\mathcal{P}'$ and $\mathcal{P}''$ such that

$$\mathcal{C}_{\mathcal{P}_{\mathcal{Q}_i}} = \mathcal{C}_{\mathcal{P}'_{\mathcal{Q}_i}} \cup \mathcal{C}_{\mathcal{P}''_{\mathcal{Q}_i}}$$

and

$$\mathcal{C}_{\mathcal{P}'_{\mathcal{Q}_i}} \cap \mathcal{C}_{\mathcal{P}''_{\mathcal{Q}_i}} = \emptyset$$

where we first consider tiles $\mathcal{P}'$ with no combinatorial intersection and represent their corresponding column indices as

$$\mathcal{C}_{\mathcal{P}'_{\mathcal{Q}_i}} = \{\mathcal{C}_{\mathcal{P}_{\mathcal{Q}_i}} \mid j_\ell > 1, \ \forall \ell \in \mathcal{Q}_i\} \ .$$

The maximum rank of each representative rank-one support for such tiles (i.e. of each tile $\mathcal{P}'$ of $\bar{\mathcal{E}} \times_1 \mathbf{D}$) from (47) and Definition 13 therefore takes the form

$$r_{\mathcal{P}'} = \tag{49}$$

$$\begin{cases} \min\Big(\Delta, \prod\limits_{\ell \in \mathcal{Q}_i} \Lambda_\ell\Big) \ , \quad \mathcal{P}' \in \mathcal{C}'_1 \ , \\ (k, j_1, j_2, \dots, j_L) \in [\lfloor \frac{K}{\Delta} \rfloor] \times \prod\limits_{\ell \in [L]} [1] \cup \Big(\mathbb{1}(\ell \in \mathcal{Q}_i) \times \\ [2 : \lfloor \frac{P_\ell}{\Lambda_\ell} \rfloor] \Big) \ , \\ \min\Big(\text{mod}(K, \Delta), \prod\limits_{\ell \in \mathcal{Q}_i} \Lambda_\ell\Big) \ , \quad \mathcal{P}' \in \mathcal{C}'_2 \ , \\ (k, j_1, j_2, \dots, j_L) \in \prod\limits_{\ell \in [L]} [1] \cup (\mathbb{1}(\ell \in \mathcal{Q}_i) \times [2 : \lfloor \frac{P_\ell}{\Lambda_\ell} \rfloor]) \ , \\ \min\Big(\Delta, \prod_{\ell} \Lambda_\ell \prod_{\ell'} \text{mod}(P_{\ell'}, \Lambda_{\ell'})\Big) \ , \quad \mathcal{P}' \in \mathcal{C}'_3 \ , \\ \ell, \ell' \in \mathcal{Q}_i : \ \Lambda_\ell | P_\ell, \ \Lambda_{\ell'} \nmid P_{\ell'} \\ (k, j_1, j_2, \dots, j_L) \in [\lfloor \frac{K}{\Delta} \rfloor] \times \prod\limits_{\ell \in [L]} [1] \cup \Big(\mathbb{1}(\ell \in \mathcal{Q}_i) \times \\ \big(\{[2 : \lceil \frac{P_\ell}{\Lambda_\ell} \rceil]\} \cup \{\mathbb{1}(\Lambda_\ell | P_\ell) \times [2 : \lfloor \frac{P_\ell}{\Lambda_\ell} \rfloor]\} \big) \Big) \ , \\ \min\Big(\text{mod}(K, \Delta), \prod_{\ell} \Lambda_\ell \prod_{\ell'} \text{mod}(P_{\ell'}, \Lambda_{\ell'})\Big) \ , \quad \mathcal{P}' \in \mathcal{C}'_4 \ , \\ \ell, \ell' \in \mathcal{Q}_i : \ \Lambda_\ell | P_\ell, \ \Lambda_{\ell'} \nmid P_{\ell'} \\ (k, j_1, j_2, \dots, j_L) \in \prod\limits_{\ell \in [L]} [1] \cup \Big(\mathbb{1}(\ell \in \mathcal{Q}_i) \times \\ \big(\{[2 : \lceil \frac{P_\ell}{\Lambda_\ell} \rceil]\} \cup \{\mathbb{1}(\Lambda_\ell | P_\ell) \times [2 : \lfloor \frac{P_\ell}{\Lambda_\ell} \rfloor]\} \big) \Big) \end{cases}$$

where the equivalence classes $\mathcal{C}'_i$, $i \in [4]$ follow Definition 10 for all possible scenarios where $\Delta | K$, $\Lambda_\ell | P_\ell$, $\Delta \nmid K$, and $\Lambda_\ell \nmid P_\ell$. We note that for the case $\Lambda_\ell | P_\ell$, we divide each Γ-dimensional space corresponding to each $\mathcal{Q}_i$, $\forall i \in [\binom{L}{\Gamma}]$ to $\prod\limits_{\ell \in \mathcal{Q}_i} \frac{P_\ell}{\Lambda_\ell}$ tiles with dimensions $\Lambda_{i_1} \times \dots \times \Lambda_{i_\Gamma}$. We do the same for the residual space for the general case $\Lambda_\ell \nmid P_\ell$.

The number of representative rank-one supports for tiles $\mathcal{P}'$ in each equivalence class is therefore

$$\begin{aligned} |\mathcal{C}'_1| &= \lfloor \frac{K}{\Delta} \rfloor \sum_{i=1}^{\binom{L}{\Gamma}} \prod_{\ell \in \mathcal{Q}_i} \big(\lfloor \frac{P_\ell}{\Lambda_\ell} \rfloor - 1 \big) \ , \\ |\mathcal{C}'_2| &= \sum_{i=1}^{\binom{L}{\Gamma}} \prod_{\ell \in \mathcal{Q}_i} \big(\lfloor \frac{P_\ell}{\Lambda_\ell} \rfloor - 1 \big) \ , \\ |\mathcal{C}'_3| &= \lfloor \frac{K}{\Delta} \rfloor \sum_{i=1}^{\binom{L}{\Gamma}} \prod_{\ell \in \mathcal{Q}_i : \ \Lambda_\ell | P_\ell} \big(\lfloor \frac{P_\ell}{\Lambda_\ell} \rfloor - 1 \big) \\ &\quad \prod_{\ell' \in \mathcal{Q}_i : \ \Lambda_{\ell'} \nmid P_{\ell'}} \big(\lceil \frac{P_{\ell'}}{\Lambda_{\ell'}} \rceil - 1 \big) \ , \\ |\mathcal{C}'_4| &= \sum_{i=1}^{\binom{L}{\Gamma}} \prod_{\ell \in \mathcal{Q}_i : \ \Lambda_\ell | P_\ell} \big(\lfloor \frac{P_\ell}{\Lambda_\ell} \rfloor - 1 \big) \\ &\quad \prod_{\ell' \in \mathcal{Q}_i : \ \Lambda_{\ell'} \nmid P_{\ell'}} \big(\lceil \frac{P_{\ell'}}{\Lambda_{\ell'}} \rceil - 1 \big) \ . \end{aligned} \tag{50}$$

We then consider tiles $\mathcal{P}''$ with combinatorial intersections, representing their corresponding column indices as

$$\mathcal{C}_{\mathcal{P}''_{\mathcal{Q}_i}} = \{\mathcal{C}_{\mathcal{P}_{\mathcal{Q}_i}} \mid \exists j_\ell = 1, \ \forall \ell \in \mathcal{Q}_i\} \ .$$

To capture the number of intersecting subtiles in each hyperplane, we define the following set.

$$\mathcal{L}_i \triangleq \{\ell \in \mathcal{Q}_i \mid j_\ell = 1\} \ . \tag{51}$$

The number of intersecting subtiles in each hyperplane is then equal to $|\mathcal{L}_i|$.

There are two variants of tiles in the family of tiles $\mathcal{P}''$, for the simple case with the assumption $\Lambda_\ell | P_\ell$ (corresponding to equivalence classes $\mathcal{C}''_1$ and $\mathcal{C}''_2$), as follows.

1) Tiles $\mathcal{P}''_{\text{main}}$ with dimensions

$$|X_i| \triangleq \prod_{\ell \in \mathcal{Q}_i} \big(\Lambda_\ell - \mathbb{1}(\ell \in \mathcal{L}_i) \big) \ , \tag{52}$$

where $\mathcal{Q}_i \in \text{Subset}([L], \Gamma), \ \forall i \in [\binom{L}{\Gamma}]$.

2) Subtiles $\mathcal{P}''_{\text{int}}$ with dimensions

$$|x_{id}| \triangleq \prod_{\ell \in \mathcal{K}_{id}} \big(\Lambda_\ell - \mathbb{1}(\ell \in \mathcal{L}_i) \big) \ , \tag{53}$$

where $\mathcal{K}_{id} \in \text{Subset}(\mathcal{Q}_i, d) \ , \forall i \in [\binom{L}{\Gamma}]$.

To utilize the disjoint support assumption for SVD-decomposition of tiles $\mathcal{P}''_{\mathcal{Q}_i}$, we need to make these tiles, which naturally have combinatorial intersections, disjoint. To that end, we need to decide how to aggregate the intersections

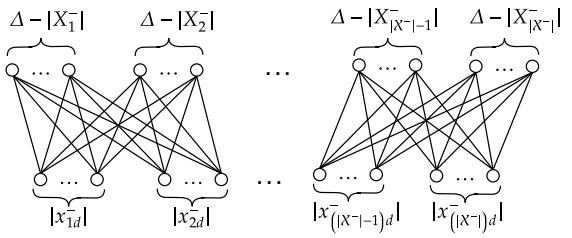

Fig. 4. The schematic for the assignment graph $G(U \cup V)$, where $d \in [\Gamma - 1]$ for each subset $|x_{nd}^-|$ of vertices in V. In this figure, we assume that each pair of consecutive subsets of vertices in U share the same subsets of V.

of such tiles, denoted by subtiles ${\mathcal{P}''}_{\text{int}}$ to some of their adjacent tiles ${\mathcal{P}''}_{\text{main}}$ to construct disjoint tiles with the minimum possible sum rank. To that end, we devise a combinatorial optimization for tile assignment based on a bipartite matching approach as follows.

Considering different possibilities for the rank of tiles in each hyperplane corresponding to each $\mathcal{Q}_i$ and assuming $\Delta | K$ (corresponding to equivalence classes ${\mathcal{C}''}_1$ and ${\mathcal{C}''}_3$), we have the following steps:

1) We first consider tiles $X_n^+ \in \mathcal{X}^+ \triangleq \{X_i \mid |X_i| \geq \Delta\}$ with the corresponding subsets $\mathcal{Q}_n^+ \in \mathcal{Q}^+ \subseteq [L]$. Since the rank of these tiles equals $\min(\Delta, |X_n^+|) = \Delta$, we assign $x_{nd}^+,\ \forall d \in [\Gamma - 1]$ to the n^{th} tile in $\mathcal{X}^+$, where x_{nd}^+ corresponds to the subsets of $\mathcal{Q}_n^+$ with cardinality d. We store the corresponding indices to set $\mathcal{T}$ of the accumulated assigned subtiles to tile X_n^+ in a set $\mathcal{P}_n^+$. We then continue doing this procedure with excluding $\cup_{m=1}^{n-1} \mathcal{P}_m^+$ from set $\mathcal{T}$ for other n^{th} tiles in $\mathcal{X}^+,\ \forall n \in [|\mathcal{X}^+|]$. We finally have
$$\mathcal{P}^+ \triangleq \cup_{n=1}^{|\mathcal{X}^+|} \mathcal{P}_n^+$$
to utilize in the next phase.
2) We next consider tiles $X_n^- \in \mathcal{X}^- \triangleq \{X_i \mid |X_i| < \Delta\}$ with the corresponding subsets $\mathcal{Q}_n^- \in \mathcal{Q}^- \subseteq [L]$.
The tile assignment procedure for these tiles is done by matching in a bipartite graph as follows.

Definition 14. (*Assignment graph.*) We devise a bipartite graph $G(U \cup V)$, demonstrated in Figure 4, comprising the set of vertices
$$U = \{(\Delta - |X_n^-|),\ \forall n \in [|\mathcal{X}^-|]\}\ ,$$
and
$$V = \{|x_{nd}^-|,\ \forall n \in [|\mathcal{X}^-|],\ d \in [\Gamma - 1]\}\ .$$
The edges of this bipartite graph are described by a family of sets $\mathcal{A}$, including $\mathcal{A}_{n,d},\ \forall n \in [|\mathcal{X}^-|],\ d \in [\Gamma - 1]$.

Our goal is to find the maximal matching, and perfect matching in the above-mentioned assignment graph if it satisfies Hall's marriage theorem, described in (58). To that end, we need to determine a system of distinct representative (SDR) for the sets $(a_{1,1}, \ldots, a_{1,\Gamma-1}, \ldots, a_{|\mathcal{X}^-|,1}, \ldots, a_{|\mathcal{X}^-|,\Gamma-1})$ with the following properties:

1) **Representative Property:**
$$a_{n,d} \in \mathcal{A}_{n,d},\ \forall n \in [|\mathcal{X}^-|],\ d \in [\Gamma - 1]\ . \tag{54}$$
2) **Distinct Property:**
$$a_{n_1,d_1} \neq a_{n_2,d_2},\ \forall (n_1, d_1) \neq (n_2, d_2)\ . \tag{55}$$

For any set
$$\mathcal{D} \subseteq \{(n, d) \mid n \in [|\mathcal{X}^-|], d \in [\Gamma - 1]\} \tag{56}$$
of indices, we have:
$$\mathcal{A}(\mathcal{D}) = \cup_{e \in \mathcal{D}} \mathcal{A}_e\ . \tag{57}$$

Hall's marriage theorem is then expressed as follows.

Lemma 2. (Hall's marriage theorem [46].) A family of finite sets $\mathcal{A}$ has an SDR if and only if it meets the condition
$$|\mathcal{A}(\mathcal{D})| \geq |\mathcal{D}|\ . \tag{58}$$

We next check the matching condition in (58). If it holds, there exists a perfect matching and we can use the Hopcroft-Corp algorithm [47], which is a graph matching algorithm that efficiently finds the maximum matching in a bipartite graph G by identifying a maximum cardinality matching, i.e., a collection of edges with the maximum possible size, where no two edges share a common endpoint. Otherwise, we first sort $|X_n^-|$s from max to min, or equivalently from $\min(\Delta - |X_n^-|)$ to $\max(\Delta - |X_n^-|),\ \forall n \in [|\mathcal{X}^-|]$. We then assign the remaining subtiles $x_{nd}^-,\ \forall d \in [\Gamma - 1]$, successively to the corresponding tiles $X_n^- \in \mathcal{X}^-$, where x_{nd}^- corresponds to the subsets $\mathcal{T} \backslash \mathcal{P}^+$ of $\mathcal{Q}_n^-$. We then continue checking Hall's marriage theorem and repeating this procedure until it is verified. We denote the result of the matching procedure by $X_{n,\text{opt}}$ for each hyperplane corresponding to $\mathcal{Q}_n^-$.

It is worth mentioning that the set of indices $\mathcal{C}_{\mathcal{P}''}$ is updated in each step of the tile assignment.

Similar to the equivalence classes $\mathcal{C}'$ in 50, for the general case $\Delta \nmid K$ (corresponding to equivalence classes ${\mathcal{C}''}_2$ and ${\mathcal{C}''}_4$), we simply substitute Δ in the above steps with $\text{mod}(K, \Delta)$ for the residual tiles.

We next consider the general case $\Lambda_\ell \nmid P_\ell$ (corresponding to equivalence classes ${\mathcal{C}''}_3$ and ${\mathcal{C}''}_4$), where we have two variants of tiles $\mathcal{P}''$ as follows.

1) Tiles ${\mathcal{P}''}_{\text{main}}$ with dimensions
$$|X_i''| \triangleq \prod_{\ell \in \mathcal{Q}_i:\ \Lambda_\ell | P_\ell} \big(\Lambda_\ell - \mathbb{1}(\ell \in \mathcal{L}_i)\big) \prod_{\ell' \in \mathcal{Q}_i:\ \Lambda_{\ell'} \nmid P_{\ell'}} \text{mod}(P_{\ell'}, \Lambda_{\ell'})\ , \tag{59}$$
where $\mathcal{Q}_i \in \text{Subset}([L], \Gamma),\ \forall i \in [\binom{L}{\Gamma}]$.
2) Subtiles ${\mathcal{P}''}_{\text{int}}$ with dimensions
$$|x_{id}''| \triangleq \prod_{\ell \in \mathcal{K}_{id}:\ \Lambda_\ell | P_\ell} \big(\Lambda_\ell - \mathbb{1}(\ell \in \mathcal{L}_i)\big)$$

$$\prod_{\ell' \in \mathcal{K}_{id}:\ \Lambda_{\ell'} \nmid P_{\ell'}} \mathrm{mod}(P_{\ell'}, \Lambda_{\ell'}) \ , \tag{60}$$

where $\mathcal{K}_{id} \in \mathrm{Subset}(\mathcal{Q}_i, d)$,$\forall i \in [\binom{L}{\Gamma}]$.

A similar procedure for matching in a bipartite graph is done with the above definitions of $|X_i''|$ and $|x_{id}''|$. We denote the result of the matching approach for this case by $|X_{i,\mathrm{opt}}''|$.

The maximum rank of each representative rank-one support for such tiles (i.e. of each tile $\mathcal{P}''$ of $\bar{\mathcal{E}} \times_1 \mathbf{D}$) from (47) and Definition 13 therefore takes the form

$$r_{\mathcal{P}''} = \tag{61}$$
$$\begin{cases} \min\Big(\Delta, |X_{i,\mathrm{opt}}|\Big) \ , \quad \mathcal{P}'' \in \mathcal{C}''_1 \ , \\ (k, j_1, j_2, \ldots, j_L) \in [\lfloor \frac{K}{\Delta} \rfloor] \times \prod_{\ell \in [L]} [1] \cup \Big(\mathbb{1}(\ell \in \mathcal{Q}_i \backslash \mathcal{L}_i) \times \\ [2 : \lfloor \frac{P_\ell}{\Lambda_\ell} \rfloor] \Big) \ , \\ \min\Big(\mathrm{mod}(K, \Delta), |X_{i,\mathrm{opt}}|\Big) \ , \quad \mathcal{P}'' \in \mathcal{C}''_2 \ , \\ (k, j_1, j_2, \ldots, j_L) \in \prod_{\ell \in [L]} [1] \cup (\mathbb{1}(\ell \in \mathcal{Q}_i \backslash \mathcal{L}_i) \times [2 : \lfloor \frac{P_\ell}{\Lambda_\ell} \rfloor]) \ , \\ \min\Big(\Delta, |X''_{i,\mathrm{opt}}|\Big) \ , \quad \mathcal{P}'' \in \mathcal{C}''_3 \ , \\ (k, j_1, j_2, \ldots, j_L) \in [\lfloor \frac{K}{\Delta} \rfloor] \times \prod_{\ell \in [L]} [1] \cup \Big(\mathbb{1}(\ell \in \mathcal{Q}_i \backslash \mathcal{L}_i) \times \\ \big(\{ [2 : \lceil \frac{P_\ell}{\Lambda_\ell} \rceil] \} \cup \{ \mathbb{1}(\Lambda_\ell | P_\ell) \times [2 : \lfloor \frac{P_\ell}{\Lambda_\ell} \rfloor] \} \big) \Big) \ , \\ \min\Big(\mathrm{mod}(K, \Delta), |X''_{i,\mathrm{opt}}|\Big) \ , \quad \mathcal{P}'' \in \mathcal{C}''_4 \ , \\ (k, j_1, j_2, \ldots, j_L) \in \prod_{\ell \in [L]} [1] \cup \Big(\mathbb{1}(\ell \in \mathcal{Q}_i \backslash \mathcal{L}_i) \times \\ \big(\{ [2 : \lceil \frac{P_\ell}{\Lambda_\ell} \rceil] \} \cup \{ \mathbb{1}(\Lambda_\ell | P_\ell) \times [2 : \lfloor \frac{P_\ell}{\Lambda_\ell} \rfloor] \} \big) \Big) \ . \end{cases}$$

The number of representative rank-one supports for tiles $\mathcal{P}''$ in each class is then

$$\begin{aligned} |\mathcal{C}''_1| &= \lfloor \frac{K}{\Delta} \rfloor \sum_{i=1}^{\binom{L}{\Gamma}} \prod_{\ell \in \mathcal{Q}_i \backslash \mathcal{L}_i} \big(\lfloor \frac{P_\ell}{\Lambda_\ell} \rfloor - 1 \big) \ , \\ |\mathcal{C}''_2| &= \sum_{i=1}^{\binom{L}{\Gamma}} \prod_{\ell \in \mathcal{Q}_i \backslash \mathcal{L}_i} \big(\lfloor \frac{P_\ell}{\Lambda_\ell} \rfloor - 1 \big) \ , \\ |\mathcal{C}''_3| &= \lfloor \frac{K}{\Delta} \rfloor \sum_{i=1}^{\binom{L}{\Gamma}} \prod_{\ell \in \mathcal{Q}_i \backslash \mathcal{L}_i:\ \Lambda_\ell | P_\ell} \big(\lfloor \frac{P_\ell}{\Lambda_\ell} \rfloor - 1 \big) \\ &\qquad \prod_{\ell' \in \mathcal{Q}_i \backslash \mathcal{L}_i:\ \Lambda_{\ell'} \nmid P_{\ell'}} \big(\lceil \frac{P_{\ell'}}{\Lambda_{\ell'}} \rceil - 1 \big) \ , \\ |\mathcal{C}''_4| &= \sum_{i=1}^{\binom{L}{\Gamma}} \prod_{\ell \in \mathcal{Q}_i \backslash \mathcal{L}_i:\ \Lambda_\ell | P_\ell} \big(\lfloor \frac{P_\ell}{\Lambda_\ell} \rfloor - 1 \big) \\ &\qquad \prod_{\ell' \in \mathcal{Q}_i \backslash \mathcal{L}_i:\ \Lambda_{\ell'} \nmid P_{\ell'}} \big(\lceil \frac{P_{\ell'}}{\Lambda_{\ell'}} \rceil - 1 \big) \ . \end{aligned} \tag{62}$$

Having established the classes and the corresponding (position of the) tiles of $\bar{\mathcal{F}}$, we proceed to fill (and crop) the $\bar{\mathcal{F}}$ tiles, as follows:

$$\bar{\mathcal{F}}_{\mathcal{P}} \triangleq (\bar{\mathcal{F}} \odot \bar{\mathcal{S}}_{\mathcal{P}})(\mathcal{R}_{\mathcal{P}}, \mathcal{C}_{\mathcal{P}}), \ \forall \mathcal{P} \in \cup_{i=1}^{4} \mathcal{C}'_i \cup \mathcal{C}''_i \ . \tag{63}$$

The first step $\bar{\mathcal{F}} \odot \bar{\mathcal{S}}_{\mathcal{P}}$ simply fills up $\bar{\mathcal{S}}_{\mathcal{P}}$ with the corresponding entries of $\bar{\mathcal{F}}$, and the second step $(\bar{\mathcal{F}} \odot \bar{\mathcal{S}}_{\mathcal{P}})(\mathcal{R}_{\mathcal{P}}, \mathcal{C}_{\mathcal{P}})$ crops these[1].

b) Step 2: Creating and filling the non-zero tiles in $\mathbf{D}$ *and* $\bar{\mathcal{E}}$*:* This step starts with the SVD decomposition as (described in Section B) of the cropped tile $\bar{\mathcal{F}}_{\mathcal{P}}$ where this decomposition takes the form

$$\bar{\mathcal{F}}_{\mathcal{P}} = \bar{\mathcal{E}}_{\mathcal{P}} \times_1 \mathbf{D}_{\mathcal{P}} \ , \tag{64}$$

where $\mathbf{D}_{\mathcal{P}} \in \mathbb{R}^{|\mathcal{R}_{\mathcal{P}}| \times r_{\mathcal{P}}}, \bar{\mathcal{E}}_{\mathcal{P}} \in \mathbb{R}^{r_{\mathcal{P}} \times |\mathcal{C}_{\mathcal{P}}|}$. In particular, $\bar{\mathcal{F}}, \mathbf{D}$, and $\bar{\mathcal{E}}$ are associated to $\bar{\mathcal{T}}$, $\mathbf{U}^{(1)}$, and $\bar{\mathcal{S}}$ in all complete SVD decomposition of (40). Naturally, $\mathrm{rank}(\bar{\mathcal{F}}_{\mathcal{P}}) \leq r_{\mathcal{P}}$.

c) Step 3: Placing the filled cropped tiles $\mathbf{D}_{\mathcal{P}}$ *and* $\bar{\mathcal{E}}_{\mathcal{P}}$ *in* $\mathbf{D}$ *and* $\bar{\mathcal{E}}$*:* Let

$$\cup_{i=1}^{4} \mathcal{C}'_i \cup \mathcal{C}''_i = \{\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_m\}, \ m \in \mathbb{N} \tag{65}$$

describe the enumeration we give to each tile. Then the position that each cropped tile takes inside $\mathbf{D}$, is given by

$$\mathcal{R}_{\mathcal{P}_j}, [\sum_{i=1}^{j-1} T \lceil \frac{r_{\mathcal{P}_i}}{T} \rceil + 1 : \sum_{i=1}^{j} T \lceil \frac{r_{\mathcal{P}_i}}{T} \rceil], \ \forall \mathcal{P}_j \in \cup_{i=1}^{4} \mathcal{C}'_i \cup \mathcal{C}''_i, \tag{66}$$

and the position of each cropped tile in $\bar{\mathcal{E}}$ is given by

$$[\sum_{i=1}^{j-1} T \lceil \frac{r_{\mathcal{P}_i}}{T} \rceil + 1 : \sum_{i=1}^{j} T \lceil \frac{r_{\mathcal{P}_i}}{T} \rceil], \ \mathcal{C}_{\mathcal{P}_j}, \ \forall \mathcal{P}_j \in \cup_{i=1}^{4} \mathcal{C}'_i \cup \mathcal{C}''_i. \tag{67}$$

In particular, for $T = 1$ this yields

$$\mathbf{D}(\mathcal{R}_{\mathcal{P}_j}, [\sum_{i=1}^{j-1} r_{\mathcal{P}_i} + 1, \sum_{i=1}^{j} r_{\mathcal{P}_i}]) = \mathbf{D}_{\mathcal{P}_j} \tag{68}$$

and

$$\bar{\mathcal{E}}([\sum_{i=1}^{j-1} r_{\mathcal{P}_i} + 1, \sum_{i=1}^{j} r_{\mathcal{P}_i}], \mathcal{C}_{\mathcal{P}_j}) = \bar{\mathcal{E}}_{\mathcal{P}_j} \tag{69}$$

while naturally the remaining non-assigned elements of $\mathbf{D}$ and $\bar{\mathcal{E}}$ are zero. Finally, as one can readily verify, the above design corresponds to

$$\begin{aligned} N &= \sum_{i \in [4]} \sum_{\mathcal{P}' \in \mathcal{C}'_i} r_{\mathcal{P}'} |\mathcal{C}'_i| + \sum_{i \in [4]} \sum_{\mathcal{P}'' \in \mathcal{C}''_i} r_{\mathcal{P}''} |\mathcal{C}''_i| \\ &= \lfloor \frac{K}{\Delta} \rfloor \sum_{i=1}^{\binom{L}{\Gamma}} \min\Big(\Delta, \prod_{\ell \in \mathcal{Q}_i} \Lambda_\ell\Big) \prod_{\ell \in \mathcal{Q}_i} \big(\lfloor \frac{P_\ell}{\Lambda_\ell} \rfloor - 1 \big) \\ &\quad + \sum_{i=1}^{\binom{L}{\Gamma}} \min\Big(\mathrm{mod}(K, \Delta), \prod_{\ell \in \mathcal{Q}_i} \Lambda_\ell\Big) \prod_{\ell \in \mathcal{Q}_i} \big(\lfloor \frac{P_\ell}{\Lambda_\ell} \rfloor - 1 \big) \end{aligned}$$

[1] We here see a small distinction between the previously uncropped tiles, and the tiles that are cropped here. These cropped tiles will be the outcomes of an SVD decomposition of subtensors of $\bar{\mathcal{F}}$, yielding our SVD-based factorization.

$$
\begin{aligned}
&+\lfloor\frac{K}{\Delta}\rfloor\sum_{i=1}^{\binom{L}{\Gamma}}\min\Big(\Delta,\prod_{\substack{\ell\\ \ell,\ell'\in\mathcal{Q}_i:\ \Lambda_\ell|P_\ell,\ \Lambda_{\ell'}\nmid P_{\ell'}}}\Lambda_\ell\prod_{\ell'}\mathrm{mod}(P_{\ell'},\Lambda_{\ell'})\Big)\\
&\prod_{\ell\in\mathcal{Q}_i:\ \Lambda_\ell|P_\ell}(\lfloor\frac{P_\ell}{\Lambda_\ell}\rfloor-1)\prod_{\ell'\in\mathcal{Q}_i:\ \Lambda_{\ell'}\nmid P_{\ell'}}(\lceil\frac{P_{\ell'}}{\Lambda_{\ell'}}\rceil-1)\\
&+\sum_{i=1}^{\binom{L}{\Gamma}}\min\Big(\mathrm{mod}(K,\Delta),\prod_{\substack{\ell\\ \ell,\ell'\in\mathcal{Q}_i:\ \Lambda_\ell|P_\ell,\ \Lambda_{\ell'}\nmid P_{\ell'}}}\Lambda_\ell\prod_{\ell'}\mathrm{mod}(P_{\ell'},\Lambda_{\ell'})\Big)\\
&\prod_{\ell\in\mathcal{Q}_i:\ \Lambda_\ell|P_\ell}(\lfloor\frac{P_\ell}{\Lambda_\ell}\rfloor-1)\prod_{\ell'\in\mathcal{Q}_i:\ \Lambda_{\ell'}\nmid P_{\ell'}}(\lceil\frac{P_{\ell'}}{\Lambda_{\ell'}}\rceil-1)\\
&+\lfloor\frac{K}{\Delta}\rfloor\sum_{i=1}^{\binom{L}{\Gamma}}\min\Big(\Delta,|X_{i,\mathrm{opt}}|\Big)\prod_{\ell\in\mathcal{Q}_i\setminus\mathcal{L}_i}(\lfloor\frac{P_\ell}{\Lambda_\ell}\rfloor-1)\\
&+\sum_{i=1}^{\binom{L}{\Gamma}}\min\Big(\mathrm{mod}(K,\Delta),|X_{i,\mathrm{opt}}|\Big)\prod_{\ell\in\mathcal{Q}_i\setminus\mathcal{L}_i}(\lfloor\frac{P_\ell}{\Lambda_\ell}\rfloor-1)\\
&+\lfloor\frac{K}{\Delta}\rfloor\sum_{i=1}^{\binom{L}{\Gamma}}\min\Big(\Delta,|X''_{i,\mathrm{opt}}|\Big)\prod_{\ell\in\mathcal{Q}_i\setminus\mathcal{L}_i:\ \Lambda_\ell|P_\ell}(\lfloor\frac{P_\ell}{\Lambda_\ell}\rfloor-1)\\
&\prod_{\ell'\in\mathcal{Q}_i\setminus\mathcal{L}_i:\ \Lambda_{\ell'}\nmid P_{\ell'}}(\lceil\frac{P_{\ell'}}{\Lambda_{\ell'}}\rceil-1)\\
&+\sum_{i=1}^{\binom{L}{\Gamma}}\min\Big(\mathrm{mod}(K,\Delta),|X''_{i,\mathrm{opt}}|\Big)\prod_{\ell\in\mathcal{Q}_i\setminus\mathcal{L}_i:\ \Lambda_\ell|P_\ell}(\lfloor\frac{P_\ell}{\Lambda_\ell}\rfloor-1)\\
&\prod_{\ell'\in\mathcal{Q}_i\setminus\mathcal{L}_i:\ \Lambda_{\ell'}\nmid P_{\ell'}}(\lceil\frac{P_{\ell'}}{\Lambda_{\ell'}}\rceil-1)\ . \qquad (70)
\end{aligned}
$$

corresponding to Theorem 3 for the case of $T=1$.

Similarly, when $P_\ell=\Lambda_\ell$, which means containing only one tile $\mathcal{P}''$ with $|C''_i|=1$ in each hyperplane including $\mathcal{Q}_i$, we have:

$$
\begin{aligned}
N&=\sum_{i\in[4]}\sum_{\mathcal{P}''\in\mathcal{C}''_i}r_{\mathcal{P}''}|\mathcal{C}''_i|\\
&=\lfloor\frac{K}{\Delta}\rfloor\sum_{i=1}^{\binom{L}{\Gamma}}\min\Big(\Delta,|X_{i,\mathrm{opt}}|\Big)\\
&\quad+\sum_{i=1}^{\binom{L}{\Gamma}}\min\Big(\mathrm{mod}(K,\Delta),|X_{i,\mathrm{opt}}|\Big)\\
&\quad+\lfloor\frac{K}{\Delta}\rfloor\sum_{i=1}^{\binom{L}{\Gamma}}\min\Big(\Delta,|X''_{i,\mathrm{opt}}|\Big)\\
&\quad+\sum_{i=1}^{\binom{L}{\Gamma}}\min\Big(\mathrm{mod}(K,\Delta),|X''_{i,\mathrm{opt}}|\Big)\ . \qquad (71)
\end{aligned}
$$

For the general multi-shot case, we have the following results.

Theorem 3. *The optimal achievable rate of the lossless $K,N,T,L,P_\ell>\Lambda_\ell,\Gamma,\Delta,\Lambda_\ell,\ell\in\mathcal{Q}_i$ distributed computing system for the general settings in which $\Delta\nmid K$, $\Lambda_\ell\nmid P_\ell$ takes the form $C=K/N_{opt}$, where*

$$
\begin{aligned}
N_{opt}\le&\lfloor\frac{K}{\Delta}\rfloor\sum_{i=1}^{\binom{L}{\Gamma}}\Big\lceil\frac{\min\Big(\Delta,\prod\limits_{\ell\in\mathcal{Q}_i}\Lambda_\ell\Big)}{T}\Big\rceil\prod_{\ell\in\mathcal{Q}_i}(\lfloor\frac{P_\ell}{\Lambda_\ell}\rfloor-1)\\
&+\sum_{i=1}^{\binom{L}{\Gamma}}\Big\lceil\frac{\min\Big(\mathrm{mod}(K,\Delta),\prod\limits_{\ell\in\mathcal{Q}_i}\Lambda_\ell\Big)}{T}\Big\rceil\prod_{\ell\in\mathcal{Q}_i}(\lfloor\frac{P_\ell}{\Lambda_\ell}\rfloor-1)\\
&+\lfloor\frac{K}{\Delta}\rfloor\sum_{i=1}^{\binom{L}{\Gamma}}\Big\lceil\frac{\min\Big(\Delta,\prod_\ell\Lambda_\ell\prod_{\ell'}\mathrm{mod}(P_{\ell'},\Lambda_{\ell'})\Big)}{T}\Big\rceil\\
&\prod_{\ell\in\mathcal{Q}_i:\ \Lambda_\ell|P_\ell}(\lfloor\frac{P_\ell}{\Lambda_\ell}\rfloor-1)\prod_{\ell'\in\mathcal{Q}_i:\ \Lambda_{\ell'}\nmid P_{\ell'}}(\lceil\frac{P_{\ell'}}{\Lambda_{\ell'}}\rceil-1)\\
&+\sum_{i=1}^{\binom{L}{\Gamma}}\Big\lceil\frac{\min\Big(\mathrm{mod}(K,\Delta),\prod_\ell\Lambda_\ell\prod_{\ell'}\mathrm{mod}(P_{\ell'},\Lambda_{\ell'})\Big)}{T}\Big\rceil\\
&\prod_{\ell\in\mathcal{Q}_i:\ \Lambda_\ell|P_\ell}(\lfloor\frac{P_\ell}{\Lambda_\ell}\rfloor-1)\prod_{\ell'\in\mathcal{Q}_i:\ \Lambda_{\ell'}\nmid P_{\ell'}}(\lceil\frac{P_{\ell'}}{\Lambda_{\ell'}}\rceil-1)\\
&+\lfloor\frac{K}{\Delta}\rfloor\sum_{i=1}^{\binom{L}{\Gamma}}\Big\lceil\frac{\min\Big(\Delta,|X_{i,opt}|\Big)}{T}\Big\rceil\prod_{\ell\in\mathcal{Q}_i\setminus\mathcal{L}_i}(\lfloor\frac{P_\ell}{\Lambda_\ell}\rfloor-1)\\
&+\sum_{i=1}^{\binom{L}{\Gamma}}\Big\lceil\frac{\min\Big(\mathrm{mod}(K,\Delta),|X_{i,opt}|\Big)}{T}\Big\rceil\prod_{\ell\in\mathcal{Q}_i\setminus\mathcal{L}_i}(\lfloor\frac{P_\ell}{\Lambda_\ell}\rfloor-1)\\
&+\lfloor\frac{K}{\Delta}\rfloor\sum_{i=1}^{\binom{L}{\Gamma}}\Big\lceil\frac{\min\Big(\Delta,|X''_{i,opt}|\Big)}{T}\Big\rceil\prod_{\ell\in\mathcal{Q}_i\setminus\mathcal{L}_i:\ \Lambda_\ell|P_\ell}(\lfloor\frac{P_\ell}{\Lambda_\ell}\rfloor\\
&-1)\prod_{\ell'\in\mathcal{Q}_i\setminus\mathcal{L}_i:\ \Lambda_{\ell'}\nmid P_{\ell'}}(\lceil\frac{P_{\ell'}}{\Lambda_{\ell'}}\rceil-1)\\
&+\sum_{i=1}^{\binom{L}{\Gamma}}\Big\lceil\frac{\min\Big(\mathrm{mod}(K,\Delta),|X''_{i,opt}|\Big)}{T}\Big\rceil\prod_{\ell\in\mathcal{Q}_i\setminus\mathcal{L}_i:\ \Lambda_\ell|P_\ell}(\lfloor\frac{P_\ell}{\Lambda_\ell}\rfloor\\
&-1)\prod_{\ell'\in\mathcal{Q}_i\setminus\mathcal{L}_i:\ \Lambda_{\ell'}\nmid P_{\ell'}}(\lceil\frac{P_{\ell'}}{\Lambda_{\ell'}}\rceil-1) \qquad (72)
\end{aligned}
$$

where $\mathcal{L}_i\triangleq\{\ell\in\mathcal{Q}_i\mid j_\ell=1\}$, and j_ℓ is the index in tensor $\bar{\mathcal{F}}$ corresponding to the ℓ^{th} basis subfunction.

Theorem 4. *The optimal achievable rate of the lossless $K,N,T,L,P_\ell=\Lambda_\ell,\Gamma,\Delta,\Lambda_\ell,\ell\in\mathcal{Q}_i$ distributed computing system for the general settings in which $\Delta\nmid K$ takes the form $C=K/N_{opt}$, where*

$$
\begin{aligned}
N_{opt}\le&\lfloor\frac{K}{\Delta}\rfloor\sum_{i=1}^{\binom{L}{\Gamma}}\Big\lceil\frac{\min\Big(\Delta,|X_{i,opt}|\Big)}{T}\Big\rceil\\
&+\sum_{i=1}^{\binom{L}{\Gamma}}\Big\lceil\frac{\min\Big(\mathrm{mod}(K,\Delta),|X_{i,opt}|\Big)}{T}\Big\rceil
\end{aligned}
$$

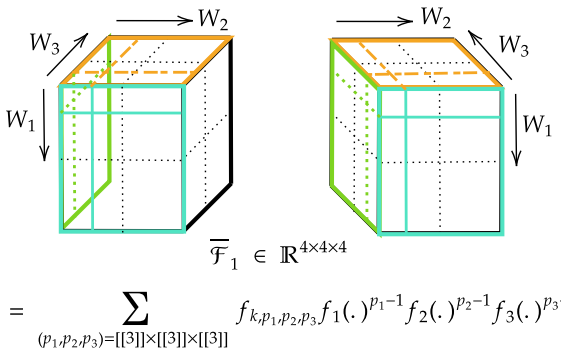

Fig. 5. Corresponding to Example 1, this figure illustrates the partitioning of $\bar{\mathcal{F}}_1$ into two dimensional tiles with some combinatorially intersecting parts.

$$+ \lfloor \frac{K}{\Delta} \rfloor \sum_{i=1}^{\binom{L}{\Gamma}} \left\lceil \frac{\min\left(\Delta, |X''_{i,opt|}\right)}{T} \right\rceil + \sum_{i=1}^{\binom{L}{\Gamma}} \left\lceil \frac{\min\left(\mathrm{mod}(K,\Delta), |X''_{i,opt|}\right)}{T} \right\rceil . \quad (73)$$

D. Example

We provide a simple yet nontrivial example to clarify the protocol for the multi-user distributed computing settings proposed in this work.

Example 1. Let consider a distributed settings in which $K = 6, \Delta = 6, L = 3, \Gamma = 2, P_\ell = 6, \Lambda_\ell = 3, \forall \ell \in [L]$. The demanded tensor of user 1 denoted by $\bar{\mathcal{F}}_1$ reveals the following details, as demonstrated in Figure 5. Following the protocol detailed in Appendix C, we have the following subsets of subfunctions with cardinality $\Gamma = 2$:

- $\mathcal{Q}_1 = \{1, 2\}$.
- $\mathcal{Q}_2 = \{1, 3\}$.
- $\mathcal{Q}_3 = \{2, 3\}$.

We next divide each Γ-dimensional space corresponding to each $\mathcal{Q}_i$, $\forall i \in [3]$ to $\prod_{\ell \in \mathcal{Q}_i} \frac{P_\ell}{\Lambda_\ell} = 4$ areas with dimensions 3×3.

We first consider the nonintersecting tiles with dimensions $\prod_{\ell \in \mathcal{Q}_i} [1 + \Lambda_\ell : 2\Lambda_\ell] = [4 : 6] \times [4 : 6]$ in each hyperplane as

- $\mathcal{Q}_1$: Corresponding to $j_1 = j_2 = 2$.
- $\mathcal{Q}_2$: Corresponding to $j_1 = j_3 = 2$.
- $\mathcal{Q}_3$: Corresponding to $j_2 = j_3 = 2$.

For the intersecting tiles, because we could assign the multiplicative terms including powers of one of W_ℓs, to more than one $\mathcal{Q}_i$, we have the following possibilities:

- $\mathcal{Q}_1$, $\mathcal{Q}_2$, and $\mathcal{Q}_3$, $(j_1, j_2, j_3) = (1, 1, 1)$:
  We have a main tile $X_1 :\ [2 : \Lambda_1] \times [2 : \Lambda_2]$ in $\mathcal{Q}_1$, a main tile $X_2 :\ [2 : \Lambda_1] \times [2 : \Lambda_3]$ in $\mathcal{Q}_2$, and a main tile $X_3 :\ [2 : \Lambda_2] \times [2 : \Lambda_3]$ in $\mathcal{Q}_3$.
  1) $(j_1, j_2) = (1, 1)$ and $(j_1, j_3) = (1, 1)$:
     There is an intersecting subtile $x_{11} :\ [2 : \Lambda_1] \times [1]$, where we need to decide to assign it to one of main tiles X_1 and X_2.
  2) $(j_1, j_2) = (1, 1)$ and $(j_2, j_3) = (1, 1)$:
     There is an intersecting subtile $x_{21} :\ [1] \times [2 : \Lambda_2]$, where we need to decide to assign it to one of main tiles X_1 and X_3.
  3) $(j_1, j_3) = (1, 1)$ and $(j_2, j_3) = (1, 1)$:
     There is an intersecting subtile $x_{31} :\ [1] \times [2 : \Lambda_3]$, where we need to decide to assign it to one of main tiles X_2 and X_3.

  Since $|X_1| = |X_2| = |X_3| = 4 < \Delta$, we directly use the bipartite matching procedure with a bipartite graph $G = (U \cup V)$, where $U = \{\Delta - |X_i|\}$ and $V = \{|x_{id}|\}$, $\forall i \in [3], d \in [1]$, which is depicted in Figure 6. As we can see, there are four possible edges for each node in V, meaning that Hall's marriage theorem is satisfied and there exists a perfect matching. We consider a possible solution as follows.
  We denote the tiles with $X_{i,\text{opt}}$ after the matching procedure. We will then have tiles $X_{1,\text{opt}} :\ [2 : \Lambda_1] \times [1 : \Lambda_2]$, $X_{2,\text{opt}} :\ [1 : \Lambda_1] \times [2 : \Lambda_3]$, $X_{3,\text{opt}} :\ [2 : \Lambda_2] \times [1 : \Lambda_3]$.

- $\mathcal{Q}_1$ and $\mathcal{Q}_2$, $(j_1, j_2) = (2, 1)$ and $(j_1, j_3) = (2, 1)$:
  We have a main tile $X_1 :\ [1+\Lambda_1 : 2\Lambda_1] \times [2 : \Lambda_2]$ in $\mathcal{Q}_1$ and a main tile $X_2 :\ [1+\Lambda_1 : 2\Lambda_1] \times [2 : \Lambda_3]$ in $\mathcal{Q}_2$. There is also an intersecting subtile $x_{11} :\ [1 + \Lambda_1 : 2\Lambda_1] \times [1]$, where we need to decide to assign it to one of main tiles X_1 and X_2.
  Since $|X_1| = |X_2| = 6 = \Delta$, we could assign x_{11} to any of X_1 and X_2. We choose to assign it to X_1. We then will have tile $X_{1,\text{opt}} :\ [1 : \Lambda_1] \times [1 : \Lambda_2]$.

- $\mathcal{Q}_1$ and $\mathcal{Q}_3$, $(j_1, j_2) = (1, 2)$ and $(j_2, j_3) = (2, 1)$:
  We have a main tile $X_1 :\ [2 : \Lambda_1] \times [1+\Lambda_2 : 2\Lambda_2]$ in $\mathcal{Q}_1$ and a main tile $X_3 :\ [1+\Lambda_2 : 2\Lambda_2] \times [2 : \Lambda_3]$ in $\mathcal{Q}_2$. There is also an intersecting subtile $x_{11} :\ [1] \times [1 + \Lambda_2 : 2\Lambda_2]$ in $\mathcal{Q}_1$ or equivalently $[1 + \Lambda_2 : 2\Lambda_2] \times [1]$ in $\mathcal{Q}_3$, where we need to decide to assign it to one of main tiles X_1 and X_3.
  Since $|X_1| = |X_3| = 6 = \Delta$, we could assign x_{11} to any of X_1 and X_3. We choose to assign it to X_1. We then will have tile $X_{1,\text{opt}} :\ [1 : \Lambda_1] \times [1 + \Lambda_2 : 2\Lambda_2]$.

- $\mathcal{Q}_2$ and $\mathcal{Q}_3$, $(j_1, j_3) = (1, 2)$ and $(j_2, j_3) = (1, 2)$:
  We have a main tile $X_2 :\ [2 : \Lambda_1] \times [1+\Lambda_3 : 2\Lambda_3]$ in $\mathcal{Q}_2$ and a main tile $X_3 :\ [2 : \Lambda_2] \times [1+\Lambda_3 : 2\Lambda_3]$ in $\mathcal{Q}_3$. There is also an intersecting subtile $x_{11} :\ [1] \times [1 + \Lambda_3 : 2\Lambda_3]$, where we need to decide to assign it to one of main tiles X_2 and X_3.
  Since $|X_2| = |X_3| = 6 = \Delta$, we could assign x_{11} to any of X_2 and X_3. We choose to assign it to X_2. We then will have tile $X_{2,\text{opt}} :\ [1 : \Lambda_1] \times [1 + \Lambda_3 : 2\Lambda_3]$.

We also note that the tiles built after the assignment and

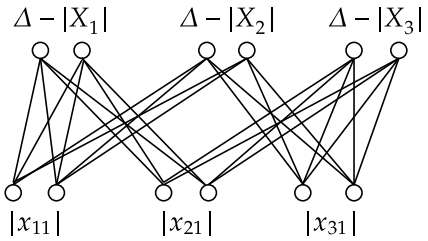

Fig. 6. Corresponding to Example 1, this figure illustrates the bipartite graph used for the assignment of intersecting subtiles to the tiles.

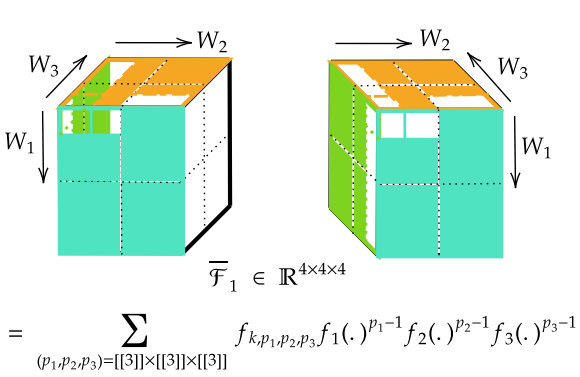

Fig. 7. The tiling result for Example 1. The same colors represent the tiles in the same plane, corresponding to the same subset $\mathcal{Q}_i$ of L subfunctions with cardinality Γ, $\forall i \in [\binom{L}{\Gamma}]$.

matching process are disjoint and we could easily apply the multilinear SVD approach described in Appendix B to $\bar{\mathcal{F}}$. Finally, the tiles in $\bar{\mathcal{F}}_1$ are shaped as shown in Figure 7.

Considering the above steps for all 6 users, the total number of required servers for this problem based on Corollary 1 is

$$\begin{aligned} N &= \frac{K}{\Delta}\sum_{i=1}^{3}\Big(\min(\Delta, \prod_{\ell\in\mathcal{Q}_i}\Lambda_\ell)\prod_{\ell\in\mathcal{Q}_i}(\frac{P_\ell}{\Lambda_\ell}-1) \\ &\quad + \min\big(\Delta, |X_{i,\text{opt}}|\big)\prod_{\ell\in\mathcal{Q}_i\setminus\mathcal{L}_i}(\frac{P_\ell}{\Lambda_\ell}-1)\Big) \\ &= 3\min(6,9)+3\min(6,6)+3\min(6,6)+3\min(6,9) \\ &= 4\times3\times6 = 72\ . \end{aligned} \tag{74}$$

E. Proof of Proposition 1

The total computation cost of non-linearly separable settings proposed in this work is obtained as

$$\Theta_1 \overset{(a)}{\leq} N\big(\Gamma+\Lambda_1+\ldots+\Lambda_L\big) \overset{(b)}{=} N\big(L(\Lambda+1)\big)$$

where (a) holds because we have N servers each computing a basis subfunction with a computation cost Γ as well as the powers of all L basis subfunctions each with a multiplication cost Λ_ℓ and and (b) holds because of the assumptions $\Gamma = L$ and $\Lambda_\ell = \Lambda$.

The total computation cost of linearly separable settings proposed in [33] is obtained as

$$\begin{aligned} \Theta_2 &\overset{(c)}{\leq} N\big(\Gamma' + \sum_{\underline{\mathbf{p}}\in[P_1-2]\times\ldots\times[P_\ell-2]} p_1\ldots p_L\big) \\ &\overset{(d)}{=} N\big(\Gamma' + \sum_{p_1\in[P_1-2]} p_1 \times\ldots\times \sum_{p_L\in[P_L-2]} p_L\big) \\ &\overset{(e)}{=} N\Big(\Gamma' + \big(\frac{(P-2)(P-1)}{2}\big)^L\Big) \end{aligned}$$

where (c) holds because we have N servers each computing a basis subfunction with a computation cost Γ' as well as the number of multiplications for powers of each basis subfunction $\ell \in [L]$, ranging from 1 to $P_\ell - 2$, i.e., $[1 : P_\ell - 2]$, (d) follows some algebraic manipulations, and (e) follows $1+\ldots+n = \frac{n(n+1)}{2}$, assuming $n = P-2$.

F. Proof of Proposition 2

Considering the tensor solution in this work with a computation cost Γ and the multiplication cost Λ, the number of required servers for our non-linearly separable scheme, assuming $\Delta > (\Lambda-1)^\Gamma$, takes the form

$$\begin{aligned} N_{\text{T}} &\overset{(a)}{\leq} \frac{K}{\Delta}\sum_{i=1}^{\binom{L}{\Gamma}}\min\big(\Delta, |X_{i,\text{opt}}|\big) \\ &\overset{(b)}{=} \frac{K}{\Delta}\Bigg[\bigg(\binom{L}{\Gamma} - \sum_{d=1}^{\Gamma-1}\text{mod}\Big(\binom{L}{d}, \binom{L}{\Gamma}\Big)\bigg) \\ &\quad\times \min\Big(\Delta, (\Lambda-1)^\Gamma + \sum_{d=1}^{\Gamma-1}\Big\lfloor\frac{\binom{L}{d}}{\binom{L}{\Gamma}}\Big\rfloor(\Lambda-1)^d\Big) \\ &\quad + \bigg(\sum_{d=1}^{\Gamma-1}\text{mod}\Big(\binom{L}{d}, \binom{L}{\Gamma}\Big)\bigg) \\ &\quad\times \min\Bigg(\Delta, (\Lambda-1)^\Gamma + \sum_{d=1}^{\Gamma-1}(\Lambda-1)^d\Big(\Big\lfloor\frac{\binom{L}{d}}{\binom{L}{\Gamma}}\Big\rfloor \\ &\quad + \text{mod}\Big(\binom{L}{d}, \binom{L}{\Gamma}\Big)\Big)\Bigg)\Bigg]\ . \end{aligned} \tag{75}$$

where (a) holds because of Theorem 2 and (b) follows the fact that after proceeding with matching in the bipartite graph with the set of nodes $u_i = \{\Delta - (\Lambda-1)^\Gamma\}$, $\forall i \in \binom{L}{\Gamma}$ and $v_{id} = \{(\Lambda-1)^d\}$, $\forall i \in \binom{L}{\Gamma}$, $d \in [\Gamma-1]$, where depending on the ratio $\frac{\binom{L}{d}}{\binom{L}{\Gamma}}$, we will have the following possibilities:

- Case I. There are $\bigg(\binom{L}{\Gamma} - \sum_{d=1}^{\Gamma-1}\text{mod}\Big(\binom{L}{d}, \binom{L}{\Gamma}\Big)\bigg)$ tiles with $|X_{i,\text{opt}}| = (\Lambda-1)^\Gamma + \sum_{d=1}^{\Gamma-1}\Big\lfloor\frac{\binom{L}{d}}{\binom{L}{\Gamma}}\Big\rfloor(\Lambda-1)^d$
- Case II. There exist $\sum_{d=1}^{\Gamma-1}\text{mod}\Big(\binom{L}{d}, \binom{L}{\Gamma}\Big)$ tiles with

$$|X_{i,\text{opt}}| = (\Lambda-1)^\Gamma + \sum_{d=1}^{\Gamma-1}(\Lambda-1)^d\Big(\Big\lfloor\frac{\binom{L}{d}}{\binom{L}{\Gamma}}\Big\rfloor$$

$$+ \operatorname{mod}\Big(\binom{L}{d}, \binom{L}{\Gamma}\Big)\Big) \ .$$

Summing the total number of tiles above with different maximum representative rank-one supports results in (75).

Plugging in $\Lambda = 2$ into (75), we have:

$$\begin{aligned} N_{\mathrm{T}} \leq \frac{K}{\Delta}\Bigg[&\Bigg(\binom{L}{\Gamma} - \sum_{d=1}^{\Gamma-1} \operatorname{mod}\Big(\binom{L}{d}, \binom{L}{\Gamma}\Big)\Bigg) \\ &\times \min\Big(\Delta, 1 + \sum_{d=1}^{\Gamma-1} \Big\lfloor \frac{\binom{L}{d}}{\binom{L}{\Gamma}} \Big\rfloor\Big) + \Bigg(\sum_{d=1}^{\Gamma-1} \operatorname{mod}\Big(\binom{L}{d}, \binom{L}{\Gamma}\Big)\Bigg) \\ &\times \min\Bigg(\Delta, 1 + \sum_{d=1}^{\Gamma-1} \Big\lfloor \frac{\binom{L}{d}}{\binom{L}{\Gamma}} \Big\rfloor + \operatorname{mod}\Big(\binom{L}{d}, \binom{L}{\Gamma}\Big)\Bigg)\Bigg] \ . \end{aligned} \tag{76}$$

While the matrix approach to multi-user linearly separable scheme in [36] results in

$$N_{\mathrm{M}} \leq \frac{K}{\Delta} \times \frac{\sum_{d=1}^{\Gamma-1} \binom{L}{d}}{\Gamma} \times \min(\Delta, \Gamma) \ . \tag{77}$$

We first compare the approximate expressions for N_{M} and N_{T} for the asymptotic case (large L) and analyze when $N_{\mathrm{M}} > N_{\mathrm{T}}$ as follows.

$$\begin{aligned} N_{\mathrm{M}} &\sim \frac{K}{\Delta} \cdot \frac{L^{\Gamma-1}}{\Gamma \cdot (\Gamma-1)!} \cdot \min(\Delta, \Gamma) \ . \\ N_{\mathrm{T}} &\sim \frac{K}{\Delta} \cdot \frac{L^{\Gamma}}{\Gamma!} \cdot \min(\Delta, 1) \ . \end{aligned}$$

Considering $N_{\mathrm{M}} > N_{\mathrm{T}}$, we have:

$$\frac{K}{\Delta} \cdot \frac{L^{\Gamma-1}}{\Gamma \cdot (\Gamma-1)!} \cdot \min(\Delta, \Gamma) > \frac{K}{\Delta} \cdot \frac{L^{\Gamma}}{\Gamma!} \cdot \min(\Delta, 1)$$

Using the relation $\Gamma! = \Gamma \cdot (\Gamma-1)!$, we have:

$$\frac{L^{\Gamma-1}}{L^{\Gamma}} \cdot \min(\Delta, \Gamma) > \frac{1}{\Gamma} \cdot \min(\Delta, 1) \ . \tag{78}$$

Because $\Delta > 1$, the relation in (78) simplifies to:

$$\min(\Delta, \Gamma) > \frac{L}{\Gamma} \ . \tag{79}$$

Considering the possible ranges for Δ, we have:

- Case I. Small Δ where $1 < \Delta \leq \Gamma$:

  $$\min(\Delta, \Gamma) = \Delta \ .$$

  The inequality (79) then takes the form

  $$\Delta > \frac{L}{\Gamma} \ .$$

  This condition holds if Δ grows faster than $\frac{L}{\Gamma}$.
- Case II. Large Δ where $\Delta > \Gamma$:

  $$\min(\Delta, \Gamma) = \Gamma \ .$$

  The inequality (79) then takes the form

  $$\Gamma^2 > L \ .$$

  This condition holds only if $\Gamma > \sqrt{L}$.

We next compare the approximate expressions for N_{M} and N_{T} for the non-asymptotic case (small L) and analyze when $N_{\mathrm{M}} > N_{\mathrm{T}}$ as follows.

For small L, we consider the dominant term in N_2 as

$$N_{\mathrm{T}} \sim \frac{K}{\Delta} \cdot \binom{L}{\Gamma} \cdot \min(\Delta, 1) = \frac{K}{\Delta} \cdot \binom{L}{\Gamma} \ .$$

The ratio of N_{M} to N_2 is then approximated as

$$\frac{N_{\mathrm{M}}}{N_{\mathrm{T}}} \sim \frac{\frac{\sum_{d=1}^{\Gamma-1} \binom{L}{d}}{\Gamma} \cdot \min(\Delta, \Gamma)}{\binom{L}{\Gamma}} \ .$$

Under the condition $1 < \Delta \leq \Gamma$, because $\min(\Delta, \Gamma) = \Delta$, the condition $\frac{N_{\mathrm{M}}}{N_{\mathrm{T}}} > 1$ holds if and only if

$$\Delta > \frac{\Gamma \cdot \binom{L}{\Gamma}}{\sum_{d=1}^{\Gamma-1} \binom{L}{d}} \ .$$