The Key Size of Secure Aggregation In Distributed Linearly Separable Computation

Yang Zhou, Wenbo Huang
The School of Electronic Information and Communications,
Huangzhong University of Science and Technology at Wuhan,
430074 Wuhan, China
Email: {hust_zhou, eric_huang}@hust.edu.cn

Claude E. Shannon and David Slepian
Bell Telephone Laboratories, Inc.
Murray Hill, NJ, USA
Email: {csh, dsl}@bell-labs.com

***Abstract*—Secure aggregation has been a key focus in distributed computation and is becoming increasingly important with the growth of distributed machine learning. In this paper, we discuss secure aggregation in a well-known distributed computation scenario called Distributed Linearly Separable Computation. We focus on the two key factors that affect the communication cost and storage cost in the secure distributed linearly separable computation problem: key size and individual key size. We first propose a computing scheme that achieves the minimum individual key size under the optimal communication cost. The proposed scheme reduces the overhead associated with key storage and transmission, which are essential considerations in secure aggregation protocols. Then we research the minimum key size under general computation cost while achieving optimal communication cost. For this purpose, we propose a new computing scheme with a novel assignment strategy.**

***Index Terms*—Distributed Linearly Separable Computation, Secure Aggregation, Key Size**

I. Introduction

Secure aggregation is a well-established topic in distributed computation, referring to the multiparty computation problem where the task is to compute a multiparty sum while no party reveals its update in the clear, not even to the aggregator [1]. The concept of secret sharing was first introduced in [2], where a secret is divided into multiple shares, and only participants meeting a threshold can reconstruct the secret. The complexity of verifiable secret sharing and its extension to multiparty computation were further studied in [3]–[6]. Recently, secure aggregation in Federated Learning has gained significant attention [7]–[11]. [8] achieves secure aggregation with only two rounds of communication, effectively reducing the communication cost. One of the bottlenecks of secure aggregation is key size. Based on the key size, the computation cost for key generation and the communication cost for key sharing will seriously influence the system computation capability. [9], [10] decrease the key size generated locally by each server, which in turn reduces both the computation cost and the communication cost between servers. [11] further explores the concept of individual key size, reducing the individual key size to decrease the storage cost for each server.

In this paper, we consider the secure aggregation in distributed linearly separable computation. The distributed linearly separable computation was first proposed in [12], which unified and extended the models of distributed gradient coding and distributed linear transformation [13]–[16]. A master wants to compute $\mathbf{F}[W_1;\ldots;W_{\mathsf{K}}]$ with the help of N workers, where $W_k, \forall k \in [\mathsf{K}]$ are the computation results of datasets and $\mathbf{F}$ is a demand matrix. From any N_{r} workers' messages, the master can complete the computation task. Each worker is assigned $\mathsf{M} = \mathsf{N} - \mathsf{N}_{\mathrm{r}} + \mathsf{m}$ datasets, where m is the communication reduction factor. Secure aggregation in distributed linearly separable computation ensures that the master can only retrieve the desired task function without obtaining any additional information. Formally, it can be expressed as:

$$I\big(W_1,\ldots,W_{\mathsf{K}};X_{[\mathsf{N}]} \mid \mathbf{F}[W_1;\ldots;W_{\mathsf{K}}]\big)=0, \qquad (1)$$

where $X_{[\mathsf{N}]}$ denotes the encoded messages sent by all workers to the master. The secure aggregation of distributed linearly separable computation problem under minimum computation cost was first proposed in [17], and it discussed the minimum key size without increasing the communication cost.[1]

Key size is a crucial factor affecting both the computation cost and communication cost in secure aggregation, while individual key size influences the storage cost for each participant. In the secure distributed linearly separable computation problem, all keys are shared by a third trusted server to the workers [17]. Since the demand matrix is constantly changing in real-time applications such as those in [18] and convolution computations [19], if we follow the key distribution method proposed in [17], the third trusted server would need to broadcast all keys to the workers. In our proposed scheme, the third trusted server sends a unique key combination to each worker at the beginning, independent of the demand matrix. It is clear that in our scheme, the third trusted server can perform key distribution without needing to know the demand. This significantly reduces the communication cost for key sharing between the third trusted server and the workers and the storage cost for each worker. Moreover, our scheme achieves the optimal individual key size with minimal communication cost. We also extend the scheme in [17] to make it work for general computation cost and provide the minimum key size.

[1]The computation cost is defined as the number of datasets assigned to each worker, while the communication cost is defined as the number of (coded) messages that the master should receive.

Main Contribution: In this paper, we focus on the secure distributed linearly separable computation problem, which prevents the master from accessing any information about the dataset besides the task function.

- We first propose a new computing scheme with a novel assignment strategy for the case $\mathsf{M} = \frac{\mathsf{K}}{\mathsf{N}}(\mathsf{N} - \mathsf{N}_\mathrm{r} + \mathsf{m})$. The computing scheme can cover the optimality results of the computational scheme with fractional repetition assignment; in general, it requires a smaller key size than the computing scheme with cyclic assignment while enabling optimal communication costs. The new computing scheme is also applicable to the case of minimal computational cost in [17].
- We propose a computing scheme that achieves the minimum individual key size under the optimal communication cost. Based on the observation that not all workers have to store all the keys, we propose a novel secure distributed computation scheme with the constraint that $\mathsf{N} \geq \frac{\mathsf{m}+\mathsf{u}-1}{\mathsf{u}} + \mathsf{u}(\mathsf{N}_\mathrm{r} - \mathsf{m} - \mathsf{u} + 1)$ where $\mathsf{u} := \frac{\mathsf{K}_\mathrm{c}\mathsf{N}}{\mathsf{K}}$. which achieves the proposed converse bound on the individual key size within this parameter.

Notation Convention: Calligraphic symbols denote sets, bold lower-case letters denote vector, bold upper-case letters denote matrices, and sans-serif symbols denote system parameters. We use $|\cdot|$ to represent the cardinality of a set or the length of a vector; $[a:b] := \{a, a+1, \ldots, b\}$; $[n] := [1:n]$; $\mathbb{F}_\mathsf{q}$ represents a finite field with order q; $\mathbf{M}^\mathrm{T}$ and $\mathbf{M}^{-1}$ represent the transpose and the inverse of matrix $\mathbf{M}$, respectively; $\mathbf{I}_n$ represents the identity matrix with dimension $n \times n$; $\mathbf{0}_{m\times n}$ represents the zero matrix with dimension $m \times n$; the matrix $[a;b]$ is written in a Matlab form, representing $[a,b]^\mathrm{T}$; $(\mathbf{M})_{m\times n}$ represents the dimension of matrix $\mathbf{M}$ is $m \times n$; $\mathbf{M}^{(\mathcal{S})_\mathrm{r}}$ represents the sub-matrix of $\mathbf{M}$ which is composed of the rows of $\mathbf{M}$ with indices in $\mathcal{S}$ (here r represents 'rows'); $\mathbf{M}^{(\mathcal{S})_\mathrm{c}}$ represents the sub-matrix of $\mathbf{M}$ which is composed of the columns of $\mathbf{M}$ with indices in $\mathcal{S}$ (here c represents 'columns'); $\mathrm{Mod}(b,a)$ represents the modulo operation on b with integer divisor a and in this paper we let $\mathrm{Mod}(b,a) \in \{1,\ldots,a\}$ (i.e., we let $\mathrm{Mod}(b,a) = a$ if a divides b); $\mathrm{GCD}(b,a)$ represents the Greatest Common Divisor of integers b and a; we let $\binom{x}{y} = 0$ if $x < 0$ or $y < 0$ or $x < y$. In this paper, for each set of integers $\mathcal{S}$, we sort the elements in $\mathcal{S}$ in an increasing order and denote the i^{th} smallest element by $\mathcal{S}(i)$, i.e., $\mathcal{S}(1) < \ldots < \mathcal{S}(|\mathcal{S}|)$.

II. System Model

We consider the security aggregation of distributed linearly separable computation over the canonical master-worker distributed system, originally proposed in [17]. A master wants to compute a function of K datasets $D_1, \ldots, D_\mathsf{K}$, with the help of N workers without obtaining any other information about the input datasets. From any N_r workers' messages, the master can complete the computation task. Each worker is assigned $\mathsf{M} = \mathsf{N} - \mathsf{N}_\mathrm{r} + \mathsf{m}$ datasets.

With the assumption that the function is linearly separable from the datasets, the computation task can be written as $\mathsf{K}_\mathrm{c} \leq \mathsf{K}$ linear combinations of K messages

$$\begin{aligned} &f(D_1, D_2, \ldots, D_\mathsf{K}) = g(f_1(D_1), \ldots, f_\mathsf{K}(D_\mathsf{K})) \\ &= g(W_1, \ldots, W_\mathsf{K}) = \mathbf{F}[W_1; \ldots; W_\mathsf{K}] = [F_1; \ldots; F_{\mathsf{K}_\mathrm{c}}], \end{aligned} \quad (2)$$

where the i^{th} message is $W_i = f_i(D_i)$, representing the outcome of the component function $f_i(\cdot)$ applied to dataset D_i. Each message W_i contains L uniformly i.i.d. symbols in $\mathbb{F}_\mathsf{q}$ for some large enough prime-power q. $\mathbf{F}$ represents the demand matrix with dimension $\mathsf{K}_\mathrm{c} \times \mathsf{K}$, where each element is uniformly i.i.d. over $\mathbb{F}_\mathsf{q}$[2].

The secure aggregation scheme imposes the following conditions:

$$I\big(W_1, \ldots, W_\mathsf{K}; X_{[\mathsf{N}]} | \mathbf{F}[W_1; \ldots; W_\mathsf{K}]\big) = 0, \quad (3)$$

where $X_{[\mathsf{N}]}$ presents the whole messages from N workers.

Compared to distributed computing without considering secure aggregation, there is a trusted third-party server to ensure the security, which generates a set of dataset-independent random variables $\mathcal{Q} = \{Q_1, \ldots, Q_\mathsf{N}\}$ on $\mathbb{F}_\mathsf{q}$ and assigns them to workers based on indices. Notice that

$$\begin{aligned} I(Q_1, \ldots, Q_\mathsf{N}; D_1, \ldots, D_\mathsf{K}) &= I(Q_1, \ldots, Q_\mathsf{N}; W_1, \ldots, W_\mathsf{K}) \\ &= 0. \end{aligned} \quad (4)$$

The key size η measures the amount of randomness, i.e.,

$$\eta = \frac{H(Q_1, \ldots, Q_\mathsf{N})}{\mathsf{L}}, \quad (5)$$

The distributed computing framework comprises three phases, *data assignment, computing, and decoding*.

Data assignment phase. We assign $\mathsf{M} := \frac{\mathsf{K}}{\mathsf{N}}(\mathsf{N} - \mathsf{N}_\mathrm{r} + \mathsf{m})$ datasets to each worker. $\mathcal{Z}_n$ denotes the set of indices of datasets assigned to worker n, where $\mathcal{Z}_n \subseteq [\mathsf{K}]$ and $|\mathcal{Z}_n| \leq \mathsf{M}$. In addition, the third trusted server distributes the keys to each worker for security.

Computing phase. Each worker $n \in [\mathsf{N}]$ first computes the message $W_k = f_k(D_k)$ for each $k \in \mathcal{Z}_n$. Then it computes the coded message

$$X_n = \psi_n\left(\{W_k : k \in \mathcal{Z}_n\}, Q\right), \quad (6)$$

where ψ_n is a function of messages $\{W_k : k \in \mathcal{Z}_n\}$,

$$\psi_n : \mathbb{F}_\mathsf{q}^{|\mathcal{Z}_n|\mathsf{L}} \times |\mathcal{Q}| \to \mathbb{F}_\mathsf{q}^{\mathsf{T}_n}, \quad (7)$$

and T_n represents the length of X_n. Finally, worker n sends X_n back to the master. $|\mathcal{Z}_n|$ is denoted as the computation cost.[3]

[2] In this paper, we assume that K/N is an integer and L is large enough such that any sub-message division is possible. We only consider $\mathsf{K}_\mathrm{c} \leq \mathsf{K}$ because when $\mathsf{K}_\mathrm{c} \leq \mathsf{K}$, $\mathrm{Rank}(\mathbf{F}) = \mathsf{K}_\mathrm{c}$ with high probability and when $\mathsf{K}_\mathrm{c} > \mathsf{K}$, $\mathrm{Rank}(\mathbf{F}) = \mathsf{K}$ with high probability, we only consider the non-trivial setting $\mathsf{K}_\mathrm{c} \leq \mathsf{K}$

[3] In a distributed system, the complexity of computing the linear combinations of the messages is much lower than computing the separable functions. So, the computation cost can be represented by the number of messages each worker computes.

Decoding phase. The master can use coded messages from any N_r arriving workers to decode the computation tasks since the transmission environment and the computation capability of each worker are unknown. Thus, for each subset of workers $\mathcal{A} \subseteq [\mathsf{N}]$ where $|\mathcal{A}| = \mathsf{N}_\mathrm{r}$, by defining $X_\mathcal{A} := \{X_n : n \in \mathcal{A}\}$, there should be a decoding function such that $\hat{g}_\mathcal{A} = \phi_\mathcal{A}\big(X_\mathcal{A}, \mathbf{F}\big)$, where

$$\phi_\mathcal{A} : \mathbb{F}_\mathsf{q}^{\sum_{n\in\mathcal{A}} \mathsf{T}_n} \times \mathbb{F}_\mathsf{q}^{\mathsf{K}_\mathrm{c}\times\mathsf{K}} \to \mathbb{F}_\mathsf{q}^{\mathsf{K}_\mathrm{c}\times\mathsf{L}}. \tag{8}$$

The worst-case error probability is defined as

$$\varepsilon := \max_{\mathcal{A}\subseteq[\mathsf{N}]:|\mathcal{A}|=\mathsf{N}_\mathrm{r}} \Pr\{\hat{g}_\mathcal{A} \neq g(W_1, \ldots, W_\mathsf{K})\}. \tag{9}$$

A computing scheme is called achievable if the worst-case error probability vanishes when $\mathsf{q} \to \infty$.

We first introduce a novel converse bound on η for a fixed assignment, whose proof can be found in Appendix A.

Theorem 1. *For the $(\mathsf{K}, \mathsf{N}, \mathsf{N}_\mathrm{r}, \mathsf{K}_\mathrm{c}, \mathsf{m})$ secure distributed linearly separable computation problem with $\mathsf{M} = \frac{\mathsf{K}}{\mathsf{N}}(\mathsf{N} - \mathsf{N}_\mathrm{r} + \mathsf{m}) = \frac{\mathsf{K}}{\mathsf{N}}\mathsf{M}'$ and $\mathsf{K}_\mathrm{c} = 1$, for a fixed assignment $\mathbf{Z} = (\mathcal{Z}_1, \ldots, \mathcal{Z}_\mathsf{N})$, if there exists an ordered set of workers in $[\mathsf{N}]$ denoted by $\mathbf{s} = (s_1, \ldots, s_{|\mathbf{s}|})$,*

$$\forall i \in [|\mathbf{s}|], \exists x \in \mathcal{Z}_{s_i} \text{ such that } \sum_{j=1}^{i-1} \mathbb{1}(x \in \mathcal{Z}_{s_j}) \leq m - 1, \tag{10}$$

where the symbol $\mathbb{1}(A)$ denotes the indicator function, which is 1 if the condition A is true, and 0 otherwise. It must hold that

$$\eta \geq \frac{|\mathbf{s}|}{\mathsf{m}} - 1. \tag{11}$$

Corollary 1. *For the $(\mathsf{K}, \mathsf{N}, \mathsf{N}_\mathrm{r}, \mathsf{K}_\mathrm{c}, \mathsf{m})$ secure distributed linearly separable computation problem with $\mathsf{M} = \frac{\mathsf{K}}{\mathsf{N}}(\mathsf{N} - \mathsf{N}_\mathrm{r} + \mathsf{m}) = \frac{\mathsf{K}}{\mathsf{N}}\mathsf{M}'$ and $\mathsf{K}_\mathrm{c} = 1$, it must hold that*

$$\eta^\star \geq \frac{\left\lceil \frac{\mathsf{mN}}{\mathsf{N}-\mathsf{N}_\mathrm{r}+1} \right\rceil}{\mathsf{m}} - 1. \tag{12}$$

It can be seen that the key size η depends on the data assignment. In this paper, we discuss the minimum η^* required for the secure aggregation of distributed linearly separable computation.

Notice that in order to satisfy security constraints, it should satisfy that

$$I\big(W_1, \ldots, W_\mathsf{K}; X_{[\mathsf{N}]} | \mathbf{F}[W_1; \ldots; W_\mathsf{K}]\big) = 0. \tag{13}$$

We define

$$\mathsf{R} := \max_{\mathcal{A}\subseteq[\mathsf{N}]:|\mathcal{A}|=\mathsf{N}_\mathrm{r}} \frac{\sum_{n\in\mathcal{A}} \mathsf{T}_n}{\mathsf{L}} \tag{14}$$

as the communication cost, which represents the maximum normalized number of symbols received from any N_r workers to recover the computational task.

In this paper, we focus on the minimum key size under the optimal communication cost, when $\mathsf{K}_\mathrm{c} = 1, \mathsf{m} \geq 1$.

III. Main Results

Theorem 2. *For the $(\mathsf{K}, \mathsf{N}, \mathsf{N}_\mathrm{r}, \mathsf{K}_\mathrm{c}, \mathsf{m})$ secure distributed linearly separable computation problem with $\mathsf{M} = \frac{\mathsf{K}}{\mathsf{N}}(\mathsf{N} - \mathsf{N}_\mathrm{r} + \mathsf{m}) = \frac{\mathsf{K}}{\mathsf{N}}\mathsf{M}'$ and $\mathsf{K}_\mathrm{c} = 1$, the optimal key size $\eta = \frac{h(\mathsf{N},\mathsf{M}')}{\mathsf{m}} - 1$ is achievable, where the output of the function $h(\cdot,\cdot)$ is given by the recursive algorithm shown in Fig. 1 with the following properties:*

- *When $\mathsf{N} > 2\mathsf{M}'$, by Scheme 1 described in Section IV-A we have*

$$h(\mathsf{N}, \mathsf{M}') = h\big(\mathsf{N} - \lfloor \mathsf{N}/\mathsf{M}' - 1 \rfloor \mathsf{M}', \mathsf{M}'\big) + \mathsf{m} \lfloor \mathsf{N}/\mathsf{M}' - 1 \rfloor. \tag{15}$$

- *When $1.5\mathsf{M}' \leq \mathsf{N} < 2\mathsf{M}'$ and M' is even, with $\mathsf{M}' \geq 2\mathsf{m}$, by Scheme 2 described in Section IV-B, we have*

$$h(\mathsf{N}, \mathsf{M}') = h\left(\mathsf{N} - \mathsf{M}', \frac{\mathsf{M}'}{2}\right) + \mathsf{m}. \tag{16}$$

- *When $1.5\mathsf{M}' \leq \mathsf{N} < 2\mathsf{M}'$ and M' is odd, with $\mathsf{M}' \geq 2\mathsf{m}+1$, by Scheme 3 described in Section IV-C we have*

$$h(\mathsf{N}, \mathsf{M}') = \mathsf{N} - \frac{3\mathsf{M}' - 1}{2} + 2\mathsf{m}; \tag{17}$$

- *When $\mathsf{M}' < \mathsf{N} < 1.5\mathsf{M}'$, with $\mathsf{M}' \geq 2\mathsf{m}$, by Scheme 4 described in Section IV-D we have*

$$h(\mathsf{N}, \mathsf{M}') = h(\mathsf{M}', 2\mathsf{M}' - \mathsf{N}). \tag{18}$$

□

Notice that λ in Fig. 1 represents the number of totally transmitted linearly independent combinations of merged messages, and we aim to minimize it.

Remark 1 (The key size in extreme cases)**.** For the $(\mathsf{K}, \mathsf{N}, \mathsf{N}_\mathrm{r}, \mathsf{K}_\mathrm{c}, \mathsf{m})$ secure distributed linearly separable computation problem, where $\mathsf{M} = \frac{\mathsf{K}}{\mathsf{N}}(\mathsf{N} - \mathsf{N}_\mathrm{r} + \mathsf{m}) = \frac{\mathsf{K}}{\mathsf{N}}\mathsf{M}'$, $\mathsf{K}_\mathrm{c} = 1$, and M divides N, the minimum required key size to achieve the optimal communication cost is

$$\eta^\star = \frac{\mathsf{N}}{\mathsf{M}'} - 1. \tag{19}$$

For the $(\mathsf{K}, \mathsf{N}, \mathsf{N}_\mathrm{r}, \mathsf{K}_\mathrm{c}, \mathsf{m})$ secure distributed linearly separable computation problem, where $\mathsf{M} = \frac{\mathsf{K}}{\mathsf{N}}(\mathsf{N} - \mathsf{N}_\mathrm{r} + \mathsf{m})$ and $\mathsf{K}_\mathrm{c} = 1$, the minimum key size required to achieve the optimal communication cost, under the cyclic assignment constraint, is given by

$$\eta^\star_\mathrm{cyc} = \frac{\mathsf{N}_\mathrm{r}}{\mathsf{m}} - 1. \tag{20}$$

Remark 2 (Simplification of the security problem)**.** As shown in [17, Theorem 2], for any $(\mathsf{K}, \mathsf{N}, \mathsf{N}_\mathrm{r}, \mathsf{K}_\mathrm{c}, \mathsf{m})$ non-secure distributed linearly separable computation problem, the encoding scheme can be made secure without increasing the communication cost. Therefore, for the secure distributed linearly separable computation problem, we can simplify the problem to the encoding schemes of non-secure distributed linearly separable computation, corresponding to the scheme we propose later in this work.

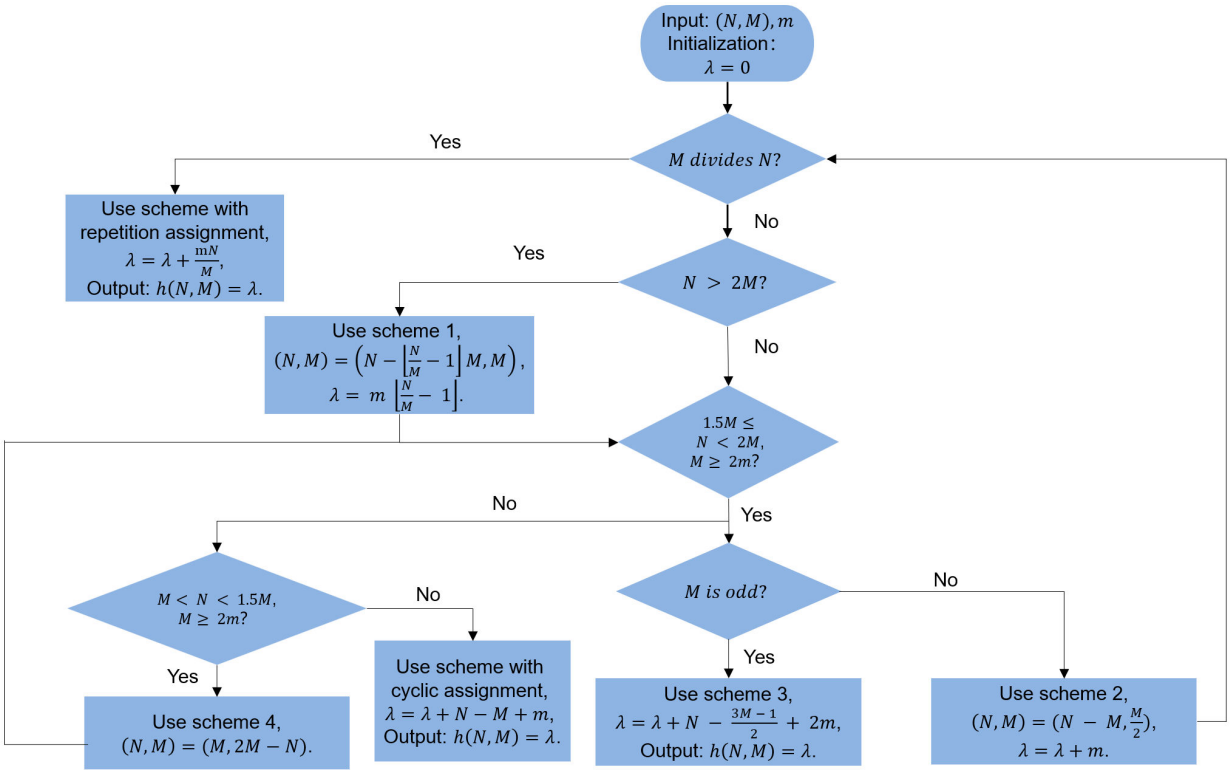


Fig. 1: Flow diagram of the combined scheme in Theorem 2.

At the end of this section, we present numerical evaluations comparing the key size required by the cyclic assignment scheme and the combined scheme from Theorem 2, both achieving optimal communication cost. In Fig. 2a, we fix $\mathsf{N} = 24, \mathsf{m} = 2$ and show the tradeoff between M and η. In Fig. 2b, we fix $\mathsf{M} = 6, \mathsf{m} = 2$ and show the tradeoff between N and η. In Fig. 2c, we fix $\mathsf{N} = 24, \mathsf{M} = 10$ and show the tradeoff between m and η. The results demonstrate that the combined scheme requires a significantly smaller key size than the cyclic assignment scheme.

IV. Novel Achievable Scheme For Theorem 2

A. Scheme 1 for (15)

In this section, we consider the case where $\mathsf{N} > 2\mathsf{M}$. We first focus on the workers and datasets in the range $[(i-1)\mathsf{M}+1 : i\mathsf{M}]$ for any $i \in [\lfloor \mathsf{N}/\mathsf{M} - 1 \rfloor]$. The datasets in this range are assigned to the corresponding workers using the fractional repetition assignment strategy. This results in $\mathsf{m}\lfloor \mathsf{N}/\mathsf{M} - 1 \rfloor$ linearly independent combinations of messages.

We then consider the remaining $\mathsf{N} - \lfloor \mathsf{N}/\mathsf{M} - 1 \rfloor \mathsf{M}$ datasets and workers. The original (N, M) problem is thus reduced to solving a smaller problem of $(\mathsf{N} - \lfloor \mathsf{N}/\mathsf{M} - 1 \rfloor \mathsf{M}, \mathsf{M})$. The specific scheme for addressing this reduced problem is detailed below.

Example 1. We consider the $(\mathsf{N}, \mathsf{M}) = (7, 3)$ non-secure problem, and assume $\mathsf{m} = 2$, which gives $\mathsf{N}_\mathrm{r} = \mathsf{N} - \mathsf{M} + \mathsf{m} = 6$. For the sake of simplicity, we assume in this example that the field is a sufficiently large prime field; in general, the proposed scheme does not need this assumption (recall that we only need the field size to be large enough).

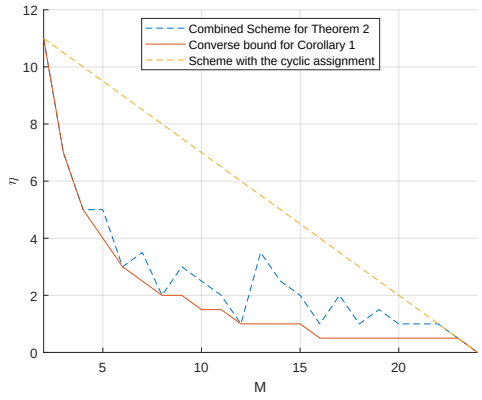
Data assignment phase. The data assignment is shown in the following figure.

Worker 1	Worker 2	Worker 3	Worker 4
D_1	D_1	D_1	D_4
D_2	D_2	D_2	D_5
D_3	D_3	D_3	D_6

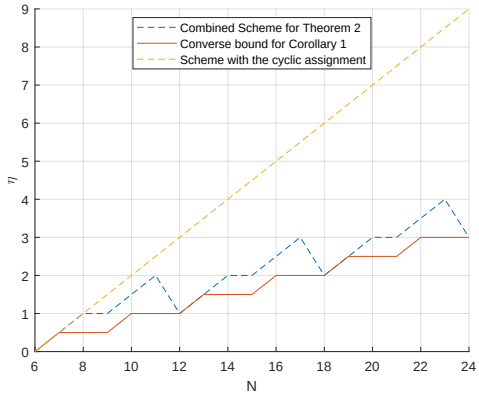
Worker 5	Worker 6	Worker 7
D_5	D_6	D_7
D_6	D_7	D_4
D_7	D_4	D_5

Computing phase. To achieve the optimal communication cost, we first divide each message $W_k, k \in [\mathsf{K}]$, into m non-overlapping and equal-length sub-messages, denoted as $W_k = \{W_{k,j} : j \in [\mathsf{m}]\}$, so that each sub-message contains $\frac{\mathsf{L}}{\mathsf{m}} = \frac{\mathsf{L}}{2}$ symbols in $\mathbb{F}_\mathsf{q}$.

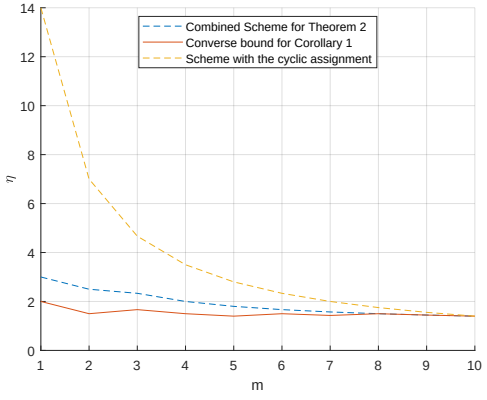
After dividing the messages, we first have workers 1, 2, and 3, which store the same data, compute random linear combinations of $(W_{1,1} + W_{2,1} + W_{3,1})$ and $(W_{1,2} + W_{2,2} + W_{3,2})$. Specifically, we can assign the following computations: Worker 1 computes $W_{1,1}+W_{2,1}+W_{3,1}+W_{1,2}+W_{2,2}+W_{3,2}$, Worker 2 computes $W_{1,1} + W_{2,1} + W_{3,1} + 2(W_{1,2} + W_{2,2} + W_{3,2})$, Worker 3 computes $W_{1,1} + W_{2,1} + W_{3,1} + 3(W_{1,2} +$



(a) (M, η) tradeoff with $\mathsf{N} = 24$ and $\mathsf{m} = 2$.



(b) (N, η) tradeoff with $\mathsf{M} = 6$ and $\mathsf{m} = 2$.



(c) (m, η) tradeoff with $\mathsf{N} = 24$ and $\mathsf{M} = 10$.

Fig. 2: Numerical evaluations for the considered secure distributed linearly separable computation problem.

$W_{2,2} + W_{3,2})$.

Next, we focus on workers 4, 5, 6, and 7. We use the cyclic assignment scheme to distribute data blocks D_4, D_5, D_6, and D_7 among these workers. According to the cyclic coding scheme, the computations are assigned as follows: Worker 4 computes $\frac{1}{2}W_{5,1} - W_{4,2} - \frac{2}{3}W_{6,2}$, Worker 5 computes $\frac{1}{2}W_{5,1} + W_{5,2} + \frac{1}{3}W_{6,2} + W_{7,2}$, Worker 6 computes $W_{4,1} + W_{6,1} + W_{7,1} + 2W_{4,2} + \frac{4}{3}W_{6,2}$, Worker 7 computes $\frac{3}{2}W_{5,1} - W_{4,2} + 2W_{5,2} + 2W_{7,2}$.

Decoding phase. Among the information received from any $\mathsf{N}_\mathrm{r} = 6$ workers, at least two messages come from workers 1, 2, and 3. This allows the master to recover $W_{1,1}+W_{2,1}+W_{3,1}$ and $W_{1,2} + W_{2,2} + W_{3,2}$. Additionally, three messages are from workers 4, 5, 6, and 7, enabling the master to reconstruct $W_{4,1} + W_{5,1} + W_{6,1} + W_{7,1}$ and $W_{4,2} + W_{5,2} + W_{6,2} + W_{7,2}$. Together with the previously recovered information, the master can obtain $W_{1,1} + W_{2,1} + W_{3,1} + W_{4,1} + W_{5,1} + W_{6,1} + W_{7,1}$ and $W_{1,2} + W_{2,2} + W_{3,2} + W_{4,2} + W_{5,2} + W_{6,2} + W_{7,2}$.

It can be observed that the number of linearly independent combinations received from workers 1, 2, and 3 is 2, and from workers 4, 5, 6, and 7, the number of linearly independent combinations is 3. The total sum gives $h(7,3) = 5$, which equals $h(4,3) + \mathsf{m}$ for $\mathsf{m} = 2$, and this satisfies (15).

Then we consider the general case of the (N, M) non-secure problem when $\mathsf{N} > 2\mathsf{M}$, and provide a generic scheme for this case to prove (15). We assume that for the non-secure problem $\left(\mathsf{N} - \lfloor \mathsf{N}/\mathsf{M} - 1 \rfloor \mathsf{M}, \mathsf{M}\right)$, a specific scheme is already available, and from that scheme, we have obtained $h\big(\mathsf{N} - \lfloor \mathsf{N}/\mathsf{M} - 1 \rfloor \mathsf{M}, \mathsf{M}\big)$ linearly independent combinations.

Data assignment phase. We partition the entire system into $\lfloor \mathsf{N}/\mathsf{M} \rfloor$ blocks. For each $i \in [\lfloor \mathsf{N}/\mathsf{M} \rfloor - 1]$, the i^{th} block includes the datasets $\{D_k : k \in [(i-1)\mathsf{M} + 1 : i\mathsf{M}]\}$ and the workers in the range $[(i-1)\mathsf{M} + 1 : i\mathsf{M}]$. The last block includes the datasets and workers in the range $[\lfloor \mathsf{N}/\mathsf{M} - 1 \rfloor \mathsf{M} + 1 : \mathsf{N}]$.

For the first $[\lfloor \mathsf{N}/\mathsf{M} \rfloor - 1]$ blocks, we use the fractional repetition assignment to distribute the data. As for the last block, it contains $\mathsf{N} - \lfloor \mathsf{N}/\mathsf{M} - 1 \rfloor \mathsf{M}$ workers and $\mathsf{N} - \lfloor \mathsf{N}/\mathsf{M} - 1 \rfloor \mathsf{M}$ datasets, with each worker being able to compute M datasets. Therefore, we can directly apply the existing data assignment scheme designed for solving the $(\mathsf{N} - \lfloor \mathsf{N}/\mathsf{M} - 1 \rfloor \mathsf{M}, \mathsf{M})$ non-secure problem to this last block.

Computing phase. For each $i \in [\lfloor \mathsf{N}/\mathsf{M} \rfloor - 1]$, each worker in the i^{th} block computes m linear combinations $\sum_{k \in [(i-1)\mathsf{M}+1:i\mathsf{M}]} W_{k,j}$, and each worker sends a random linear combination of these m computed combinations.

Next, we focus on the last block. The computation phase in this block still follows the $(\mathsf{N} - \lfloor \mathsf{N}/\mathsf{M} - 1 \rfloor \mathsf{M}, \mathsf{M})$ non-secure scheme, which results in a total of $h\left(\mathsf{N} - \lfloor \mathsf{N}/\mathsf{M} - 1 \rfloor \mathsf{M}, \mathsf{M}\right)$ linearly independent combinations being sent.

Decoding phase. The master receives information from $\mathsf{N}_\mathrm{r} = \mathsf{N} - \mathsf{M} + \mathsf{m}$ workers, meaning that responses from $\mathsf{M} - \mathsf{m}$ workers will not be received. For the first $\lfloor \mathsf{N}/\mathsf{M} - 1 \rfloor$ blocks, each block has M workers, so at least m workers' responses from each block are received by the master. Thus, from each block, the master can recover $\sum_{k \in [(i-1)\mathsf{M}+1:i\mathsf{M}]} W_{k,j}$, where j ranges over $[\mathsf{m}]$. For the last block, at least $\mathsf{N} - \lfloor \mathsf{N}/\mathsf{M} - 1 \rfloor \mathsf{M} - \mathsf{M} + \mathsf{m}$ workers will respond. By carefully designing the coding scheme, we can recover $\sum_{k \in [\lfloor \mathsf{N}/\mathsf{M} - 1 \rfloor \mathsf{M} + 1 : \mathsf{N}]} W_k$ from the information sent by any $\mathsf{N} - \lfloor \mathsf{N}/\mathsf{M} - 1 \rfloor \mathsf{M} - \mathsf{M} + \mathsf{m}$ workers in the last block. together with the transmissions of the first $\lfloor \mathsf{N}/\mathsf{M} - 1 \rfloor$ blocks, the master can recover $W_1 + \cdots + W_\mathsf{N}$.

In conclusion, we have demonstrated that $h(\mathsf{N}, \mathsf{M}) = \mathsf{m} \lfloor \mathsf{N}/\mathsf{M} - 1 \rfloor + h\left(\mathsf{N} - \lfloor \mathsf{N}/\mathsf{M} - 1 \rfloor \mathsf{M}, \mathsf{M}\right)$, which aligns with equation (15). Furthermore, it is evident that $\mathsf{M} < \mathsf{N} -$

$$\mathbf{F} = \begin{bmatrix} \mathbf{f}_1 \\ \mathbf{f}_2 \\ \mathbf{f}_3 \\ \mathbf{f}_4 \\ \mathbf{f}_5 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 2 & 2 & 2 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 2 & 2 & 1 & 1 & 1 \\ 0 & 2a_1 & 2a_2 & 2a_3 & a_1 & a_2 & a_3 & 0 & 2a_4 & 2a_5 & 2a_6 & a_4 & a_5 & a_6 \end{bmatrix} \tag{21}$$

$$= \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 2 & 2 & 2 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 2 & 2 & 1 & 1 & 1 \\ 0 & 2 & 4 & 6 & 1 & 2 & 3 & 0 & 8 & 10 & 12 & 4 & 5 & 6 \end{bmatrix} \tag{22}$$

$\lfloor \mathsf{N}/\mathsf{M} - 1 \rfloor \mathsf{M} < 2\mathsf{M}$ when $\mathsf{N} > 2\mathsf{M}$ and M does not evenly divide N.

B. Scheme 2 for (16)

We first provide an example to illustrate the main idea.

Example 2. *We consider the* $(\mathsf{N}, \mathsf{M}) = (7, 4)$ *non-secure problem, and assume* $\mathsf{m} = 2$*, which gives* $\mathsf{N}_\mathrm{r} = \mathsf{N} - \mathsf{M} + \mathsf{m} = 5$*.*

Data assignment phase. We assign the datasets as follows.

worker 1	*worker 2*	*worker 3*	*worker 4*
D_1	D_1	D_1	D_1
D_2	D_2	D_5	D_5
D_3	D_3	D_6	D_6
D_4	D_4	D_7	D_7

worker 5	*worker 6*	*worker 7*
D_2	D_3	D_4
D_5	D_6	D_7
D_3	D_4	D_2
D_6	D_7	D_5

Notice that datasets in the range $[2:7]$ *are divided into three parts:* $\mathcal{P}_1 = \{2, 5\}$*,* $\mathcal{P}_2 = \{3, 6\}$*, and* $\mathcal{P}_3 = \{4, 7\}$*. These three pairs of datasets are assigned cyclically to workers 5, 6, and 7.*

Computing phase. To minimize communication cost, each message W_k*, for* $k \in [\mathsf{K}]$*, is divided into* m *equal-length sub-messages, denoted as* $W_k = \{W_{k,j} : j \in [\mathsf{m}]\}$*, with each sub-message containing* $\frac{\mathsf{L}}{\mathsf{m}} = \frac{\mathsf{L}}{2}$ *symbols in* $\mathbb{F}_\mathsf{q}$*. As shown at the top of the next page, our demand matrix is defined as* $[F_1; F_2; F_3; F_4; F_5] = \mathbf{F}$ $[W_{1,1}; \ldots; W_{7,2}]$*, where each* a_i *is uniformly independently and identically distributed (i.i.d.) over the finite field* $\mathbb{F}_\mathsf{q}$*. In this example, we assume the values of* $(a_1, a_2, a_3, a_4, a_5, a_6)$ *are* $(1, 2, 3, 4, 5, 6)$*.*

To recover $\mathbf{F}$ $[W_{1,1}; \ldots; W_{7,2}]$ *from any five responding workers, we let the workers 1 and 2, who have datasets* $[1:4]$*, compute the following:*

$$F_1 - F_3 = W_{1,1} - W_{2,1} - W_{3,1} - W_{4,1},$$
$$F_2 - F_4 = W_{1,2} - W_{2,2} - W_{3,2} - W_{4,2}.$$

Since both workers 1 and 2 can compute $F_1 - F_3$ *and* $F_2 - F_4$*, we let workers 1 and 2 calculate random linear combinations of* $(F_1 - F_3)$ *and* $(F_2 - F_4)$*. For example, we can assign worker 1 to compute* $X_1 = (F_1 - F_3) + (F_2 - F_4)$*, and worker 2 to compute* $X_2 = (F_1 - F_3) - (F_2 - F_4)$*.*

For workers 3 and 4, they can compute

$$2F_1 - F_3 = 2W_{1,1} + W_{5,1} + W_{6,1} + W_{7,1},$$
$$2F_2 - F_4 = 2W_{1,2} + W_{5,2} + W_{6,2} + W_{7,2}.$$

Here, we adopt the same computation scheme: let worker 3 compute $X_3 = (2F_1 - F_3) + 2(2F_2 - F_4)$*, and let worker 4 compute* $X_4 = (2F_1 - F_3) - 2(2F_2 - F_4)$*.*

We then focus on workers 5, 6, 7. Previously, we divided the datasets $[2:7]$ *into three parts. Here, we define the following:*

$$P_{1,1} = 2W_{2,1} + W_{5,1},$$
$$P_{1,2} = 2W_{2,2} + W_{5,2},$$
$$P_{2,1} = 2W_{3,1} + W_{6,1},$$
$$P_{2,2} = 2W_{3,2} + W_{6,2},$$
$$P_{3,1} = 2W_{4,1} + W_{7,1},$$
$$P_{3,2} = 2W_{4,2} + W_{7,2}.$$

We assume that $P_k = \{P_{k,j} : j \in [\mathsf{m}]\}$*. Hence, we can treat workers 5, 6, and 7, along with* P_1*,* P_2*, and* P_3*, as a* $(3, 2)$ *non-secure problem when* $m = 2$*. From the responses of these three workers, we can recover* $\mathbf{F}'$ $[P_{1,1}; \ldots; P_{3,2}]$*, which corresponds to recovering the sub-matrix* $[\mathbf{F}_3; \mathbf{F}_4; \mathbf{F}_5]$ $[W_{1,1}; \ldots; W_{7,2}]$*. The demand matrix* $\mathbf{F}'$ *for this* $(3, 2)$ *non-secure problem is given as follows:*

$$\mathbf{F}' = \begin{bmatrix} \mathbf{f}'_1 \\ \mathbf{f}'_2 \\ \mathbf{f}'_3 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 & 5 & 6 \end{bmatrix} \tag{23}$$

Here, we assign worker 5 to compute

$$\begin{aligned} X_5 =& 3F'_1 - F'_3 + 6F'_2 = 2P_{1,1} + P_{2,1} + 2P_{1,2} + P_{2,2} \\ =& 4W_{2,1} + 2W_{3,1} + 2W_{5,1} + W_{6,1} \\ &+ 4W_{2,2} + 2W_{3,2} + 2W_{5,2} + W_{6,2}. \end{aligned}$$

Worker 6 computes

$$\begin{aligned} X_6 =& F'_3 - F'_1 - 4F'_2 = P_{2,1} + 2P_{3,1} + P_{2,2} + 2P_{3,2} \\ =& 2W_{3,1} + 4W_{4,1} + W_{6,1} + 2W_{7,1} \\ &+ 2W_{3,2} + 4W_{4,2} + W_{6,2} + 2W_{7,2}. \end{aligned}$$

Finally, worker 7 computes

$$\begin{aligned} X_7 &= F_3' - 2F_1' - 5F_2' = -P_{1,1} + P_{3,1} - P_{1,2} + P_{3,2} \\ &= -2W_{2,1} + 2W_{4,1} - W_{5,1} + W_{7,1} \\ &\quad - 2W_{2,2} + 2W_{4,2} - W_{5,2} + W_{7,2}. \end{aligned}$$

Notice that X_5, X_6, and X_7 are in fact linear combinations of F_3, F_4, and F_5. From these linear combinations, we can recover the submatrix $[\mathbf{f}_3; \mathbf{f}_4; \mathbf{f}_5]$ $[W_{1,1}; \ldots; W_{7,2}]$.

Decoding phase. For the case when $\mathsf{m} = 2$, the master can receive $\mathsf{N_r} = \mathsf{N} - \mathsf{M} + \mathsf{m} = 5$ responses from the workers. Since the information computed by each worker is linearly independent, the master receives five linearly independent combinations of F_1, F_2, F_3, F_4, and F_5. Thus, we can fully recover the demand matrix $\mathbf{F}$. We find that the number of received linearly independent combinations is $h(7,4) = h(3,2) + \mathsf{m} = 5$, coinciding with (16)*.*

We now focus on the case where $1.5\mathsf{M} \leq \mathsf{N} < 2\mathsf{M}$, and M is even, with $\mathsf{M} \geq 2\mathsf{m}$. For this case, we consider the general Scheme 3 for the (N, M) non-secure problem to prove (16). We assume that a specific feasible scheme already exists for the $\left(\mathsf{N} - \mathsf{M}, \frac{\mathsf{M}}{2}\right)$ non-secure problem and that this scheme can transmit a total of $h\left(\mathsf{N} - \mathsf{M}, \frac{\mathsf{M}}{2}\right)$ linearly independent combinations of messages.

Data assignment phase. We first consider the dataset assignment for the workers in $[\mathsf{M}]$, which is structured as follows:

worker 1	$\cdots$	worker $\frac{\mathsf{M}}{2}$
D_1	$\cdots$	D_1
$\cdots$	$\cdots$	$\cdots$
D_{y}	$\cdots$	D_{y}
$D_{\mathsf{y}+1}$	$\cdots$	$D_{\mathsf{y}+1}$
$\cdots$	$\cdots$	$\cdots$
D_{M}	$\cdots$	D_{M}

worker $\frac{\mathsf{M}}{2}+1$	$\cdots$	worker M
D_1	$\cdots$	D_1
$\cdots$	$\cdots$	$\cdots$
D_{y}	$\cdots$	D_{y}
$D_{\mathsf{M}+1}$	$\cdots$	$D_{\mathsf{M}+1}$
$\cdots$	$\cdots$	$\cdots$
D_{N}	$\cdots$	D_{N}

In this setup, we assign datasets D_1 through D_{y} to all workers in $[\mathsf{M}]$, while each dataset D_k for $k \in [\mathsf{y}+1 : \mathsf{N}]$ is distributed among $\frac{\mathsf{M}}{2}$ workers in $[\mathsf{M}]$.

Next, we consider the assignment for the last $\mathsf{N}-\mathsf{M}$ workers. We need to allocate $\mathsf{N}-\mathsf{y} = 2(\mathsf{N}-\mathsf{M})$ datasets (from $[\mathsf{y}+1 : \mathsf{N}]$) to a total of $\mathsf{N}-\mathsf{M}$ workers. Each dataset is assigned to $\frac{\mathsf{M}}{2}$ workers, and each worker receives M datasets.

Datasets in $[\mathsf{y}+1 : \mathsf{N}]$ are divided into $\frac{\mathsf{N}-\mathsf{y}}{2} = \mathsf{N}-\mathsf{M}$ pairs, where the i^{th} pair is defined as $\mathcal{P}_i = \{\mathsf{y}+i, \mathsf{M}+i\}$ for $i \in [\mathsf{N}-\mathsf{M}]$. Thus, we apply the assignment phase of the scheme for the $\left(\mathsf{N}-\mathsf{M}, \frac{\mathsf{M}}{2}\right)$ non-secure problem, assigning each pair to $\frac{\mathsf{M}}{2}$ workers, with each worker being allocated $\frac{\mathsf{M}}{2}$ pairs.

Computing phase. We divide each message W_k for $k \in [\mathsf{K}]$ into m equal-length, non-overlapping sub-messages, denoted as $W_k = \{W_{k,j} : j \in [\mathsf{m}]\}$. We first focus on the workers in $[\mathsf{M}]$. By constructing the messages sent by these workers, we can recover the following $2\mathsf{m}$ linear combinations:

$$\begin{aligned} F_1 &= W_{1,1} + W_{2,1} + \cdots + W_{\mathsf{N},1}, \\ &\vdots \\ F_{\mathsf{m}} &= W_{1,\mathsf{m}} + W_{2,\mathsf{m}} + \cdots + W_{\mathsf{N},\mathsf{m}}, \end{aligned}$$

$$\begin{aligned} F_{\mathsf{m}+1} &= 2(W_{\mathsf{y}+1,1} + \cdots + W_{\mathsf{M},1}) + W_{\mathsf{M}+1,1} + \cdots + W_{\mathsf{N},1}, \\ &\vdots \\ F_{2\mathsf{m}} &= 2(W_{\mathsf{y}+1,\mathsf{m}} + \cdots + W_{\mathsf{M},\mathsf{m}}) + W_{\mathsf{M}+1,\mathsf{m}} + \cdots + W_{\mathsf{N},\mathsf{m}}. \end{aligned}$$

We group the m linear combinations $(F_1 - F_{\mathsf{m}+1}), \ldots, (F_{\mathsf{m}} - F_{2\mathsf{m}})$ into one group, and the m linear combinations $(2F_1 - F_{\mathsf{m}+1}), \ldots, (2F_{\mathsf{m}} - F_{2\mathsf{m}})$ into another group. Since the workers in $[\mathsf{M}/2]$ are not assigned any data from $[\mathsf{M}+1 : \mathsf{N}]$, we let each worker in this range compute a random linear combination of the m linear combinations in the first group. The workers in $[\mathsf{M}/2+1 : \mathsf{M}]$, who are not assigned any data from $[\mathsf{y}+1 : \mathsf{M}]$, are assigned to compute a random linear combination of the m linear combinations in the second group.

We then focus on the workers in $[\mathsf{M}+1 : \mathsf{N}]$. For each dataset pair $\mathcal{P}_i = \{\mathsf{y}+i, \mathsf{M}+i\}$ where $i \in [\mathsf{N}-\mathsf{M}]$, we define $P_i = 2W_{\mathsf{y}+i} + W_{\mathsf{M}+i}$. Therefore, we can express $F_{\mathsf{m}+1}$ to $F_{2\mathsf{m}}$ as $P_{1,1} + \cdots + P_{\mathsf{N}-\mathsf{M},1}, \ldots, P_{1,\mathsf{m}} + \cdots + P_{\mathsf{N}-\mathsf{M},\mathsf{m}}$. Thus, we can apply the computing phase of the proposed scheme to the $\left(\mathsf{N}-\mathsf{M}, \frac{\mathsf{M}}{2}\right)$ non-secure problem.

Decoding phase.

The master receives $\mathsf{N_r} = \mathsf{N} - \mathsf{M} + \mathsf{m}$ worker responses, which can be categorized into two cases.

First, we consider the case where at least $\mathsf{N} - 1.5\mathsf{M} + \mathsf{m}$ responses come from workers in $[\mathsf{M}+1 : \mathsf{N}]$. In this scenario, $F_{\mathsf{m}+1}$ to $F_{2\mathsf{m}}$ can be recovered. Additionally, at least m responses from workers in $[\mathsf{M}]$ are received, providing at least m linearly independent combinations. Together with the combinations from $F_{\mathsf{m}+1}$ to $F_{2\mathsf{m}}$, this gives a total of $2\mathsf{m}$ linearly independent combinations, enabling recovery of F_1 to $F_{2\mathsf{m}}$. We then consider the second case, where the workers in $[\mathsf{M}+1 : \mathsf{N}]$ respond with $\mathsf{N}-1.5\mathsf{M}+\mathsf{m}-\mathsf{x}$ responses, providing at least $h(\mathsf{N}-\mathsf{M}, \mathsf{M}/2)-\mathsf{x}$ linearly independent combinations. Meanwhile, the workers in $[\mathsf{M}]$ will have $\mathsf{M}/2+\mathsf{x}$ responses, contributing at least $\mathsf{m}+\mathsf{x}$ linearly independent combinations. In total, these responses provide $h(\mathsf{N}-\mathsf{M}, \mathsf{M}/2)$ linearly independent combinations, which are sufficient to recover F_1 to $F_{2\mathsf{m}}$.

The number of linearly independent combinations transmitted by workers in $[\mathsf{M}+1 : \mathsf{N}]$ is $h\left(\mathsf{N}-\mathsf{M}, \frac{\mathsf{M}}{2}\right)$, which spans the space containing $F_{\mathsf{m}+1}$ to $F_{2\mathsf{m}}$. Additionally, workers in $[\mathsf{M}]$ transmit $2\mathsf{m}$ linearly independent combinations, also spanning the space of $F_{\mathsf{m}+1}$ to $F_{2\mathsf{m}}$. Therefore, the total number of transmitted linearly independent combinations is

$$\mathbf{F}=\begin{bmatrix}\mathbf{f}_1\\ \mathbf{f}_2\\ \mathbf{f}_3\\ \mathbf{f}_4\\ \mathbf{f}_5\\ \mathbf{f}_6\end{bmatrix}=\begin{bmatrix}1&1&1&1&1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&1&1&1&1&1&1&1&1&1&1&1&1\\ 0&0&0&2&2&2&2&1&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&2&2&2&2&1&1&1&1&1\\ 0&0&0&0&0&0&0&*&*&*&*&*&0&0&0&0&0&0&0&*&*&*&*&*\\ 0&0&0&0&0&0&0&*&*&*&*&*&0&0&0&0&0&0&0&*&*&*&*&*\end{bmatrix}\tag{24}$$

$h(\mathsf{N},\mathsf{M})=2\mathsf{m}+h\left(\mathsf{N}-\mathsf{M},\frac{\mathsf{M}}{2}\right)-\mathsf{m}=h\left(\mathsf{N}-\mathsf{M},\frac{\mathsf{M}}{2}\right)+\mathsf{m}$, which matches (16).

C. Scheme 3 for (17)

We first provide an example to illustrate the main idea.

Example 3. *We consider the* $(\mathsf{N},\mathsf{M})=(12,7)$ *non-secure problem, where* $\mathsf{m}=2$.

Data assignment phase. We assign the datasets as follows.

worker 1	*worker 2*	*worker 3*	*worker 4*
D_1	D_1	D_1	D_1
D_2	D_2	D_2	D_2
D_3	D_3	D_3	D_3
D_4	D_4	D_8	D_9
D_5	D_5	D_9	D_{10}
D_6	D_6	D_{10}	D_{11}
D_7	D_7	D_{11}	D_{12}

worker 5	*worker 6*	*worker 7*	*worker 8*
D_1	D_1	D_1	D_4
D_2	D_2	D_2	D_5
D_3	D_3	D_3	D_6
D_{10}	D_{11}	D_{12}	D_7
D_{11}	D_{12}	D_8	D_8
D_{12}	D_8	D_9	D_9
D_8	D_9	D_{10}	D_{10}

worker 9	*worker 10*	*worker 11*	*worker 12*
D_4	D_4	D_4	D_4
D_5	D_5	D_5	D_5
D_6	D_6	D_6	D_6
D_7	D_7	D_7	D_7
D_9	D_{10}	D_{11}	D_{12}
D_{10}	D_{11}	D_{12}	D_8
D_{11}	D_{12}	D_8	D_9

Computing phase. To minimize communication cost, each message W_k, *for* $k\in[\mathsf{K}]$, *is divided into* m *equal-length sub-messages, denoted as* $W_k=\{W_{k,j}:j\in[\mathsf{m}]\}$, *with each sub-message containing* $\frac{\mathsf{L}}{\mathsf{m}}=\frac{\mathsf{L}}{2}$ *symbols in* $\mathbb{F}_{\mathsf{q}}$. *We let each worker send a linear combination of messages, so that the master can recover* $\mathbf{F}\ [W_{1,1};\ldots;W_{12,2}]$ *from the responses of any* $\mathsf{N}_{\mathrm{r}}=12-7+2=7$ *workers, where the matrix* $\mathbf{F}$ *is shown at the top of the next page in* (24).

In this case, the entries denoted by $*$ *in the matrix cannot be simply chosen as random numbers, as in [17]. Instead, the values of* $*$ *need to be carefully constructed, and the specific reasons for this will be explained in the following details. We also define that* $[F_1;F_2;F_3;F_4;F_5;F_6]=\mathbf{F}\ [W_{1,1};\ldots;W_{12,2}]$.

First, we focus on workers 1 and 2, which have the same datasets and do not contain datasets from $[8:11]$. *We aim to recover:*

$$F_1-F_3=W_{1,1}+W_{2,1}+W_{3,1}-W_{4,1}-W_{5,1}-W_{6,1}-W_{7,1},$$
$$F_2-F_4=W_{1,2}+W_{2,2}+W_{3,2}-W_{4,2}-W_{5,2}-W_{6,2}-W_{7,2}.$$

Thus, we can have workers 1 and 2 compute a random linear combination of (F_1-F_3) *and* (F_2-F_4). *Specifically, we let worker 1 compute* X_1 *and worker 2 compute* X_2, *where:*

$$X_1=(F_1-F_3)+(F_2-F_4),$$
$$X_2=(F_1-F_3)-(F_2-F_4).$$

For the workers handling datasets from $[8:12]$, *we design their transmissions such that we can recover* F_3, F_4, F_5, *and* F_6 *from any 4 responses. For* $n\in[8,12]$, *we let worker* n *compute*

$$\mathbf{s}_n\left[\mathbf{f}_3;\mathbf{f}_4;\mathbf{f}_5;\mathbf{f}_6\right]\left[W_{1,1};W_{2,1};\ldots;W_{12,2}\right],$$

where $[\mathbf{f}_3;\mathbf{f}_4;\mathbf{f}_5;\mathbf{f}_6]$ *is shown in* (25).

We design $\mathbf{s}_n$ *according to the following steps. We focus on the datasets in* $[8:12]$ *under cyclic assignment, extracting the corresponding columns from the matrix to form a submatrix* $\mathbf{F}_1'$, *where*

$$\mathbf{F}_1'=\begin{bmatrix}1&1&1&1&1&0&0&0&0&0\\ 0&0&0&0&0&1&1&1&1&1\\ *&*&*&*&*&*&*&*&*&*\\ *&*&*&*&*&*&*&*&*&*\end{bmatrix}.\tag{26}$$

Notice that worker n *cannot compute 2 of the datasets in* $[8:12]$. *We extract the corresponding columns from the matrix to form a submatrix* $\overline{\mathbf{F}_1'}$, *where*

$$\overline{\mathbf{F}_1'}=\begin{bmatrix}1&1&0&0\\ 0&0&1&1\\ *&*&*&*\\ *&*&*&*\end{bmatrix}.\tag{27}$$

Our required $\mathbf{s}_n$ *is the left null vector of* $\overline{\mathbf{F}_1'}$. *If we choose the entries denoted by* $*$ *in* $\mathbf{F}_1'$ *to be uniformly i.i.d. in* $\mathbb{F}_q$ *as in [17], then* $\overline{\mathbf{F}_1'}$ *will be full rank with high probability, meaning there will be no non-zero left null vector. Therefore, we employ the interference alignment strategy mentioned in [20] to construct* $\overline{\mathbf{F}_1'}$.

There should be a linear combination of the columns in $\overline{\mathbf{F}_1'}$

$$[\mathbf{f}_3;\mathbf{f}_4;\mathbf{f}_5;\mathbf{f}_6]=\begin{bmatrix}0&0&0&2&2&2&2&1&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&2&2&2&2&1&1&1&1&1\\0&0&0&0&0&0&0&*&*&*&*&*&0&0&0&0&0&0&0&*&*&*&*&*\\0&0&0&0&0&0&0&*&*&*&*&*&0&0&0&0&0&0&0&*&*&*&*&*\end{bmatrix} \tag{25}$$

leading to one rank deficiency, with the form

$$\mathbf{F}_1'\mathbf{e}_n^{\mathrm{T}}=\mathbf{0}_{4\times1}, \tag{28}$$

for the first row of $\overline{\mathbf{F}_1'}$*, since* $\begin{bmatrix}1&1\end{bmatrix}[1,-1]^{\mathrm{T}}=\mathbf{0}_{1\times1}$*, the corresponding elements in* $\mathbf{e}_n$ *should be a multiple of* $(1,-1)$*, and we choose a random multiple. Similarly, the same applies to the second row of* $\overline{\mathbf{F}_1'}$*. Repeating the above steps, we obtain:*

$$\begin{aligned}\mathbf{e}_1&=(0,0,0,1,-1,0,0,0,-1,1),\\ \mathbf{e}_2&=(2,0,0,0,-2,-1,0,0,0,1),\\ \mathbf{e}_3&=(1,-1,0,0,0,2,-2,0,0,0),\\ \mathbf{e}_4&=(0,1,-1,0,0,0,1,-1,0,0),\\ \mathbf{e}_5&=(0,0,-1,1,0,0,0,2,-2,0).\end{aligned}$$

These vectors form the matrix $\mathbf{E}$*, where*

$$\begin{aligned}\mathbf{E}&=[\mathbf{e}_1;\mathbf{e}_2;\mathbf{e}_3;\mathbf{e}_4;\mathbf{e}_5]\\&=\begin{bmatrix}0&0&0&1&-1&0&0&0&-1&1\\2&0&0&0&-2&-1&0&0&0&1\\1&-1&0&0&0&2&-2&0&0&0\\0&1&-1&0&0&0&1&-1&0&0\\0&0&-1&1&0&0&0&2&-2&0\end{bmatrix}.\end{aligned} \tag{29}$$

By (28) *and* (29)*, we have*

$$\mathbf{F}_1'\mathbf{E}^{\mathrm{T}}=\mathbf{0}_{4\times5}. \tag{30}$$

This equation is solvable because $\mathbf{E}^{\mathrm{T}}$ *has dimensions* 10×5 *and is full rank; therefore, the left null space contains* $10-5=5$ *linearly independent vectors, which can span the rows of* $\mathbf{F}_1'$*. Specifically, by our construction, the first two rows are required to be in the left null space of* $\mathbf{E}^{\mathrm{T}}$*. Hence, we select any two vectors from the remaining three left null space vectors to fill the last two rows of* $\mathbf{F}_1'$*. Here, we set the matrix* $\mathbf{F}_1'$ *as*

$$\mathbf{F}_1'=\begin{bmatrix}1&1&1&1&1&0&0&0&0&0\\0&0&0&0&0&1&1&1&1&1\\2&4&2&4&2&2&1&3&4&2\\2&2&4&4&-2&3&3&1&1&-5\end{bmatrix}.$$

Since

$$\mathbf{s}_n\overline{\mathbf{F}_1'}=\mathbf{0}_{1\times4}, \tag{31}$$

we obtain the matrix $\mathbf{S}$*.*

$$\mathbf{S}=\begin{bmatrix}\mathbf{s}_1\\\vdots\\\mathbf{s}_5\end{bmatrix}=\begin{bmatrix}2&11/4&-3/4&1/4\\4&4&-2&0\\2&3&0&-1\\8&16/3&-4/3&-4/3\\6&3/2&0&-3/2\end{bmatrix}. \tag{32}$$

For $n\in[8,12]$*, we let worker* n *compute* $X_n=\mathbf{s}_n[F_3;F_4;F_5;F_6]$*.*

We finally focus on the workers in $[3:7]$*. These workers do not contain datasets from* $[4:7]$*, so we design the transmission scheme for these workers to ensure that the responses from any 4 of them can recover* $(2F_1-F_3)$*,* $(2F_2-F_4)$*,* F_5*, and* F_6*. For* $n\in[3,7]$*, we let worker* n *compute*

$$X_n=\mathbf{s}_n[2\mathbf{f}_1-\mathbf{f}_3;2\mathbf{f}_2-\mathbf{f}_4;\mathbf{f}_5;\mathbf{f}_6]\,[W_{1,1};\ldots;W_{12,2}]\,,$$

where

$$[2\mathbf{f}_1-\mathbf{f}_3;2\mathbf{f}_2-\mathbf{f}_4;\mathbf{f}_5;\mathbf{f}_6]$$

as shown in (33)*. Notice that the datasets in* $[8:12]$ *are still placed using the cyclic assignment. We extract the corresponding columns to form a new matrix* $\mathbf{F}_2'$*. It is easy to see that*

$$\mathbf{F}_2'=\mathbf{F}_1'=\begin{bmatrix}1&1&1&1&1&0&0&0&0&0\\0&0&0&0&0&1&1&1&1&1\\2&4&2&4&2&2&1&3&4&2\\2&2&4&4&-2&3&3&1&1&-5\end{bmatrix}. \tag{34}$$

For $n\in[3,7]$*, the datasets that cannot be computed by worker* n *are included in the datasets that cannot be computed by worker* $n+5$*. Combined with* (34)*, we have*

$$\mathbf{S}'=\mathbf{S}=\begin{bmatrix}\mathbf{s}_1\\\vdots\\\mathbf{s}_5\end{bmatrix}=\begin{bmatrix}2&11/4&-3/4&1/4\\4&4&-2&0\\2&3&0&-1\\8&16/3&-4/3&-4/3\\6&3/2&0&-3/2\end{bmatrix}.$$

So for $n\in[3,7]$*, we let worker* n *compute* $X_n=\mathbf{s}_n[2F_1-F_3;2F_2-F_4;F_5;F_6]$*.*

Decoding phase. From workers 1 and 2, the master can receive up to two linearly independent combinations. From the workers in $[3:7]$*, the master can receive up to four linearly independent combinations, with any four of these five workers providing linearly independent combinations. Similarly, for the workers in* $[8:12]$*, the master can receive four linearly independent combinations from any four worker responses.*

The master receives responses from $N_{\mathrm{r}}=12-7+2=7$ *workers. To recover* F_1*,* F_2*,* F_3*,* F_4*,* F_5*, and* F_6*, the master needs a total of six linearly independent combinations. We consider the following three cases:*

- *Case 1: the master does not receive responses from workers 1 and 2. In this case, the master will receive responses from any 7 workers in* $[3:12]$*. The worst case is when the master receives all responses from either the group* $[3:7]$ *or* $[8:12]$*, providing 4 linearly independent combinations, and responses from*

$$\begin{bmatrix} 2\mathbf{f}_1 - \mathbf{f}_3 \\ 2\mathbf{f}_2 - \mathbf{f}_4 \\ \mathbf{f}_5 \\ \mathbf{f}_6 \end{bmatrix} = \begin{bmatrix} 2 & 2 & 2 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 2 & 2 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 4 & 2 & 4 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 1 & 3 & 4 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 2 & 4 & 4 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 3 & 1 & 1 & -5 \end{bmatrix}. \tag{33}$$

worker 1	$\cdots$	worker y	worker y + 1	worker y + 2	$\cdots$	worker M	worker M+1	worker M + 2	$\cdots$	worker N
D_1	$\cdots$	D_1	D_1	D_1	$\cdots$	D_1	$D_{\frac{M-1}{2}+1}$	$D_{\frac{M-1}{2}+1}$	$\cdots$	$D_{\frac{M-1}{2}+1}$
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
$D_{\frac{M-1}{2}}$	$\cdots$	$D_{\frac{M-1}{2}}$	$D_{\frac{M-1}{2}}$	$D_{\frac{M-1}{2}}$	$\cdots$	$D_{\frac{M-1}{2}}$	D_M	D_M	$\cdots$	D_M
$D_{\frac{M-1}{2}+1}$	$\cdots$	$D_{\frac{M-1}{2}+1}$	D_{M+1}	D_{M+2}	$\cdots$	D_N	D_{M+1}	D_{M+2}	$\cdots$	D_N
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
D_M	$\cdots$	D_M	$D_{\frac{3M+1}{2}}$	$D_{\frac{3M+1}{2}+1}$	$\cdots$	$D_{\frac{M+1}{2}-1}$	$D_{\frac{3M-1}{2}}$	$D_{\frac{3M-1}{2}+1}$	$\cdots$	$D_{\frac{M-1}{2}-1}$

2 workers in the other group, providing 2 additional linearly independent combinations. This totals 6 linearly independent combinations.

- *Case 2: the master receives a response from either worker 1 or 2. Here, the master also receives responses from any 6 workers in* $[3:12]$. *The worst case is when all responses are received from one of the groups* $[3:7]$ *or* $[8:12]$, *providing 4 linearly independent combinations, along with a response from 1 worker in the other group, contributing 1 additional linearly independent combination. This also totals 6 linearly independent combinations.*
- *Case 3: the master receives responses from both workers 1 and 2. The master then receives responses from any 5 workers in* $[3:12]$. *In the worst case, the master receives all responses from either group* $[3:7]$ *or* $[8:12]$, *providing 4 linearly independent combinations. This again totals 6 linearly independent combinations.*

It can be seen that under the condition $\mathsf{m}=2$, *the number of totally transmitted linearly independent combinations is* $h(12,7)=12-10+4=6$, *which coincides with* (17).

We now consider the (N,M) non-secure problem, where $1.5\mathsf{M}\le\mathsf{N}<2\mathsf{M}$, M is odd, and $\mathsf{M}'\ge 2\mathsf{m}+1$. Our goal is to construct a scheme (Scheme 3) to prove (17). We also define that $\mathsf{N}=2\mathsf{M}-\mathsf{y}$.

Data assignment phase. The assignment is shown at the top of the next page. In this assignment, we split the N datasets into three sections. The first section includes $D_1,\ldots,D_t$ (The reason for choosing $t=\frac{\mathsf{M}-1}{2}$ is provided in [17].), and these datasets are assigned to workers in $[\mathsf{M}]$. The second section, containing $D_{t+1},\ldots,D_{\mathsf{M}}$, is assigned to workers in $[\mathsf{y}]\cup[\mathsf{M}+1:\mathsf{N}]$. The third section, consisting of $D_{\mathsf{M}+1},\ldots,D_{\mathsf{N}}$, is cyclically allocated to workers in $[\mathsf{y}+1:\mathsf{M}]$, where each worker receives $\mathsf{M}-t$ adjacent datasets within $\left[\frac{\mathsf{M}-1}{2}+1:\mathsf{N}\right]$. Additionally, datasets $D_{\mathsf{M}+1},\ldots,D_{\mathsf{N}}$ are assigned to workers in $[\mathsf{M}+1:\mathsf{N}]$ in a cyclic manner such that each worker receives t consecutive datasets within $\left[\frac{\mathsf{M}-1}{2}+1:\mathsf{N}\right]$.

Computing phase. We first divide each message W_k for $k\in[\mathsf{K}]$ into m equal-length, non-overlapping sub-messages, denoted as $W_k=\{W_{k,j}:j\in[\mathsf{m}]\}$. We design the computing phase so that the total number of linearly independent combinations computed by all workers is $\frac{\mathsf{M}+1}{2}-\mathsf{y}+2\mathsf{m}$. We let each worker send a linear combination of messages so that the master can recover $\mathbf{F}'\,[W_{1,1};\ldots;W_{\mathsf{N},\mathsf{m}}]$ from the responses of any $N_{\mathrm{r}}=\mathsf{N}-\mathsf{M}+\mathsf{m}$ workers, where $\mathbf{F}'$ is shown at the top of the next page in (35).

We design the demand matrix

$$\mathbf{F}=\underbrace{\begin{bmatrix}1,\ldots,1\\0,\ldots,0\end{bmatrix}}_{\mathbf{C}_1}\underbrace{\begin{bmatrix}1,\ldots,1\\2,\ldots,2\end{bmatrix}}_{\mathbf{C}_2}\underbrace{\begin{bmatrix}1,\ldots,1\\1,\ldots,1\end{bmatrix}}_{\mathbf{C}_3} \tag{36}$$

We divide matrix $\mathbf{F}$ into three column-wise sub-matrices: $\mathbf{C}_1$, containing $\frac{\mathsf{M}-1}{2}$ columns, corresponding to the messages in $\left[\frac{\mathsf{M}-1}{2}\right]$, $\mathbf{C}_2$, containing $\frac{\mathsf{M}+1}{2}$ columns, corresponding to the messages in $\left[\frac{\mathsf{M}+1}{2}:\mathsf{M}\right]$, and $\mathbf{C}_3$, containing $(\mathsf{N}-\mathsf{M})$ columns, corresponding to the messages in $[\mathsf{M}+1:\mathsf{N}]$.

Let the matrix $\mathbf{V}_i$, for $i\in[\mathsf{m}]$, be defined as

$$\mathbf{V}_i=\underbrace{\begin{bmatrix}0,\ldots,0\\\vdots\\0,\ldots,0\end{bmatrix}}_{\mathbf{C}_1}\underbrace{\begin{bmatrix}0,\ldots,0\\\vdots\\0,\ldots,0\end{bmatrix}}_{\mathbf{C}_2}\underbrace{\begin{bmatrix}*,\ldots,*\\\vdots\\*,\ldots,*\end{bmatrix}}_{\mathbf{C}_3} \tag{37}$$

In the matrix $\mathbf{V}_i$, the entries denoted by $*$ are the unknowns that we need to design. And $\mathbf{C}_1$, $\mathbf{C}_2$, and $\mathbf{C}_3$ have column counts consistent with the definitions above.

Now, we focus on the workers in $[\mathsf{y}]$, which have the same datasets and do not contain datasets from $[\mathsf{M}+1:\mathsf{N}]$. From these workers, we aim to recover:

$$F_1-F_2=W_{1,1}+\cdots+W_{t,1}-W_{t+1,1}-\cdots-W_{\mathsf{N},1},$$
$$\vdots$$
$$F_{2\mathsf{m}-1}-F_{2\mathsf{m}}=W_{1,\mathsf{m}}+\cdots+W_{t,\mathsf{m}}-W_{t+1,\mathsf{m}}-\cdots-W_{\mathsf{N},\mathsf{m}}.$$

Thus, we let the workers in $[\mathsf{y}]$ each compute a random linear combination of the m linearly independent combinations $(F_1-F_2),\ldots,(F_{2\mathsf{m}-1}-F_{2\mathsf{m}})$.

For the workers in $[M+1:N]$ that do not contain datasets from $[t]$, we design their transmissions so that the

$$\mathbf{F}' = \begin{bmatrix} \mathbf{f}_1 \\ \vdots \\ \mathbf{f}_{\frac{\mathsf{M}+1}{2}-\mathsf{y}+2\mathsf{m}} \end{bmatrix} = \begin{bmatrix} (\mathbf{F})_{2\times\mathsf{N}} & \mathbf{0}_{2\times\mathsf{N}} & \cdots & \mathbf{0}_{2\times\mathsf{N}} \\ \mathbf{0}_{2\times\mathsf{N}} & (\mathbf{F})_{2\times\mathsf{N}} & \cdots & \mathbf{0}_{2\times\mathsf{N}} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0}_{2\times\mathsf{N}} & \mathbf{0}_{2\times\mathsf{N}} & \cdots & (\mathbf{F})_{2\times\mathsf{N}} \\ (\mathbf{V}_1)_{(\frac{\mathsf{M}+1}{2}-\mathsf{y})\times\mathsf{N}} & (\mathbf{V}_2)_{(\frac{\mathsf{M}+1}{2}-\mathsf{y})\times\mathsf{N}} & \cdots & (\mathbf{V}_\mathsf{m})_{(\frac{\mathsf{M}+1}{2}-\mathsf{y})\times\mathsf{N}} \end{bmatrix}_{(\frac{\mathsf{M}+1}{2}-\mathsf{y}+2\mathsf{m})\times\mathsf{N}\mathsf{m}} . \tag{35}$$

$$\mathbf{F}_\mathbf{s} = \begin{bmatrix} \mathbf{1}_{1\times(\mathsf{N}-\mathsf{M})} & \mathbf{0}_{1\times(\mathsf{N}-\mathsf{M})} & \cdots & \mathbf{0}_{1\times(\mathsf{N}-\mathsf{M})} \\ \mathbf{0}_{1\times(\mathsf{N}-\mathsf{M})} & \mathbf{1}_{1\times(\mathsf{N}-\mathsf{M})} & \cdots & \mathbf{0}_{1\times(\mathsf{N}-\mathsf{M})} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0}_{1\times(\mathsf{N}-\mathsf{M})} & \mathbf{0}_{1\times(\mathsf{N}-\mathsf{M})} & \cdots & \mathbf{1}_{1\times(\mathsf{N}-\mathsf{M})} \\ (\mathbf{V}'_1)_{(\frac{\mathsf{M}+1}{2}-\mathsf{y})\times(\mathsf{N}-\mathsf{M})} & (\mathbf{V}'_2)_{(\frac{\mathsf{M}+1}{2}-\mathsf{y})\times(\mathsf{N}-\mathsf{M})} & \cdots & (\mathbf{V}'_\mathsf{m})_{(\frac{\mathsf{M}+1}{2}-\mathsf{y})\times(\mathsf{N}-\mathsf{M})} \end{bmatrix}_{(\frac{\mathsf{M}+1}{2}-\mathsf{y}+\mathsf{m})\times\mathsf{m}(\mathsf{N}-\mathsf{M})} \tag{39}$$

master can recover $F_2, F_4, \ldots, F_{2\mathsf{m}}, F_{2\mathsf{m}+1}, \ldots, F_{\frac{\mathsf{M}+1}{2}-\mathsf{y}+2\mathsf{m}}$ from any $\frac{\mathsf{M}+1}{2}-\mathsf{y}+\mathsf{m}$ responses. For $n \in [\mathsf{M}+1, \mathsf{N}]$, we let worker n compute

$$\mathbf{s}_n[F_2; F_4; \ldots; F_{2\mathsf{m}}; F_{2\mathsf{m}+1}; \ldots; F_{\frac{\mathsf{M}+1}{2}-\mathsf{y}+2\mathsf{m}}] \times [W_{1,1}; W_{2,1}; \ldots; W_{\mathsf{N},\mathsf{m}}]. \tag{38}$$

We design $\mathbf{s}_n$ as follows. Notice that $W_1, \ldots, W_{\frac{\mathsf{M}-1}{2}}$ can be computed by worker n; and that in the linear combination (38), the coefficients of $W_{\frac{\mathsf{M}-1}{2}+1}, \ldots, W_\mathsf{M}$ are 0. Hence, in order to guarantee that in (38) the coefficients of the messages which worker n cannot compute are 0, we only need to consider the messages in $W_{\mathsf{M}+1}, \ldots, W_\mathsf{N}$, corresponding to the columns in $\mathbf{C}_3$ in the matrix, with a total of $\mathsf{m}(\mathsf{N}-\mathsf{M})$ columns. By extracting the corresponding rows and columns from the matrix $\mathbf{F}'$, we obtain $\mathbf{F}_\mathbf{s}$, as shown at the top of the next page in (39), where $\mathbf{V}'_i$ is the sub-matrix formed by the last $\mathsf{N}-\mathsf{M}$ columns of $\mathbf{V}_i$. Notice that the datasets in $[\mathsf{M}+1:\mathsf{N}]$ are placed in a cyclic assignment among the workers in $[\mathsf{M}+1:\mathsf{N}]$. Each worker cannot compute $\mathsf{N}-\mathsf{M}-\frac{\mathsf{M}-1}{2}$ of the datasets in $[\mathsf{M}+1:\mathsf{N}]$. By extracting the corresponding columns from the matrix $\mathbf{F}_\mathbf{s}$, we form a submatrix $\mathbf{F}_{\mathbf{s}_n}$, with dimensions $\left(\frac{\mathsf{M}+1}{2}-\mathsf{y}+\mathsf{m}\right) \times \mathsf{m}\left(\mathsf{N}-\frac{3\mathsf{M}-1}{2}\right)$. Our desired vector $\mathbf{s}_n$ is then the left null vector of this matrix. If the number of rows in the matrix $\mathbf{F}_{\mathbf{s}_n}$ is greater than the number of columns, we can directly assign random values to the entries denoted by $*$ as in [17] to obtain $\mathbf{s}_n$. If the number of rows is less than or equal to the number of columns, we apply the interference alignment strategy from [20] to design the values of the $*$ entries in order to obtain $\mathbf{s}_n$.

Finally, we focus on the workers in $[\mathsf{y}+1:\mathsf{M}]$. We design their transmissions so that the master can recover $2F_1 - F_2, 2F_3 - F_4, \ldots, 2F_{2\mathsf{m}-1} - F_{2\mathsf{m}}, F_{2\mathsf{m}+1}, \ldots, F_{\frac{\mathsf{M}+1}{2}-\mathsf{y}+2\mathsf{m}}$ from any $\frac{\mathsf{M}+1}{2}-\mathsf{y}+\mathsf{m}$ responses. For $n \in [\mathsf{y}+1, \mathsf{M}]$, we also let worker n compute

$$\mathbf{s}_n[2F_1 - F_2; \ldots; 2F_{2\mathsf{m}-1} - F_{2\mathsf{m}}; F_{2\mathsf{m}+1}; \ldots; F_{\frac{\mathsf{M}+1}{2}-\mathsf{y}+2\mathsf{m}}] \times [W_{1,1}; W_{2,1}; \ldots; W_{\mathsf{N},\mathsf{m}}]. \tag{40}$$

We design $\mathbf{s}_n$ as follows. Notice that $W_1, \ldots, W_{\frac{\mathsf{M}-1}{2}}$ can be computed by worker n; and that in the linear combination (40) the coefficients of $W_{\frac{\mathsf{M}-1}{2}+1}, \ldots, W_\mathsf{M}$ are 0. Hence, in order to guarantee that in (40) the coefficients of the messages which worker n cannot compute are 0, we only need to consider the messages in $W_{\mathsf{M}+1}, \ldots, W_\mathsf{N}$,corresponding to the columns in $\mathbf{C}_3$ in the matrix, with a total of $\mathsf{m}(\mathsf{N}-\mathsf{M})$ columns.By extracting the corresponding rows and columns from the matrix $\mathbf{F}'$, we obtain $\mathbf{F}'_\mathbf{s}$, as shown at the top of the next page in (41). It is easy to observe that $\mathbf{F}'_\mathbf{s} = \mathbf{F}_\mathbf{s}$.Notice that the datasets in $[\mathsf{M}+1, \mathsf{N}]$ are also cyclically assigned to the workers in $[\mathsf{y}+1:\mathsf{M}]$. For $n \in [\mathsf{y}+1:\mathsf{M}]$, worker n cannot compute $\mathsf{N}-\frac{3\mathsf{M}+1}{2}$ datasets from $[\mathsf{M}+1, \mathsf{N}]$, and these datasets are also not computable by worker $n+\mathsf{M}-\mathsf{y}$. Since $\mathbf{F}'_\mathbf{s} = \mathbf{F}_\mathbf{s}$, we can set $\mathbf{s}_n = \mathbf{s}_{n+\mathsf{M}-\mathsf{y}}$.

Decoding phase. The master receives responses from $N_\mathrm{r} = \mathsf{N}-\mathsf{M}+\mathsf{m}$ workers, of which x responses are from workers in $[\mathsf{y}]$. Depending on the value of x, we can categorize the scenarios into the following two cases:

- *Case 1:* $\mathsf{x} \le \mathsf{m}$. In this case, the master receives x linearly independent combinations sent by the workers in $[\mathsf{y}]$. The master still needs to collect an additional $2\mathsf{m}-\mathsf{x}+\mathsf{N}-\frac{3\mathsf{M}-1}{2}$ linearly independent combinations from any $(\mathsf{N}-\mathsf{M}-\mathsf{x}+\mathsf{m})$ workers in $[\mathsf{y}+1:\mathsf{N}]$. In the worst-case scenario, the workers in one of the groups, either $[\mathsf{y}+1:\mathsf{M}]$ or $[\mathsf{M}+1:\mathsf{N}]$, all respond, transmitting a total of $\mathsf{m}+\mathsf{N}-\frac{3\mathsf{M}-1}{2}$ linearly independent combinations. Additionally, $\mathsf{m}-\mathsf{x}$ workers from the other group respond, providing another $\mathsf{m}-\mathsf{x}$ linearly independent combinations.
  Together, this results in a total of $2\mathsf{m}+\mathsf{N}-\frac{3\mathsf{M}-1}{2}$ linearly independent combinations, sufficient to fully recover the demand matrix $\mathbf{F}'$.
- *Case 2:* $\mathsf{m} < \mathsf{x} \le \mathsf{y}$. In this case, the master receives m linearly independent combinations sent by the workers in $[\mathsf{y}]$. We consider the worst-case scenario, where $x = \mathsf{y}$, meaning that the master still needs to collect an additional $\mathsf{m}+\mathsf{N}-\frac{3\mathsf{M}-1}{2}$ linearly independent combinations from any $2\mathsf{N}-3\mathsf{M}+\mathsf{m}$ workers in $[\mathsf{y}+1:\mathsf{N}]$. Since $\mathsf{N} \ge \frac{3\mathsf{M}+1}{2}$, it follows that $2\mathsf{N}-3\mathsf{M}+\mathsf{m} \ge \mathsf{m}+\mathsf{N}-\frac{3\mathsf{M}-1}{2}$. Additionally,

$$\mathbf{F}'_{\mathbf{s}} = \begin{bmatrix} \mathbf{1}_{1\times(\mathsf{N}-\mathsf{M})} & \mathbf{0}_{1\times(\mathsf{N}-\mathsf{M})} & \cdots & \mathbf{0}_{1\times(\mathsf{N}-\mathsf{M})} \\ \mathbf{0}_{1\times(\mathsf{N}-\mathsf{M})} & \mathbf{1}_{1\times(\mathsf{N}-\mathsf{M})} & \cdots & \mathbf{0}_{1\times(\mathsf{N}-\mathsf{M})} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0}_{1\times(\mathsf{N}-\mathsf{M})} & \mathbf{0}_{1\times(\mathsf{N}-\mathsf{M})} & \cdots & \mathbf{1}_{1\times(\mathsf{N}-\mathsf{M})} \\ (\mathbf{V}'_1)_{(\frac{\mathsf{M}+1}{2}-\mathsf{y})\times(\mathsf{N}-\mathsf{M})} & (\mathbf{V}'_2)_{(\frac{\mathsf{M}+1}{2}-\mathsf{y})\times(\mathsf{N}-\mathsf{M})} & \cdots & (\mathbf{V}'_{\mathsf{m}})_{(\frac{\mathsf{M}+1}{2}-\mathsf{y})\times(\mathsf{N}-\mathsf{M})} \end{bmatrix}_{(\frac{\mathsf{M}+1}{2}-\mathsf{y}+\mathsf{m})\times\mathsf{m}(\mathsf{N}-\mathsf{M})}. \tag{41}$$

any $\mathsf{m}+\mathsf{N}-\frac{3\mathsf{M}-1}{2}$ workers in either of the groups $[\mathsf{y}+1:\mathsf{M}]$ or $[\mathsf{M}+1:\mathsf{N}]$ compute linearly independent combinations. Therefore, it is possible to receive $\mathsf{m}+\mathsf{N}-\frac{3\mathsf{M}-1}{2}$ linearly independent combinations from any $2\mathsf{N}-3\mathsf{M}+\mathsf{m}$ workers in $[\mathsf{y}+1:\mathsf{N}]$.

Thus, the master obtains a total of $2\mathsf{m}+\mathsf{N}-\frac{3\mathsf{M}-1}{2}$ linearly independent combinations, which suffices to reconstruct the demand matrix $\mathbf{F}'$.

By the above scheme, the number of linearly independent transmissions by all workers is equal to the number of rows in $\mathbf{F}'$, i.e., $\frac{\mathsf{M}+1}{2}-\mathsf{y}+2\mathsf{m}=\mathsf{N}-\frac{3\mathsf{M}-1}{2}+2\mathsf{m}$, coinciding with (17).

D. Scheme 4 for (18)

In this section, we examine the case where $\mathsf{M}<\mathsf{N}<1.5\mathsf{M}$, with $\mathsf{M}\geq 2\mathsf{m}$, and construct a recursive scheme (Scheme 4) that aims to demonstrate (18). This recursive scheme builds on the solution for the $(\mathsf{M},2\mathsf{M}-\mathsf{N})$ non-secure problem, assuming that this solution has been previously developed, and provides a total of $h(\mathsf{M},2\mathsf{M}-\mathsf{N})$ linearly independent combinations of messages.

Data assignment phase. We begin by allocating datasets $D_1,\ldots,D_{\mathsf{N}-\mathsf{M}}$ to each worker in $[\mathsf{M}]$. Subsequently, datasets $D_{\mathsf{N}-\mathsf{M}+1},\ldots,D_{\mathsf{N}}$ are assigned to each worker in $[\mathsf{M}+1:\mathsf{N}]$, ensuring that each worker in $[\mathsf{M}+1:\mathsf{N}]$ has access to M datasets, while each worker in $[\mathsf{M}]$ holds fewer than M datasets, specifically $\mathsf{N}-\mathsf{M}$. Additionally, each dataset in $[\mathsf{N}-\mathsf{M}+1:\mathsf{N}]$ is assigned to $\mathsf{N}-\mathsf{M}<\mathsf{M}$ workers. Therefore, in the next step, we assign each dataset D_k for $k\in[\mathsf{N}-\mathsf{M}+1:\mathsf{N}]$ to $2\mathsf{M}-\mathsf{N}$ workers in $[\mathsf{M}]$, such that each worker in $[\mathsf{M}]$ obtains $2\mathsf{M}-\mathsf{N}$ datasets in $[\mathsf{N}-\mathsf{M}+1:\mathsf{N}]$. This allocation follows the assignment phase of the proposed scheme for the $(\mathsf{M},2\mathsf{M}-\mathsf{N})$ non-secure problem.

Computing phase. Each message W_k for $k\in[\mathsf{K}]$ is divided into m equal-length, non-overlapping sub-messages, noted as $W_k=\{W_{k,j}:j\in[\mathsf{m}]\}$. First, we focus on the $(\mathsf{M},2\mathsf{M}-\mathsf{N})$ non-secure problem, where the messages are defined as $W'_1,\ldots,W'_{\mathsf{M}}$. Under this scheme, each worker computes a linear combination of these M messages. Letting the total number of linearly independent combinations across all workers be $h(\mathsf{M},2\mathsf{M}-\mathsf{N})$, we can represent these $h(\mathsf{M},2\mathsf{M}-\mathsf{N})$ combinations as

$$\mathbf{F}'\ [W'_{1,1};\ldots;W'_{\mathsf{M},\mathsf{m}}]. \tag{42}$$

Each transmission from a worker $n'\in[\mathsf{M}]$ is represented as

$$\mathbf{s}_{n'}\ \mathbf{F}'\ [W'_{1,1};\ldots;W'_{\mathsf{M},\mathsf{m}}].$$

Returning to the (N,M) non-secure problem, we set up the worker transmissions so that, together, they produce $h(\mathsf{M},2\mathsf{M}-\mathsf{N})$ linearly independent combinations, represented as $\mathbf{F}[W_{1,1};\ldots;W_{\mathsf{N},\mathsf{m}}]$, where $\mathbf{F}$ is shown at the top of the next page in (43). Each row in the submatrix $\mathbf{A}$ contains identical random numbers. For each $i\in[\mathsf{m}]$, $\mathbf{V}_i$ shares elements with the corresponding elements in matrix $\mathbf{F}$, allowing matrix $\mathbf{F}$ to be viewed as a submatrix extracted from $\mathbf{F}'$ based on the columns in each $\mathbf{V}_i$.

For each worker $n\in[\mathsf{M}]$, by the construction in the assignment phase, it has access to datasets $D_1,\ldots,D_{\mathsf{N}-\mathsf{M}}$, while the assignment for datasets $D_{\mathsf{N}-\mathsf{M}+1},\ldots,D_{\mathsf{N}}$ is based on the assignment for the $(\mathsf{M},2\mathsf{M}-\mathsf{N})$ non-secure problem. Thus, worker n computes $\mathbf{s}_n\ \mathbf{F}\ [W_1;\ldots;W_{\mathsf{N}}]$, where $\mathbf{s}_n$ corresponds to the transmission vector of worker n in the $(\mathsf{M},2\mathsf{M}-\mathsf{N})$ problem.

For each worker $n\in[\mathsf{M}+1:\mathsf{N}]$, it cannot compute $W_1,\ldots,W_{\mathsf{N}-\mathsf{M}}$, which corresponds to the columns of submatrix $\mathbf{A}$ in $\mathbf{F}$. Extracting these columns creates a matrix $\mathbf{F}''$ of size $h(\mathsf{M},2\mathsf{M}-\mathsf{N})\times\mathsf{m}(\mathsf{N}-\mathsf{M})$, whose rank is m. Consequently, the left null space of $\mathbf{F}''$ includes $h(\mathsf{M},2\mathsf{M}-\mathsf{N})-\mathsf{m}$ independent vectors. We let the transmission vector $\mathbf{s}_n$ for each worker n be a random linear combination of these vectors, using uniformly i.i.d. coefficients from $\mathbb{F}_{\mathsf{q}}$, so that each worker n computes $\mathbf{s}_n\mathbf{F}[W_1;\ldots;W_{\mathsf{N}}]$.

Decoding phase. Suppose $\mathcal{A}$ denotes the set of responding workers with $|\mathcal{A}|=\mathsf{N}_{\mathrm{r}}$, $\mathcal{A}_1=\mathcal{A}\cap[\mathsf{M}]$, and $\mathcal{A}_2=\mathcal{A}\setminus[\mathsf{M}]$, where $|\mathcal{A}_2|\leq\mathsf{N}-\mathsf{M}=\mathsf{N}_{\mathrm{r}}-\mathsf{m}$. When $\mathsf{M}\geq\mathsf{N}-\mathsf{M}+\mathsf{m}=\mathsf{N}_{\mathrm{r}}$, if $|\mathcal{A}_2|=0$, the master can recover the first m rows of $\mathbf{F}$, leveraging the decodability of the $(\mathsf{M},2\mathsf{M}-\mathsf{N})$ scheme.

When $0<|\mathcal{A}_2|\leq\mathsf{N}-\mathsf{M}$ and $\mathcal{A}_1$ alone does not suffice, the master receives at least m independent responses from $\mathcal{A}_1$. Let the linearly independent combinations from $\mathcal{A}_1$ be denoted by λ_1. In addition to the responses from $\mathcal{A}_1$, the master will receive responses from at least $h(\mathsf{M},2\mathsf{M}-\mathsf{N})-\lambda_1$ workers in $[\mathsf{M}+1:\mathsf{N}]$. Since each worker sends an independent combination of $h(\mathsf{M},2\mathsf{M}-\mathsf{N})-\mathsf{m}$, any $h(\mathsf{M},2\mathsf{M}-\mathsf{N})-\lambda_1$ responses in $[\mathsf{M}+1:\mathsf{N}]$ are linearly independent with high probability. Combining with λ_1, this totals $h(\mathsf{M},2\mathsf{M}-\mathsf{N})$ combinations, allowing the master to recover $\mathbf{F}\left[W_{1,1};\ldots;W_{\mathsf{N},\mathsf{m}}\right]$.

In summary, $h(\mathsf{N},\mathsf{M})=h(\mathsf{M},2\mathsf{M}-\mathsf{N})$, consistent with (18).

V. Extensions

In this section, we consider a secure distributed linearly separable computation problem where the demand is unknown, unlike the case with the known demand discussed earlier. We will discuss the storage key size for each worker, i.e., the

$$\mathbf{F}=\left[\begin{array}{c:c:c:c:c:c:c} \underset{h(\mathsf{M},2\mathsf{M}-\mathsf{N})-\mathsf{m})\times(\mathsf{N}-\mathsf{M})}{\mathbf{A}} & \underset{h(\mathsf{M},2\mathsf{M}-\mathsf{N})-\mathsf{m})\times\mathsf{M}}{\mathbf{V}_1} & \underset{h(\mathsf{M},2\mathsf{M}-\mathsf{N})-\mathsf{m})\times(\mathsf{N}-\mathsf{M})}{\mathbf{A}} & \underset{h(\mathsf{M},2\mathsf{M}-\mathsf{N})-\mathsf{m})\times\mathsf{M}}{\mathbf{V}_2} & \cdots & \underset{h(\mathsf{M},2\mathsf{M}-\mathsf{N})-\mathsf{m})\times(\mathsf{N}-\mathsf{M})}{\mathbf{A}} & \underset{h(\mathsf{M},2\mathsf{M}-\mathsf{N})-\mathsf{m})\times\mathsf{M}}{\mathbf{V}_\mathsf{m}} \\ \hline \mathbf{0}_{1\times(\mathsf{N}-\mathsf{M})} & \mathbf{0}_{1\times\mathsf{M}} & \mathbf{0}_{1\times(\mathsf{N}-\mathsf{M})} & \mathbf{0}_{1\times\mathsf{M}} & \cdots & \mathbf{0}_{1\times(\mathsf{N}-\mathsf{M})} & \mathbf{0}_{1\times\mathsf{M}} \\ \mathbf{0}_{1\times(\mathsf{N}-\mathsf{M})} & \mathbf{0}_{1\times\mathsf{M}} & \mathbf{1}_{1\times(\mathsf{N}-\mathsf{M})} & \mathbf{1}_{1\times\mathsf{M}} & \cdots & \mathbf{0}_{1\times(\mathsf{N}-\mathsf{M})} & \mathbf{0}_{1\times\mathsf{M}} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0}_{1\times(\mathsf{N}-\mathsf{M})} & \mathbf{0}_{1\times\mathsf{M}} & \mathbf{0}_{1\times(\mathsf{N}-\mathsf{M})} & \mathbf{0}_{1\times\mathsf{M}} & \cdots & \mathbf{1}_{1\times(\mathsf{N}-\mathsf{M})} & \mathbf{1}_{1\times\mathsf{M}} \end{array}\right]_{h(\mathsf{M},2\mathsf{M}-\mathsf{N})\times\mathsf{mN}}. \tag{43}$$

individual key size in the secure distributed linearly separable computation problem with unknown demand. We use Q_s to represent the individual key size, given by

$$\mathsf{Q}_\mathrm{s}=\frac{H(Q_n)}{\mathsf{L}}. \tag{44}$$

Theorem 3. *For the $(\mathsf{K},\mathsf{N},\mathsf{N}_\mathrm{r},\mathsf{K}_\mathrm{c},\mathsf{m})$ secure distributed linearly separable computation problem with unknown demand, the optimal communication cost that can be achieved is the same as that with known demand. This can be expressed as*

$$\mathsf{R}^\star_{ach}=\mathsf{R}^\star,$$

where $\mathsf{R}^\star$ represents the optimal communication cost required for secure aggregation with known demand.

Theorem 4. *For the $(\mathsf{K},\mathsf{N},\mathsf{N}_\mathrm{r},\mathsf{K}_\mathrm{c},\mathsf{m})$ secure distributed linearly separable computation problem, by defining $\mathsf{u}:=\lceil\frac{\mathsf{N}}{\mathsf{K}}\mathsf{K}_\mathrm{c}\rceil$, when $\mathsf{N}\geq\mathsf{u}(\mathsf{N}_\mathrm{r}-\mathsf{m}-\mathsf{u}+1)+\frac{\mathsf{m}+\mathsf{u}-1}{\mathsf{u}}$, the minimum storage key size for each worker, or the minimum individual key size, can be expressed as*

$$\mathsf{Q}^*_\mathrm{s}=\frac{\mathsf{R}^\star_{ach}}{\mathsf{N}_\mathrm{r}}=\frac{\mathsf{R}^\star}{\mathsf{N}_\mathrm{r}}. \tag{45}$$

Notice that in our scheme, the communication cost for key sharing is equal to the individual key size stored by each worker, so we only need to focus on the individual key size. The applicability of (45) *will be analyzed in Appendix C.*

Remark 3 (The minimum communication cost under cyclic data assignment)**.** For the $(\mathsf{K},\mathsf{N},\mathsf{N}_\mathrm{r},\mathsf{K}_\mathrm{c},\mathsf{m})$ distributed linearly separable computation problem under the constraint of the data cyclic assignment, as shown in [20, Theorem 4], when K/N is an integer, the optimal communication cost can be expressed as

$$\mathsf{R}^\star_\mathrm{cyc}=\frac{\mathsf{N}_\mathrm{r}\mathsf{K}_\mathrm{c}}{\mathsf{m}+\mathsf{u}-1}. \tag{46}$$

Novel Achievement Scheme For Theorem 4

We first provide an example to illustrate the main idea, and compare it with the computation schemes in [17] to demonstrate that our proposed scheme reduces individual key size.

Example 4. *We consider a secure distributed linearly separable computation problem with parameters $(\mathsf{K},\mathsf{N},\mathsf{N}_\mathrm{r},\mathsf{K}_\mathrm{c},\mathsf{m})=(5,5,4,2,1)$, where $\mathsf{M}=\frac{\mathsf{K}}{\mathsf{N}}(\mathsf{N}-\mathsf{N}_\mathrm{r}+\mathsf{m})=2$. We assume the demand matrix is*

$$\mathbf{F}=\begin{bmatrix} f_{1,1} & f_{1,2} & f_{1,3} & f_{1,4} & f_{1,5} \\ f_{2,1} & f_{2,2} & f_{2,3} & f_{2,4} & f_{2,5} \end{bmatrix}=\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 & 5 \end{bmatrix}. \tag{47}$$

Data assignment phase. For simplicity, we discuss the cyclic data assignment shown in the following.

Worker 1	*Worker 2*	*Worker 3*	*Worker 4*	*Worker 5*
D_1	D_2	D_3	D_4	D_5
D_2	D_3	D_4	D_5	D_1

According to Theorem 2, the minimum key size required to achieve the optimal communication cost in this example, denoted by $\eta^\star_{cyc}$, is 1.

Computing phase. To achieve the optimal communication cost $\mathsf{R}^*=\mathsf{N}_\mathrm{r}\frac{\mathsf{K}_\mathrm{c}}{\mathsf{m}+\mathsf{K}_\mathrm{c}-1}=4$, each worker is required to send $\frac{\mathsf{K}_\mathrm{c}}{\mathsf{m}+\mathsf{K}_\mathrm{c}-1}=1$ coded messages. Each message W_k, where $k\in[\mathsf{K}]$, consists of L symbols from the finite field $\mathbb{F}_\mathsf{q}$. Additionally, we generate $\mathsf{Q}=\frac{\mathsf{N}_\mathrm{r}\mathsf{K}_\mathrm{c}}{\mathsf{m}+\mathsf{K}_\mathrm{c}-1}-\mathsf{K}_\mathrm{c}=2$ dataset-independent keys with L length, $\{K_1,K_2\}$.

From $\mathsf{N}_\mathrm{r}=4$ workers, the master receives $\mathsf{N}_\mathrm{r}\frac{\mathsf{K}_\mathrm{c}}{\mathsf{m}+\mathsf{K}_\mathrm{c}-1}=4$ coded messages, which contain the desired $\mathsf{K}_\mathrm{c}=2$ linear combinations, $\mathbf{F}[W_1;\ldots;W_K]$. Consequently, we introduce $\mathsf{v}=\mathsf{N}_\mathrm{r}-\mathsf{K}_\mathrm{c}=4-2=2$ virtual demand as the additional linear combinations of messages. Thus, the complete demand matrix can be written as:

$$\mathbf{F}_1=\left[\begin{array}{ccccc:cc} 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 2 & 3 & 4 & 5 & 0 & 0 \\ a_1 & a_2 & a_3 & a_4 & a_5 & 1 & 0 \\ a_6 & a_7 & a_8 & a_9 & a_{10} & 0 & 1 \end{array}\right], \tag{48}$$

where the last two rows of the matrix $\mathbf{F}_l$ represent the virtual demand, and $\mathbf{F}_r$ denotes the columns corresponding to the keys, and we design the last two rows of $\mathbf{F}_r$ as an identity matrix.

Our coding strategy is to let each worker $n\in[5]$ send $\frac{\mathsf{K}_\mathrm{c}}{\mathsf{m}+\mathsf{K}_\mathrm{c}-1}=1$ coded messages

$$\mathbf{s}^n\mathbf{F}_1[W_1;W_2;W_3;W_4;W_5;K_1;K_2]. \tag{49}$$

Denote the encoding matrix as

$$\mathbf{S}_1=\begin{bmatrix}\mathbf{s}^1\\ \mathbf{s}^2\\ \mathbf{s}^3\\ \mathbf{s}^4\\ \mathbf{s}^5\end{bmatrix}=\left[\begin{array}{cc:cc} s^1_1 & s^1_2 & b^1_1 & b^1_2 \\ s^2_1 & s^2_2 & b^2_1 & b^2_2 \\ s^3_1 & s^3_2 & b^3_1 & b^3_2 \\ s^4_1 & s^4_2 & b^4_1 & b^4_2 \\ s^5_1 & s^5_2 & b^5_1 & b^5_2 \end{array}\right]. \tag{50}$$

For each $n\in[\mathsf{N}]$, worker n cannot compute $D_{Mod(n+2,5)}, D_{Mod(n+3,5)}, D_{Mod(n+4,5)}$, thus we have

$$\begin{aligned}\mathbf{s}^n\mathbf{c}_{Mod(n+2,5)}&=0,\\ \mathbf{s}^n\mathbf{c}_{Mod(n+3,5)}&=0,\\ \mathbf{s}^n\mathbf{c}_{Mod(n+4,5)}&=0,\end{aligned} \tag{51}$$

where $\mathbf{c}_k$ represents the k^{th} column of the matrix $\mathbf{F}_1$.

In the following, we will introduce our proposed scheme and compare it with the previous computation schemes in terms of individual key size. The proposed computing scheme contains three main steps:

Step 1: We assign values to $\mathbf{S}_r$ in the following form, which correspond to the coefficients of the key linear combinations sent by the workers:

$$\mathbf{S}_r = \begin{bmatrix} b_1^1 & b_2^1 \\ b_1^2 & b_2^2 \\ b_1^3 & b_2^3 \\ b_1^4 & b_2^4 \\ b_1^5 & b_2^5 \end{bmatrix} = \begin{bmatrix} * & * \\ * & * \\ * & * \\ * & * \\ * & * \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 3 \\ 3 & 2 \\ 2 & 3 \\ 9 & 1 \end{bmatrix}, \tag{52}$$

where each '$*$' represents a uniformly i.i.d. symbol over $\mathbb{F}_\mathsf{q}$. Once these values are determined, they remain fixed.

Step 2: Next, we focus on the variables in matrices $\mathbf{F}_1$ and $\mathbf{S}_1$. As indicated in (51), for each $n \in [5]$, worker n cannot compute three datasets, corresponding to 3 columns in the matrix $\mathbf{F}_1$, which yields three equations. Additionally, each worker n will transmit one linear combination of messages, resulting in a total of $5 \times 3 \times 1 = 15$ equations. There are 20 remaining variables:

$$a_1, \ldots, a_{10}, s_1^1, \ldots, s_1^5, s_2^1, \ldots, s_2^5. \tag{53}$$

Since the number of variables exceeds the number of equations, we first assign values to $20 - 15 = 5$ variables. We uniformly assign values i.i.d. over $\mathbb{F}_\mathsf{q}$ to s_1^1, s_2^2, s_1^3, s_2^4, and s_1^5 i.e., the values 1 to 5 are assigned to them sequentially. The specific method for selecting these variables will be presented later. We solve the remaining 15 variables according to the 15 linear equations. Then we have matrices $\mathbf{F}_1$ and $\mathbf{S}_1$:

$$\mathbf{F}_1 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 2 & 3 & 4 & 5 & 0 & 0 \\ -\frac{26}{11} & -\frac{261}{55} & -\frac{76}{11} & -\frac{199}{22} & -\frac{126}{11} & 1 & 0 \\ -\frac{14}{11} & -\frac{56}{55} & -\frac{10}{11} & -\frac{23}{22} & -\frac{10}{11} & 0 & 1 \end{bmatrix}, \tag{54}$$

$$\mathbf{S}_1 = \begin{bmatrix} 1 & \frac{25}{11} & 1 & 1 \\ \frac{46}{11} & 2 & 1 & 3 \\ 3 & \frac{73}{11} & 3 & 2 \\ \frac{50}{11} & 4 & 2 & 3 \\ 5 & \frac{213}{11} & 9 & 1 \end{bmatrix}. \tag{55}$$

Step 3: For each $n \in [5]$, we let each worker n compute and send one linear combination of messages, $\mathbf{s}^n\mathbf{F}_1[W_1; W_2; W_3; W_4; W_5; K_1; K_2]$. Notice that the individual key set stored by the workers is $\{K_1 + K_2, K_1 + 3K_2, 3K_1 + 2K_2, 2K_1 + 3K_2, 9K_1 + K_2\}$, as shown in Fig. 3.

Decoding phase. For any set of $N_\mathrm{r} = 4$ workers whose responses are received by the master, we extract the corresponding three rows from the matrix $\mathbf{S}$ to form a new matrix. It can be observed that this new matrix is full rank, indicating that the coding scheme is decodable. This implies that with

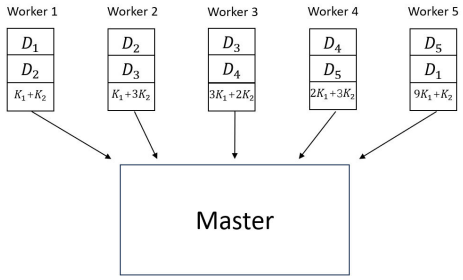


Fig. 3: The individual key size of proposed scheme is 1.

responses from any $N_\mathrm{r} = 4$ workers, the master can recover $\mathbf{F}\left[W_1; W_2; W_3; W_4; W_5\right]$.

We then attempt the computation scheme proposed in [17], which we refer to as the baseline scheme. In the baseline scheme, when the demand matrix changes, the key combinations each worker needs to send also change. We randomly assign values to the virtual demands in the matrix $\mathbf{F}_2$, as shown below:

$$\mathbf{F}_2 = \underbrace{\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 & 5 \\ 4 & 1 & 1 & 4 & 8 \\ 3 & 7 & 8 & 7 & 9 \end{bmatrix}}_{\mathbf{F}_l} \underbrace{\begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}}_{\mathbf{F}_r}. \tag{56}$$

For each $n \in [\mathsf{N}]$, we extract the columns corresponding to the data set that worker n cannot compute in $\mathbf{F}_l$, forming a new matrix. By calculating the left null vector of this matrix, we can obtain $\mathbf{s}^n$. This gives us the matrix $\mathbf{S}_2$, as shown in (57):

$$\mathbf{S}_2 = \begin{bmatrix} \mathbf{s}^1 \\ \mathbf{s}^2 \\ \mathbf{s}^3 \\ \mathbf{s}^4 \\ \mathbf{s}^5 \end{bmatrix} = \underbrace{\begin{bmatrix} -35 & 10 \\ -6 & -8 \\ -27 & -17 \\ -6 & -1 \\ -17 & -3 \end{bmatrix}}_{\mathbf{S}_l} \underbrace{\begin{bmatrix} -3 & 1 \\ -1 & 6 \\ 5 & 8 \\ 1 & 1 \\ 2 & 3 \end{bmatrix}}_{\mathbf{S}_r}. \tag{57}$$

It can be seen that the individual key set stored by the workers is $\{-3K_1 + K_2, -K_1 + 6K_2, 5K_1 + 8K_2, K_1 + K_2, 2K_1 + 3K_2\}$.

When the demand matrix changes, for example, it becomes

$$\mathbf{F}' = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 4 & 9 & 2 & 1 \end{bmatrix}, \tag{58}$$

following the same computation scheme as above, we obtain

$$\mathbf{F}_3 = \underbrace{\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 4 & 9 & 2 & 1 \\ 8 & 10 & 8 & 3 & 10 \\ 5 & 9 & 7 & 2 & 8 \end{bmatrix}}_{\mathbf{F}_l} \underbrace{\begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}}_{\mathbf{F}_r}. \tag{59}$$

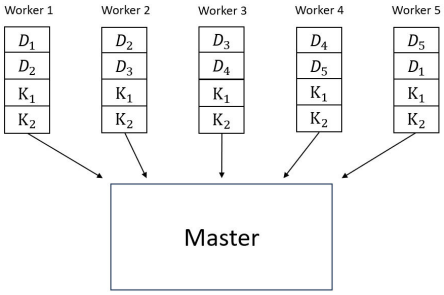


Fig. 4: The individual key size of the baseline scheme is 2.

Next, we compute the matrix $\mathbf{S}_3$*, which is given by:*

$$\mathbf{S}_3 = \begin{bmatrix} \mathbf{s}^1 \\ \mathbf{s}^2 \\ \mathbf{s}^3 \\ \mathbf{s}^4 \\ \mathbf{s}^5 \end{bmatrix} = \underbrace{\begin{bmatrix} 43 & -5 \\ 23 & -9 \\ 44 & -2 \\ 66 & -2 \\ 1 & 0 \end{bmatrix}}_{\mathbf{S}_l} \underbrace{\begin{bmatrix} -47 & 54 \\ -3 & 2 \\ -9 & 6 \\ -13 & 8 \\ -1 & 1 \end{bmatrix}}_{\mathbf{S}_r}. \tag{60}$$

As we can see, when the demand matrix changes, $\mathbf{S}_r$ *also changes. Consequently, the product of* $\mathbf{S}_r$ *and the identity matrix in* $\mathbf{F}_r$ *also changes. In other words, the workers need to send new key linear combinations, and the key storage requirement for each worker increases. It can be seen that the additional individual key set stored by the workers is* $\{-47K_1 + 54K_2, -3K_1 + 2K_2, -9K_1 + 6K_2, -13K_1 + 8K_2, -K_1 + K_2\}$ *i.e., each worker needs to store the complete keys* K_1 *and* K_2 *to handle the different demand matrices, meaning the individual key size is 2, as shown in Fig. 4.*

Notice that in our scheme, when we change the demand matrix, the number of unknowns and resulting equations remain the same, implying that we can still solve for a new set of unknowns. By keeping $\mathbf{S}_r$ *and* $\mathbf{F}_r$ *fixed, the key combinations sent by each worker remain constant. The individual key set stored by workers is still* $\{K_1 + K_2, K_1 + 3K_2, 3K_1 + 2K_2, 2K_1 + 3K_2, 9K_1 + K_2\}$*. In other words, the individual key size is*

$$\mathsf{Q}_\mathrm{s} = \frac{\mathsf{K}_\mathrm{c}}{\mathsf{m} + \mathsf{K}_\mathrm{c} - 1} = \frac{\mathsf{R}^\star}{\mathsf{N}_\mathrm{r}} = 1, \tag{61}$$

satisfying (45).

In Fig. 5, we consider the secure distributed linearly separable computation problem where $\mathsf{N} = \mathsf{K} = 11$, $\mathsf{N}_\mathrm{r} = 7$, and $\mathsf{K}_\mathrm{c} = 2$. It can be seen that our scheme significantly outperforms the baseline scheme in terms of individual key size.

We now generalize the distributed computing scheme described in Example 4. The general case considers $\mathsf{K}_\mathrm{c} = \frac{\mathsf{K}}{\mathsf{N}}\mathsf{u}$, where $\mathsf{u} \in [\mathsf{N}_\mathrm{r} - \mathsf{m} + 1]$ and $\mathsf{u}(\mathsf{N}_\mathrm{r} - \mathsf{m} - \mathsf{u} + 1) + \frac{\mathsf{m}+\mathsf{u}-1}{\mathsf{u}}$. During the data assignment phase, we adopt the cyclic assignment strategy, where each worker is assigned $\mathsf{M} = \frac{\mathsf{K}}{\mathsf{N}}(\mathsf{N} - \mathsf{N}_\mathrm{r} + \mathsf{m})$ datasets.

Computing phase: To achieve the optimal communication cost $R^* = \mathsf{N}_\mathrm{r}\frac{\mathsf{K}_\mathrm{c}}{\mathsf{m}+\mathsf{u}-1}$, we first divide each message $W_k, k \in [\mathsf{K}]$, into $\mathsf{m}+\mathsf{u}-1$ non-overlapping and equal-length

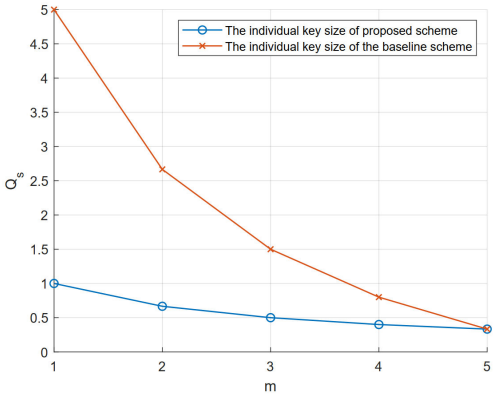


Fig. 5: Numerical evaluations for the individual key size of the $(\mathsf{K}, \mathsf{N}, \mathsf{N}_\mathrm{r}, \mathsf{K}_\mathrm{c}, \mathsf{m}) = (11, 11, 7, 2, \mathsf{m})$ secure distributed linearly separable computation problem.

sub-messages, denoted as $W_k = \{W_{k,j} : j \in [\mathsf{m} + \mathsf{K}_\mathrm{c} - 1]\}$, so that each sub-message contains $\frac{\mathsf{L}}{\mathsf{m}+\mathsf{u}-1}$ symbols in $\mathbb{F}_\mathsf{q}$. Additionally, to satisfy the security constraint in (1), we generate $\mathsf{Q} = \mathsf{N}_\mathrm{r}\mathsf{K}_\mathrm{c} - \mathsf{K}_\mathrm{c}(\mathsf{m} + \mathsf{u} - 1)$ independent keys, each of size $\frac{\mathsf{L}}{\mathsf{m}+\mathsf{u}-1}$. Each worker is tasked to send K_c linear combinations of sub-messages. From the answers of N_r workers, the master receives a total of $\mathsf{N}_\mathrm{r}\mathsf{K}_\mathrm{c}$ linear combinations of sub-messages, which contain the desired $(\mathsf{m}+\mathsf{u}-1)\mathsf{K}_\mathrm{c}$ linear combinations. Consequently, we generate

$$\mathsf{v} = \mathsf{N}_\mathrm{r}\mathsf{K}_\mathrm{c} - (\mathsf{m} + \mathsf{u} - 1)\mathsf{K}_\mathrm{c} = \mathsf{K}_\mathrm{c}(\mathsf{N}_\mathrm{r} - \mathsf{m} - \mathsf{u} + 1)$$

virtually requested linear combinations; Thus, the effective demand matrix $\mathbf{F}$ has dimension $\mathsf{N}_\mathrm{r}\mathsf{K}_\mathrm{c} \times \mathsf{K}(\mathsf{m}+\mathsf{u}-1)$, with its form shown at the top of the next page in (62), where $\mathbf{F}_{\mathsf{m}+\mathsf{u}}$ corresponds to the key portion. For convenience, we set the part of the virtual demand corresponding to the keys as an identity matrix.

Our coding strategy is to let each worker $n \in [\mathsf{N}]$ send K_c linear combinations of sub-messages

$$\mathbf{s}^{n,i}\mathbf{F}[W_{1,1}; \ldots; W_{\mathsf{K},1}; W_{1,2}; \ldots; W_{\mathsf{K},\mathsf{m}+\mathsf{u}-1}], \tag{63}$$

where $i \in [\mathsf{K}_\mathrm{c}]$, for a total of $\frac{\mathsf{K}_\mathrm{c}\mathsf{L}}{\mathsf{m}+\mathsf{u}-1}$ symbols, coinciding with the converse bound. Note that $\mathbf{s}^{n,i}$ is vector with length $\mathsf{N}_\mathrm{r}\mathsf{K}_\mathrm{c}$. The overall coding matrix $\mathbf{S}$ is further expanded, as shown at the top of the next page.

For each $n \in [\mathsf{N}]$, worker n cannot compute the $\mathrm{Mod}(n + l, \mathsf{K})$ datasets, where $l \in [\mathsf{M} : \mathsf{K} - 1]$. This constraint can be expressed as:

$$\mathbf{s}^{n,i}\mathbf{c}_{\mathrm{Mod}(n+l,\mathsf{K})+(j-1)\mathsf{K}}W_{\mathrm{Mod}(n+l,\mathsf{K}),j} = 0, \tag{65}$$

where $i \in [\mathsf{K}_\mathrm{c}]$ denotes the i^{th} linear combination sent by worker n, and $\mathbf{c}_k$ is the k^{th} column of matrix $\mathbf{F}$.

Additionally, for any subset $\mathcal{A} \subseteq [\mathsf{N}]$ with $|\mathcal{A}| = \mathsf{N}_\mathrm{r}$, the master must recover the desired linear combinations from the

$$\mathbf{F}=\Bigg[\underbrace{\begin{bmatrix} f_{1,1} & \cdots & f_{1,\mathsf{K}} \\ \vdots & \ddots & \vdots \\ f_{\mathsf{K_c},1} & \cdots & f_{\mathsf{K_c},\mathsf{K}} \\ 0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 0 \\ a_{1,1} & \cdots & a_{1,\mathsf{K}} \\ \vdots & \ddots & \vdots \\ a_{\mathsf{v},1} & \cdots & a_{\mathsf{v},\mathsf{K}} \end{bmatrix}}_{\mathbf{F}_1} \underbrace{\begin{bmatrix} 0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 0 \\ f_{1,1} & \cdots & f_{1,\mathsf{K}} \\ \vdots & \ddots & \vdots \\ f_{\mathsf{K_c},1} & \cdots & f_{\mathsf{K_c},\mathsf{K}} \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 0 \\ a_{1,\mathsf{K}+1} & \cdots & a_{1,2\mathsf{K}} \\ \vdots & \ddots & \vdots \\ a_{\mathsf{v},\mathsf{K}+1} & \cdots & a_{\mathsf{v},2\mathsf{K}} \end{bmatrix}}_{\mathbf{F}_2} \cdots \underbrace{\begin{bmatrix} 0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 0 \\ f_{1,1} & \cdots & f_{1,\mathsf{K}} \\ \vdots & \ddots & \vdots \\ f_{\mathsf{K_c},1} & \cdots & f_{\mathsf{K_c},\mathsf{K}} \\ a_{1,(\mathsf{m}+\mathsf{u}-2)\mathsf{K}+1} & \cdots & a_{1,(\mathsf{m}+\mathsf{u}-1)\mathsf{K}} \\ \vdots & \ddots & \vdots \\ a_{\mathsf{v},(\mathsf{m}+\mathsf{u}-2)\mathsf{K}+1} & \cdots & a_{\mathsf{v},(\mathsf{m}+\mathsf{u}-1)\mathsf{K}} \end{bmatrix}}_{\mathbf{F}_{\mathsf{m}+\mathsf{u}-1}} \underbrace{\begin{bmatrix} 0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 0 \\ 0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 0 \\ 1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 1 \end{bmatrix}}_{\mathbf{F}_{\mathsf{m}+\mathsf{u}}}\Bigg]. \tag{62}$$

$$\mathbf{S}=\begin{bmatrix} \mathbf{s}^{1,1} \\ \vdots \\ \mathbf{s}^{1,\mathsf{K_c}} \\ \mathbf{s}^{2,1} \\ \vdots \\ \mathbf{s}^{\mathsf{N},\mathsf{K_c}} \end{bmatrix}=\Bigg[\underbrace{\begin{bmatrix} s_1^{1,1} & \cdots & s_{\mathsf{K_c}}^{1,1} \\ \vdots & \ddots & \vdots \\ s_1^{1,\mathsf{K_c}} & \cdots & s_{\mathsf{K_c}}^{1,\mathsf{K_c}} \\ s_1^{2,1} & \cdots & s_{\mathsf{K_c}}^{2,1} \\ \vdots & \ddots & \vdots \\ s_1^{\mathsf{N},\mathsf{K_c}} & \cdots & s_{\mathsf{K_c}}^{\mathsf{N},\mathsf{K_c}} \end{bmatrix}}_{\mathbf{S}_1} \cdots \underbrace{\begin{bmatrix} s_{(\mathsf{m}+\mathsf{u}-2)\mathsf{K_c}+1}^{1,1} & \cdots & s_{(\mathsf{m}+\mathsf{u}-1)\mathsf{K_c}}^{1,1} \\ \vdots & \ddots & \vdots \\ s_{(\mathsf{m}+\mathsf{u}-2)\mathsf{K_c}+1}^{1,\mathsf{K_c}} & \cdots & s_{(\mathsf{m}+\mathsf{u}-1)\mathsf{K_c}}^{1,\mathsf{K_c}} \\ s_{(\mathsf{m}+\mathsf{u}-2)\mathsf{K_c}+1}^{2,1} & \cdots & s_{(\mathsf{m}+\mathsf{u}-1)\mathsf{K_c}}^{2,1} \\ \vdots & \ddots & \vdots \\ s_{(\mathsf{m}+\mathsf{u}-2)\mathsf{K_c}+1}^{\mathsf{N},\mathsf{K_c}} & \cdots & s_{(\mathsf{m}+\mathsf{u}-1)\mathsf{K_c}}^{\mathsf{N},\mathsf{K_c}} \end{bmatrix}}_{\mathbf{S}_{\mathsf{m}+\mathsf{u}-1}} \underbrace{\begin{bmatrix} b_1^{1,1} & \cdots & b_{\mathsf{v}}^{1,1} \\ \vdots & \ddots & \vdots \\ b_1^{1,\mathsf{K_c}} & \cdots & b_{\mathsf{v}}^{1,\mathsf{K_c}} \\ b_1^{2,1} & \cdots & b_{\mathsf{v}}^{2,1} \\ \vdots & \ddots & \vdots \\ b_1^{\mathsf{N},\mathsf{K_c}} & \cdots & b_{\mathsf{v}}^{\mathsf{N},\mathsf{K_c}} \end{bmatrix}}_{\mathbf{S}_{\mathsf{m}+\mathsf{u}}}\Bigg]. \tag{64}$$

---

responses of workers in $\mathcal{A}$. This requires:

$$[\mathbf{s}^{\mathcal{A}(1),1};\ldots;\mathbf{s}^{\mathcal{A}(1),\mathsf{K_c}};\mathbf{s}^{\mathcal{A}(2),1};\ldots;\mathbf{s}^{\mathcal{A}(\mathsf{N_r}),\mathsf{K_c}}] \text{ to be full rank, } \forall\mathcal{A}\subseteq[\mathsf{N}]:|\mathcal{A}|=\mathsf{N_r}. \tag{66}$$

Our goal is to determine the variables in $\mathbf{S}$ (i.e., $s_i^{n,j}$ where $n\in[\mathsf{N}],j\in[\mathsf{K_c}],i\in[(\mathsf{m}+\mathsf{u}-1)\mathsf{K_c}]$, and $b_i^{n,j}$ where $n\in[\mathsf{N}],j\in[\mathsf{K_c}],i\in[\mathsf{v}]$), and in $\mathbf{F}$ (i.e., $a_{i,k}$ where $i\in[\mathsf{v}],k\in[(\mathsf{m}+\mathsf{u}-1)\mathsf{K}]$), such that the constraints in (65) and (66) are satisfied.

We divide the matrix $\mathbf{F}$ into $\mathsf{m}+\mathsf{u}-1$ sub-matrices, $\mathbf{F}_1,\ldots,\mathbf{F}_{\mathsf{m}+\mathsf{u}-1}$, each with dimensions $\mathsf{N_r}\mathsf{K_c}\times\mathsf{K}$, as shown in (62). Similarly, matrix $\mathbf{S}$ is divided into $\mathsf{m}+\mathsf{u}$ sub-matrices: $\mathbf{S}_1,\ldots,\mathbf{S}_{\mathsf{m}+\mathsf{u}-1}$, each with dimensions $\mathsf{N}\mathsf{K_c}\times\mathsf{K_c}$, and $\mathbf{S}_{\mathsf{m}+\mathsf{u}}$ with dimensions $\mathsf{N}\mathsf{K_c}\times\mathsf{v}$, as illustrated in (64).

As in Example 4, the proposed computing scheme consists of three main steps:

1) Choose values for the variables in $\mathbf{S}_{\mathsf{m}+\mathsf{u}}$, which are associated with the coefficients of the key linear combinations sent by the workers.
2) Use the constraints in (65), which become linear after fixing $\mathbf{S}_{\mathsf{m}+\mathsf{u}}$, to solve for the remaining variables;
3) Verify that the constraints in (66) are satisfied, ensuring the scheme is decodable.

Step 1: We assign values to $\mathbf{S}_{\mathsf{m}+\mathsf{u}}$ in the following form:

$$\mathbf{S}_{\mathsf{m}+\mathsf{u}}=\begin{bmatrix} b_1^{1,1} & \cdots & b_{\frac{\mathsf{v}}{\mathsf{K_c}}}^{1,1} & \cdots & b_{\frac{(\mathsf{K_c}-1)\mathsf{v}}{\mathsf{K_c}}+1}^{1,1} & \cdots & b_{\mathsf{v}}^{1,1} \\ b_1^{1,2} & \cdots & b_{\frac{\mathsf{v}}{\mathsf{K_c}}}^{1,2} & \cdots & b_{\frac{(\mathsf{K_c}-1)\mathsf{v}}{\mathsf{K_c}}+1}^{1,2} & \cdots & b_{\mathsf{v}}^{1,2} \\ \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ b_1^{1,\mathsf{K_c}} & \cdots & b_{\frac{\mathsf{v}}{\mathsf{K_c}}}^{1,\mathsf{K_c}} & \cdots & b_{\frac{(\mathsf{K_c}-1)\mathsf{v}}{\mathsf{K_c}}+1}^{1,\mathsf{K_c}} & \cdots & b_{\mathsf{v}}^{1,\mathsf{K_c}} \\ b_1^{2,1} & \cdots & b_{\frac{\mathsf{v}}{\mathsf{K_c}}}^{2,1} & \cdots & b_{\frac{(\mathsf{K_c}-1)\mathsf{v}}{\mathsf{K_c}}+1}^{2,1} & \cdots & b_{\mathsf{v}}^{2,1} \\ \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ b_1^{\mathsf{N},\mathsf{K_c}} & \cdots & b_{\frac{\mathsf{v}}{\mathsf{K_c}}}^{\mathsf{N},\mathsf{K_c}} & \cdots & b_{\frac{(\mathsf{K_c}-1)\mathsf{v}}{\mathsf{K_c}}+1}^{\mathsf{N},\mathsf{K_c}} & \cdots & b_{\mathsf{v}}^{\mathsf{N},\mathsf{K_c}} \end{bmatrix}$$

$$=\begin{bmatrix} * & \cdots & * & 0 & \cdots & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 \\ 0 & \cdots & 0 & * & \cdots & * & 0 & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 & * & \cdots & * \\ * & \cdots & * & 0 & \cdots & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 & * & \cdots & * \end{bmatrix}, \tag{67}$$

where each '$*$' represents a uniformly i.i.d. symbol in $\mathbb{F}_{\mathsf{q}}$.

This structure ensures that the constraints in (65) are linearly independent with respect to the remaining variables, which will be determined in the next step.

Step 2: We now determine the variables in matrices $\mathbf{F}$ and $\mathbf{S}$. Fixing one $i \in [\mathsf{K_c}]$ and one $j \in [\mathsf{m}+\mathsf{u}-1]$, the constraints in (65) become:

$$\begin{aligned} 0 = &\sum_{i_1 \in [\mathsf{K_c}]} f_{i_1,\mathrm{Mod}(n+l,\mathsf{K})}\ s^{n,i}_{(j-1)\mathsf{K_c}+i_1} \\ &+ \sum_{i_2 \in [\mathsf{v}]} b^{n,i}_{i_2}\ a_{i_2,(j-1)\mathsf{K}+\mathrm{Mod}(n+l,\mathsf{K})}. \end{aligned} \tag{68}$$

In (68), the coefficients $f_{i_1,k}$ are elements of the demand matrix $\mathbf{F}$, and the values $b^{n,i}_{i_2}$ were determined in Step 1. The total number of variables in (68) is:

$$\begin{aligned} \mathsf{N}\mathsf{K_c} + \frac{\mathsf{v}}{\mathsf{K_c}}\mathsf{K} &= \mathsf{N}\frac{\mathsf{K}}{\mathsf{N}}\mathsf{u} + (\mathsf{N_r}-\mathsf{m}-\mathsf{u}+1)\mathsf{K} \\ &= \mathsf{K}(\mathsf{N_r}-\mathsf{m}+1). \end{aligned} \tag{69}$$

There are a total of $\mathsf{K}(\mathsf{N_r}-\mathsf{m})$ constraints in (68). Since the number of variables exceeds the number of constraints, we solve the resulting system of equations, which is likely to have a solution. For each $n \in [\mathsf{N}]$, we can assign:

$$s^{n,i}_{(j-1)\mathsf{K_c}+(p-1)\mathsf{u}+\mathrm{Mod}(n,\mathsf{u})}, \quad \forall p \in [\mathsf{K}/\mathsf{N}], \tag{70}$$

random values uniformly i.i.d. over $\mathbb{F}_{\mathsf{q}}$. This determines K variables out of the $\mathsf{K}(\mathsf{N_r}-\mathsf{m}+1)$, leaving $\mathsf{K}(\mathsf{N_r}-\mathsf{m})$ variables to be solved by $\mathsf{K}(\mathsf{N_r}-\mathsf{m})$ linear equations. As shown in Appendix B, these equations are linearly independent with high probability, allowing us to solve for all variables in (69).

By iterating over all pairs (i,j) where $i \in [\mathsf{K_c}]$ and $j \in [\mathsf{m}+\mathsf{u}-1]$, we determine all elements in $\mathbf{S}$ and $\mathbf{F}$.

Step 3: As shown in Appendix B, the constraints in (66) hold with high probability. Consequently, each worker $n \in [\mathsf{N}]$ computes and transmits $\mathsf{K_c}$ linear combinations of sub-messages:

$$\begin{aligned} &\mathbf{s}^{n,i}\mathbf{F}[W_{1,1};\ldots;W_{\mathsf{K},1};W_{1,2};\ldots;W_{\mathsf{K},\mathsf{m}+\mathsf{i}-1}] \\ &= \mathbf{s}^{n,i}[F_1;\ldots;F_{\mathsf{K_c}}], \end{aligned} \tag{71}$$

where $i \in [\mathsf{K_c}]$.

Decoding Phase: Let $\mathcal{A} \subseteq [\mathsf{N}]$ be the set of responding workers with $|\mathcal{A}| = \mathsf{N_r}$. The master receives a total of $\mathsf{N_r}\mathsf{K_c}$ messages:

$$\begin{aligned} \mathbf{X}_{\mathcal{A}} =&[\mathbf{s}^{\mathcal{A}(1),1};\ldots;\mathbf{s}^{\mathcal{A}(1),\mathsf{K_c}};\mathbf{s}^{\mathcal{A}(2),1};\ldots;\mathbf{s}^{\mathcal{A}(\mathsf{N_r}),\mathsf{K_c}}] \\ &\times [F_1;\ldots;F_{\mathsf{K_c}}]. \end{aligned} \tag{72}$$

Since the matrix $[\mathbf{s}^{\mathcal{A}(1),1};\ldots;\mathbf{s}^{\mathcal{A}(\mathsf{N_r}),\mathsf{K_c}}]$ is full-rank, the master computes:

$$[\mathbf{s}^{\mathcal{A}(1),1};\ldots;\mathbf{s}^{\mathcal{A}(\mathsf{N_r}),\mathsf{K_c}}]^{-1}\mathbf{X}_{\mathcal{A}}, \tag{73}$$

recovering $[F_1;\ldots;F_{\mathsf{K_c}}]$, which contains the required linear combinations. It is worth noting that the above process remains unchanged regardless of the demand. Thus, each worker can consistently send the invariant linear combinations of keys, as the coefficients of the key linear combinations have already been fixed in Step 1. This implies that each worker only needs to store $\mathsf{K_c}$ key combinations, with a total size of

$$\mathsf{Q_s} = \frac{\mathsf{K_c}}{\mathsf{m}+\mathsf{u}-1} = \frac{\mathsf{R}^\star}{\mathsf{N_r}} \tag{74}$$

satisfying (45).

VI. Conclusion

We focus on the secure distributed linearly separable computation problem, which prevents the master from accessing any information about the dataset besides the task function. We first research the minimum key size under general computation cost while achieving optimal communication cost. For this purpose, we propose a new computing scheme with a novel assignment strategy. The computing scheme can cover the optimality results of the computational scheme with fractional repetition assignment. The new computing scheme is also applicable to the case of minimal computational cost in [17]. Then we propose a computing scheme that achieves the minimum individual key size under the optimal communication cost when the demand is unknown.

Appendix A
Proof of Theorem 1

By the security constraint in (1), the master can only obtain $W_1+\cdots+W_{\mathsf{K}}$ without accessing any additional information about the messages, even after receiving answers from all workers. Let $X_{\mathcal{S}} = \{X_n : n \in \mathcal{S}\}$. From (1), we derive:

$$\begin{aligned} 0 &= I(W_1,\ldots,W_{\mathsf{K}};X_{[\mathsf{N}]}|W_1+W_2+\cdots+W_{\mathsf{K}}) && \text{(75a)}\\ &= H(X_{[\mathsf{N}]}|W_1+W_2+\cdots+W_{\mathsf{K}}) - H(X_{[\mathsf{N}]}|W_1,\ldots,W_{\mathsf{K}}) && \text{(75b)}\\ &\geq H(X_{[\mathsf{N}]}|W_1+W_2+\cdots+W_{\mathsf{K}}) \\ &\quad - H(Q,W_1,\ldots,W_{\mathsf{K}}|W_1,\ldots,W_{\mathsf{K}}) && \text{(75c)}\\ &= H(X_{[\mathsf{N}]}|W_1+W_2+\cdots+W_{\mathsf{K}}) - H(Q) && \text{(75d)}\\ &= H(X_{[\mathsf{N}]}) - I(X_{[\mathsf{N}]};W_1+W_2+\cdots+W_{\mathsf{K}}) - H(Q) && \text{(75e)}\\ &\geq H(X_{[\mathsf{N}]}) - H(W_1+W_2+\cdots+W_{\mathsf{K}}) - H(Q), && \text{(75f)} \end{aligned}$$

where (75c) follows from the fact that $X_{[\mathsf{N}]}$ is a function of Q and $W_1,\ldots,W_{\mathsf{K}}$, and (75d) relies on Q being independent of $W_1,\ldots,W_{\mathsf{K}}$.

From (75f), we further obtain:

$$\begin{aligned} \eta\mathsf{L} \geq H(Q) &\geq H(X_{[\mathsf{N}]}) - H(W_1+\cdots+W_{\mathsf{K}}) && \text{(76a)}\\ &\geq H(X_{[\mathsf{N}]}) - \mathsf{L} && \text{(76b)}\\ &\geq H(X_{s_1},\ldots,X_{s_{|\mathbf{s}|}}) - \mathsf{L} && \text{(76c)}\\ &\geq \frac{|\mathbf{s}|\mathsf{L}}{\mathsf{m}} - \mathsf{L}, && \text{(76d)} \end{aligned}$$

where (76b) follows from the independence of K messages, each uniformly i.i.d. over $[\mathbb{F}_{\mathsf{q}}]^{\mathsf{L}}$, and (76d) is derived using the chain rule of entropy.

Note that worker s_i always contains at least one message W_{s_i} which can not be computed by m workers from $\{s_1,\ldots,s_{i-1}\}$. Consider the case that apart from $\mathsf{N}-\mathsf{N_r}$

stragglers, there are exactly m workers who can compute W_{s_i}. Then, the entropy of messages from these workers should be larger than L.

APPENDIX B
FEASIBILITY PROOF OF THE PROPOSED COMPUTING SCHEME IN SECTION V

First, we consider the $(\mathsf{K},\mathsf{N},\mathsf{N_r},\mathsf{K_c},\mathsf{m}) = (\mathsf{N},\mathsf{N},\mathsf{N_r},\mathsf{u},\mathsf{m})$ distributed linearly separable computation problem, where $\mathsf{N} \geq \mathsf{u}(\mathsf{N_r}-\mathsf{m}-\mathsf{u}+1)+\frac{\mathsf{m}+\mathsf{u}-1}{\mathsf{u}}$. We prove that the system of linear equations given in (68) is solvable with high probability, and the constraints in (66) are also satisfied with high probability. Next, we prove that if the proposed scheme works with high probability for $(\mathsf{K},\mathsf{N},\mathsf{N_r},\mathsf{K_c},\mathsf{m}) = (\mathsf{N},\mathsf{N},\mathsf{N_r},\mathsf{u},\mathsf{m})$, then it also works with high probability for $(\mathsf{K},\mathsf{N},\mathsf{N_r},\mathsf{K_c},\mathsf{m}) = (\mathsf{aN},\mathsf{N},\mathsf{N_r},\mathsf{au},\mathsf{m})$ distributed linearly separable computation problem, where $\mathsf{a} = \frac{\mathsf{K}}{\mathsf{N}}$ is a positive integer.

A. $\mathsf{N} = \mathsf{K}$

We use the Schwartz-Zippel Lemma [21]–[23] to prove the feasibility of the proposed computation scheme. This method has already been applied in [12, Appendix C] to demonstrate the feasibility of the computation scheme for the case where $\mathsf{m} = 1$.

Recall that for each $i \in [\mathsf{K_c}]$ and $j \in [\mathsf{m}+\mathsf{u}-1]$, in Step 2 of the proposed scheme, we obtain $\mathsf{K}(\mathsf{N_r}-\mathsf{m})$ linear equations as shown in (68). After determining the values of the K elements in (70), the system of linear equations still has $\mathsf{K}(\mathsf{N_r}-\mathsf{m})$ unknowns to be determined(the vector representing these $\mathsf{K}(\mathsf{N_r}-\mathsf{m})$ unknowns is denoted as $\mathbf{x}$). Therefore, we can represent the system of linear equations in (68) as the following matrix form:

$$(\mathbf{A})_{\mathsf{K}(\mathsf{N_r}-\mathsf{m})\times\mathsf{K}(\mathsf{N_r}-\mathsf{m})}\ (\mathbf{x})_{\mathsf{K}(\mathsf{N_r}-\mathsf{m})\times 1} = (\mathbf{b})_{\mathsf{K}(\mathsf{N_r}-\mathsf{m})\times 1}, \quad (77)$$

where the coefficients in $\mathbf{A}$ are composed of the elements in $\mathbf{F}$ and $\mathbf{S}_{\mathsf{m}+\mathsf{u}}$, and the elements in $\mathbf{b}$ are formed from the variables in (70) and the elements in $\mathbf{F}$, all of which are generated uniformly i.i.d. over $\mathbb{F}_{\mathsf{q}}$. Thus, the determinant of $\mathbf{A}$ can be viewed as a multivariate polynomial in which the variables are the elements of $\mathbf{F}$ and $\mathbf{S}_{\mathsf{m}+\mathsf{u}}$. If this multivariate polynomial is a non-zero polynomial (i.e., a multivariate polynomial whose coefficients are not all zero), then we can apply the Schwartz-Zippel Lemma [21]–[23]. Furthermore, since each variable of this multivariate polynomial is uniformly i.i.d. over $\mathbb{F}_{\mathsf{q}}$, after applying the Schwartz-Zippel Lemma [21]–[23], it is easy to see that the probability of this multivariate polynomial being zero tends to zero. In other words, the determinant of $\mathbf{A}$ is non-zero with high probability, which implies that the linear system given in (68) is solvable with high probability. Therefore, we will next prove that this multivariate polynomial is non-zero. To do so, it suffices to find a set of values for the elements of $\mathbf{F}$ and $\mathbf{S}_{\mathsf{m}+\mathsf{u}}$ such that $\det(\mathbf{A})$ is non-zero, which would show that the multivariate polynomial is non-zero. Under the constraint $\mathsf{N} \geq \frac{\mathsf{m}+\mathsf{u}-1}{\mathsf{u}}+\mathsf{u}(\mathsf{N_r}-\mathsf{m}-\mathsf{u}+1)$, we tested all cases where $\mathsf{N} \leq 40$. By randomly assigning values to the elements of $\mathbf{F}$ and $\mathbf{S}_{\mathsf{m}+\mathsf{u}}$, we found that there always exists a set of values that makes $\det(\mathbf{A})$ non-zero. This implies that for each $i \in [\mathsf{K_c}]$ and $j \in [\mathsf{m}+\mathsf{u}-1]$, the system of linear equations given in (68) is solvable with probability close to 1. Using the probability union bound, we conclude that the proposed computation scheme is also solvable with probability close to 1 for all $i \in [\mathsf{K_c}]$ and $j \in [\mathsf{m}+\mathsf{u}-1]$.

Next, we will prove that the constraints in (66) are satisfied with high probability. For each $n \in [\mathsf{K}(\mathsf{N_r}-\mathsf{m})\times 1]$, by Cramer's rule, the n-th element of $\mathbf{x}$ is given by

$$x_n = \frac{\det(\mathbf{A}_n)}{\det(\mathbf{A})}, \quad (78)$$

where $\mathbf{A}_n$ is the matrix formed by replacing the n-th column of $\mathbf{A}$ with $\mathbf{b}$. The determinant in the denominator has been proven to be non-zero with high probability as shown earlier. Note that each of the elements to be determined in $\mathbf{S}$ can be expressed as the ratio of two such polynomials, where the variables in the polynomials are the elements in $\mathbf{F}$, $\mathbf{S}_{\mathsf{m}+\mathsf{u}}$, and the variables in (70). For each $\mathcal{A} \subseteq [\mathsf{N}]$ with $|\mathcal{A}| = \mathsf{N_r}$, the determinant of $\mathbf{S}_{\mathcal{A}}$ (i.e., the matrix formed by stacking the vectors $[\mathbf{s}^{\mathcal{A}(1),1};\ldots;\mathbf{s}^{\mathcal{A}(1),\mathsf{K_c}};\mathbf{s}^{\mathcal{A}(2),1};\ldots;\mathbf{s}^{\mathcal{A}(\mathsf{N_r}),\mathsf{K_c}}]$) can be written as

$$\det(\mathbf{S}_{\mathcal{A}}) = \sum_{k\in[(\mathsf{N_r}\mathsf{u})!]} \frac{P_k}{T_k}, \quad (79)$$

where P_k and T_k are polynomials in the elements of $\mathbf{F}$, $\mathbf{S}_{\mathsf{m}+\mathsf{u}}$, and the variables in (70), for all i and j. We then define

$$P_{\mathcal{A}} = \det(\mathbf{S}_{\mathcal{A}}) \prod_{k\in[(\mathsf{N_r}\mathsf{u})!]} T_k. \quad (80)$$

If $P_{\mathcal{A}} \neq 0$, then $\det(\mathbf{S}_{\mathcal{A}}) \neq 0$, and thus $\mathbf{S}_{\mathcal{A}}$ is full rank. Since $P_{\mathcal{A}}$ is a polynomial in the elements of $\mathbf{F}$, $\mathbf{S}_{\mathsf{m}+\mathsf{u}}$, and the variables in (70), all of which are generated uniformly i.i.d. over $\mathbb{F}_{\mathsf{q}}$, if $P_{\mathcal{A}}$ is a non-zero polynomial, we can apply the Schwartz-Zippel Lemma [21]–[23]. Consequently, the probability of this multivariate polynomial being zero tends to zero, implying that $\mathbf{S}_{\mathcal{A}}$ is full rank with high probability. We now aim to prove that $P_{\mathcal{A}}$ is non-zero. We tested all cases where $\mathsf{N} \leq 40$ under the constraint $\mathsf{N} \geq \mathsf{u}(\mathsf{N_r}-\mathsf{m}-\mathsf{u}+1)+\frac{\mathsf{m}+\mathsf{u}-1}{\mathsf{u}}$. By randomly assigning values to the elements of $\mathbf{F}$ and $\mathbf{S}_{\mathsf{m}+\mathsf{u}}$, we observed that there always exists a set of values for which $P_{\mathcal{A}}$ is non-zero.

In conclusion, we prove that the proposed computing scheme is feasible with high probability for the case where $\mathsf{u}(\mathsf{N_r}-\mathsf{m}-\mathsf{u}+1)+\frac{\mathsf{m}+\mathsf{u}-1}{\mathsf{u}} \leq \mathsf{K} = \mathsf{N} \leq 40$.

B. N *divides* K

We then focus the $(\mathsf{K},\mathsf{N},\mathsf{N_r},\mathsf{K_c},\mathsf{m}) = (\mathsf{aN},\mathsf{N},\mathsf{N_r},\mathsf{au},\mathsf{m})$ distributed linearly separable computation problem, where $\mathsf{a} = \frac{\mathsf{K}}{\mathsf{N}}$ is a positive integer. We construct the demand matrix $\mathbf{F}$ (recall that $(\mathbf{M})_{m\times n}$ indicates that the dimension of matrix $\mathbf{M}$ is $m\times n$;$(\mathbf{0})_{m\times n}$ indicates that the zero matrix with dimension

$m \times n$) as follows,

$$\mathbf{F} = \begin{bmatrix} (\mathbf{F}_1)_{\mathsf{u}\times\mathsf{N}} & \mathbf{0}_{\mathsf{u}\times\mathsf{N}} & \cdots & \mathbf{0}_{\mathsf{u}\times\mathsf{N}} \\ \mathbf{0}_{\mathsf{u}\times\mathsf{N}} & (\mathbf{F}_2)_{\mathsf{u}\times\mathsf{N}} & \cdots & \mathbf{0}_{\mathsf{u}\times\mathsf{N}} \\ \vdots & \vdots & \vdots & \vdots \\ \mathbf{0}_{\mathsf{u}\times\mathsf{N}} & \mathbf{0}_{\mathsf{u}\times\mathsf{N}} & \cdots & (\mathbf{F}_{\mathsf{a}})_{\mathsf{u}\times\mathsf{N}} \end{bmatrix}, \tag{81}$$

where each element in $\mathbf{F}_i, i \in [\mathsf{a}]$, is uniformly i.i.d. over $\mathbb{F}_{\mathsf{q}}$. In the above construction, the $\left(\mathsf{K}, \mathsf{N}, \mathsf{N}_{\mathrm{r}}, \frac{\mathsf{K}}{\mathsf{N}}\mathsf{u}, \mathsf{m}\right) = (\mathsf{aN}, \mathsf{N}, \mathsf{N}_{\mathrm{r}}, \mathsf{au}, \mathsf{m})$ distributed linearly separable computation problem is divided into a independent $\left(\mathsf{K}, \mathsf{N}, \mathsf{N}_{\mathrm{r}}, \frac{\mathsf{K}}{\mathsf{N}}\mathsf{u}, \mathsf{m}\right) = (\mathsf{N}, \mathsf{N}, \mathsf{N}_{\mathrm{r}}, \mathsf{u}, \mathsf{m})$ distributed linearly separable computation sub-problems. In each sub-problem, assuming that the coding matrix of the workers in $\mathcal{A}$ is $\mathbf{S}'_{\mathcal{A}}$, as proven in Appendix B-A, we can obtain that $\det(\mathbf{S}'_{\mathcal{A}}) \neq 0$ with high probability. Hence, it can be seen that in the distributed linearly separable computation problem $\left(\mathsf{K}, \mathsf{N}, \mathsf{N}_{\mathrm{r}}, \frac{\mathsf{K}}{\mathsf{N}}\mathsf{u}, \mathsf{m}\right) = (\mathsf{aN}, \mathsf{N}, \mathsf{N}_{\mathrm{r}}, \mathsf{au}, \mathsf{m})$, $\det(\mathbf{S}_{\mathcal{A}})$ is also non-zero with high probability.

Appendix C
The proof of the applicability of the proposed computing scheme in Section V

We have already proven in Appendix B that if the proposed scheme works for the $(\mathsf{N}, \mathsf{N}, \mathsf{N}_{\mathrm{r}}, \mathsf{u}, \mathsf{m})$ distributed linearly separable computation problem, it also works with high probability for the $(\mathsf{K}, \mathsf{N}, \mathsf{N}_{\mathrm{r}}, \mathsf{K}_{\mathrm{c}}, \mathsf{m})$ distributed linearly separable computation problem, where $\frac{\mathsf{K}}{\mathsf{N}}$ is a positive integer. Therefore, we now consider the case where $\mathsf{K} = \mathsf{N}$.

Returning to the beginning of Step 2 in our proposed scheme, we fix one $j \in [\mathsf{m} + \mathsf{u} - 1]$. The constraint in (65) becomes:

$$\begin{aligned} 0 = &\sum_{i_1 \in [\mathsf{K}_{\mathrm{c}}]} f_{i_1, \mathrm{Mod}(n+l, \mathsf{K})} \; s^{n,i}_{(j-1)\mathsf{K}_{\mathrm{c}}+i_1} \\ &+ \sum_{i_2 \in [\mathsf{v}]} b^{n,i}_{i_2} \; a_{i_2, (j-1)\mathsf{K}+\mathrm{Mod}(n+l, \mathsf{K})}, \end{aligned} \tag{82}$$

$\forall i \in [\mathsf{K}_{\mathrm{c}}], n \in [\mathsf{N}], l \in [\mathsf{M} : \mathsf{K} - 1]$. Notice that for each $j \in [\mathsf{m} + \mathsf{u} - 1]$, we obtain a system of linear equations to solve. In fact, these $(\mathsf{m} + \mathsf{u} - 1)$ problems are all equivalent. Therefore, the system of equations obtained from (82) should have at least $(\mathsf{m}+\mathsf{u}-1)$ linearly independent solutions. There are a total of $\mathsf{uK}(\mathsf{N}_{\mathrm{r}} - \mathsf{m})$ constraints and $\mathsf{uK}(\mathsf{N}_{\mathrm{r}} - \mathsf{m} + 1)$ variables in (82). Therefore, this system of linear equations has at least $\mathsf{uK}(\mathsf{N}_{\mathrm{r}} - \mathsf{m} + 1) - \mathsf{uK}(\mathsf{N}_{\mathrm{r}} - \mathsf{m}) = \mathsf{uK}$ linearly independent solutions.

At the same time, we observe that among these linearly independent solutions, some solutions with specific structures do not satisfy (66), and therefore, we must discard these solutions. These solutions have the following structure: For each $k \in [\mathsf{u}]$ and $p \in [\mathsf{v}]$, we design that

$$(a_{p,(j-1)\mathsf{K}+1}, a_{p,(j-1)\mathsf{K}+2}, \ldots, a_{p,j\mathsf{K}}) = \mathbf{f}_k,$$

where $\mathbf{f}_k$ represents the k-th row of the demand matrix. Additionally, we let

$$s^{n,i}_k = -b^{n,i}_k, \quad \forall n \in [\mathsf{N}], i \in [\mathsf{K}_{\mathrm{c}}].$$

We also set all other variables in the problem to zero. It is easy to see that the solutions constructed in this way satisfy the constraints shown in (82). However, these solutions make the column vectors in the $\mathbf{S}$ matrix linearly dependent, which prevents them from satisfying (66). Therefore, the uv solutions cannot be used, and thus the system of linear equations in (82) should have at least $\mathsf{uv} + \mathsf{m} + \mathsf{u} - 1$ linearly independent solutions. This gives the following inequalities:

$$\begin{aligned} &\mathsf{uK} \geq \mathsf{uv} + \mathsf{m} + \mathsf{u} - 1, \\ &\mathsf{N} \geq \mathsf{u}(\mathsf{N}_{\mathrm{r}} - \mathsf{m} - \mathsf{u} + 1) + \frac{\mathsf{m} + \mathsf{u} - 1}{\mathsf{u}}. \end{aligned} \tag{83}$$

Acknowledgment

We are indebted to Michael Shell for maintaining and improving `IEEEtran.cls`.

References

[1] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 1175–1191. [Online]. Available: https://doi.org/10.1145/3133956.3133982

[2] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, p. 612–613, Nov. 1979. [Online]. Available: https://doi.org/10.1145/359168.359176

[3] T. P. Pedersen, "Non-interactive and information-theoretic secure verifiable secret sharing," in *Annual international cryptology conference*. Springer, 1991, pp. 129–140.

[4] R. Cramer, I. Damgård, and S. Dziembowski, "On the complexity of verifiable secret sharing and multiparty computation," in *Proceedings of the thirty-second annual ACM symposium on Theory of computing*, 2000, pp. 325–334.

[5] D. Chaum, C. Crépeau, and I. Damgard, "Multiparty unconditionally secure protocols," in *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing*, ser. STOC '88. New York, NY, USA: Association for Computing Machinery, 1988, p. 11–19. [Online]. Available: https://doi.org/10.1145/62212.62214

[6] X. Yue, H. Wang, D. Jin, M. Li, and W. Jiang, "Healthcare data gateways: found healthcare intelligence on blockchain with novel privacy risk control," *Journal of medical systems*, vol. 40, pp. 1–8, 2016.

[7] H. Fereidooni, S. Marchal, M. Miettinen, A. Mirhoseini, H. Möllering, T. D. Nguyen, P. Rieger, A.-R. Sadeghi, T. Schneider, H. Yalame, and S. Zeitouni, "Safelearn: Secure aggregation for private federated learning," in *2021 IEEE Security and Privacy Workshops (SPW)*, 2021, pp. 56–62.

[8] C.-S. Yang, J. So, C. He, S. Li, Q. Yu, and S. Avestimehr, "Lightsecagg: Rethinking secure aggregation in federated learning," *arXiv preprint arXiv:2109.14236*, 2021.

[9] B. Choi, J.-y. Sohn, D.-J. Han, and J. Moon, "Communication-computation efficient secure aggregation for federated learning," *arXiv preprint arXiv:2012.05433*, 2020.

[10] T. Jahani-Nezhad, M. A. Maddah-Ali, S. Li, and G. Caire, "Swiftagg: Communication-efficient and dropout-resistant secure aggregation for federated learning with worst-case security guarantees," in *2022 IEEE International Symposium on Information Theory (ISIT)*. IEEE, 2022, pp. 103–108.

[11] Y. Zhao and H. Sun, "Information theoretic secure aggregation with user dropouts," *IEEE Transactions on Information Theory*, vol. 68, no. 11, pp. 7471–7484, 2022.

[12] K. Wan, H. Sun, M. Ji, and G. Caire, "Distributed linearly separable computation," *IEEE Transactions on Information Theory*, vol. 68, no. 2, pp. 1259–1278, 2021.

[13] R. Tandon, Q. Lei, A. G. Dimakis, and N. Karampatziakis, "Gradient coding: Avoiding stragglers in distributed learning," in *Proceedings of the 34th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, D. Precup and Y. W. Teh, Eds., vol. 70. PMLR, 06–11 Aug 2017, pp. 3368–3376. [Online]. Available: https://proceedings.mlr.press/v70/tandon17a.html

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

[15] T. Jahani-Nezhad and M. A. Maddah-Ali, "Optimal communication-computation trade-off in heterogeneous gradient coding," *IEEE Journal on Selected Areas in Information Theory*, vol. 2, no. 3, pp. 1002–1011, 2021.

[16] 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.

[17] K. Wan, H. Sun, M. Ji, and G. Caire, "On secure distributed linearly separable computation," *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 3, pp. 912–926, 2022.

[18] A. Yu, R. Li, M. Tancik, H. Li, R. Ng, and A. Kanazawa, "Plenoctrees for real-time rendering of neural radiance fields," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2021, pp. 5752–5761.

[19] J. Cong and B. Xiao, "Minimizing computation in convolutional neural networks," in *International conference on artificial neural networks*. Springer, 2014, pp. 281–290.

[20] W. Huang, K. Wan, H. Sun, M. Ji, R. C. Qiu, and G. Caire, "Fundamental limits of distributed linearly separable computation under cyclic assignment," in *2023 IEEE International Symposium on Information Theory (ISIT)*, 2023, pp. 2296–2301.

[21] R. A. DeMillo and R. J. Lipton, "A probabilistic remark on algebraic program testing," *Information processing letters*, vol. 7, no. 4, pp. 193–195, 1978.

[22] R. Zippel, "Probabilistic algorithms for sparse polynomials," in *International symposium on symbolic and algebraic manipulation*. Springer, 1979, pp. 216–226.

[23] J. T. Schwartz, "Fast probabilistic algorithms for verification of polynomial identities," *Journal of the ACM (JACM)*, vol. 27, no. 4, pp. 701–717, 1980.