A Group-Type Distributed Secure Coded Computation Scheme Based on a Secret Sharing

Koki Kazama
Waseda University
Email: kokikazama@aoni.waseda.jp

Toshiyasu Matsushima
Waseda University
Email: toshimat@waseda.jp

***Abstract*—We focus on a distributed secure coded computation scheme for matrix multiplication. We propose new distributed coded computation schemes and evaluate the computation time complexity, the erasure-correcting capability of the overall system, and the information security.**

The full version of this paper is in [1].

I. Introduction

We focus on a distributed computation scheme in which errors are corrected in computing multiplication of two matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ on a finite field $\mathbb{F}_q$. Specifically, the main computer (master) wants to leak no information about the value of $\boldsymbol{A}$, while the value of $\boldsymbol{B}$ is public. In the system of a distributed computation scheme, the master partitions the matrix and distributes them to multiple computers (workers), and (2) workers perform parallel computing. This system has the advantage of decreasing the computation time complexity, while this has the disadvantage of increasing the possibility of occurring erasure in computation and leaking information about $\boldsymbol{A}$. In this paper, the erasure of the result of a worker means that the master does not receive the result for some reasons such as failure of the worker or delay. To eliminate the disadvantages, a distributed secure coded computation scheme (DSCC) [2] uses a threshold secret sharing system (SSS) to correct erasures in distributed computation and to keep the value of $\boldsymbol{A}$ secret. On performance evaluation on DSCCs, (a) computation time complexity (CTC), (b) erasure-correcting capability of the overall system, and (c) information security are important criteria while they are a tradeoff. Considering the reason to construct DSCCs, we would like to propose the DSCC which can compute $\boldsymbol{AB}$ more efficiently than the stand-alone scheme (SA), of which the system computes $\boldsymbol{AB}$ solely. Moreover, the system of the proposed scheme can correct some erasures. Moreover, even if a part of the workers tries to know the value of the matrix $\boldsymbol{A}$, this value cannot be recovered even if some workers collude and collect their information.

In this paper, we propose new distributed coded computation schemes called a *basic scheme* and an *improved scheme* of a *group-type distributed secure coded computation scheme* (GDSCC). We explain the basic scheme here. All workers are equally divided into multiple groups. Each group consists of the same number of workers.

1) When the matrix $\boldsymbol{A}$ is input, the master encodes the matrix $(\boldsymbol{A}^\top, \boldsymbol{R}^\top)^\top$, which consists of the matrix $\boldsymbol{A}$ and its independent random variable matrix $\boldsymbol{R}$, to generate the matrix $\tilde{\boldsymbol{A}}$ as preprocessing. Then, $\tilde{\boldsymbol{A}}$ is divided into n_{A} submatrices, and the submatrices are sent to n_{A} groups one by one. The i-th group receives the i-th submatrix and each worker in the group stores the submatrix.
2) Upon receiving the matrix $\boldsymbol{B}$, the master divides the matrix $\boldsymbol{B}$ into n_{B} submatrices and sends them to each worker. The j-th worker in the i-th group computes the multiplication of the i-th submatrix of $\tilde{\boldsymbol{A}}$ and the j-th submatrix of $\boldsymbol{B}$ and sends the result to the master. In this distributed computing, the master does not receive results from some groups or some workers.
3) The master collects the results received from each worker and decodes them. If the number of erasures is less than a certain constant, the correct value of the $\boldsymbol{AB}$ is obtained.

In 1 above, the matrix $\tilde{\boldsymbol{A}}$ generated using the matrix $\boldsymbol{A}$ and the random variable matrix $\boldsymbol{R}$ is partitioned into submatrices so that the values of the matrix $\boldsymbol{A}$ are kept secret from each worker. Specifically, when the number of workers colluding among groups is less than a certain number, it is guaranteed that no information in the matrix $\boldsymbol{A}$ is leaked. Furthermore, the coding in 1 above allows the master to correct errors and erasures that occur between groups. In addition, the partitioning of the matrix $\boldsymbol{B}$ in 2 above reduces the computation time of each worker. This scheme introduces the concept of worker groups into the previous scheme [2] to decrease the CTC. Note, that the evaluation criteria of the scheme [2] differs from that of the proposed scheme in that it focuses on (b) and (c), but not on (a) directly. Moreover, the improved scheme considers erasures of results of a part of workers in some groups.

Moreover, we evaluate (a) the CTCs, (b) the erasure-correcting capabilities, and (c) the information securities of the basic scheme and the improved scheme of the GSDCC and show the advantages as follows. In the evaluation on (a), we evaluate the CTC of the proposed schemes and that of the SA and we show the condition of parameters (the number of workers and groups) in which the proposed schemes are superior to the SA. We define the CTC as the number of four arithmetic

This research is supported in part by Grant-in-Aid JP17K06446 for Scientific Research (C).

operations (an addition, a subtraction, a multiplication, and an inversion[1]) over $\mathbb{F}_q$ in parallel computing of each worker and decoding of the master. In the evaluation of (b), it is shown that $\boldsymbol{AB}$ can be correctly recovered even if the computation results of some groups are erased. In the evaluation of (c), it is shown that no information of $\boldsymbol{A}$ is leaked in the collusion of less than $h_{\mathrm{A}}-1$ groups. Similarly, we evaluate (a), (b), and (c) of the improved scheme.

II. The Purpose of Constructing a Distributed Coded Computation Scheme

As a preliminary, we define notations and explain the purpose of a distributed secure coded computation scheme. $\mathbb{N}$ denotes the set of all positive integers. For two integers m and n which satisfies $m \leq n$, $[m,n]$ denotes $[m,n] := \{\, m, m+1, \ldots, n \,\}$. $[n]$ denotes $[1,n]$ for any $n \in \mathbb{N}$. All vectors are column vectors except specifically noted. $\boldsymbol{E}^\top$ is the transpose of a matrix $\boldsymbol{E}$. $\mathbb{F}_q$ is a finite field with q elements, where q is a power of a prime number. $\mathbb{F}_q^{n\times b}$ denotes the set of all $n\times b$ matrices over $\mathbb{F}_q$, and $\mathbb{F}_q^n := \mathbb{F}_q^{n\times 1}$. $\boldsymbol{e}_{\cdot j} \in \mathbb{F}_q^n$ denotes the j-th column of a matrix $\boldsymbol{E} \in \mathbb{F}_q^{n\times b}$. $\boldsymbol{e}_{i\cdot} \in \mathbb{F}_q^b$ denotes the transpose of the i-th row vector. Thus the matrix $\boldsymbol{E}$ is $(\boldsymbol{e}_{\cdot 1}, \ldots, \boldsymbol{e}_{\cdot b}) = (\boldsymbol{e}_{1\cdot}, \ldots, \boldsymbol{e}_{n\cdot})^\top$. For any $n, k_{\mathrm{A}} \in \mathbb{N}$, $A \subset [n]$, and $\boldsymbol{G} \in \mathbb{F}_q^{n\times k_{\mathrm{A}}}$, we define a matrix $\boldsymbol{G}_{A\cdot}^\top \in \mathbb{F}_q^{|A|\times k_{\mathrm{A}}}$ as a matrix constructed from all $i(\in A)$-th row $\boldsymbol{g}_{i\cdot}^\top \in \mathbb{F}_q^{1\times k_{\mathrm{A}}}$ of $\boldsymbol{G}$. ? denotes the symbol representing an erasure or decoding failure. We formally define the sum and difference of a and ? as ? for any $a \in \mathbb{F}_q$. $\boldsymbol{I}_n$ denotes an $n\times n$ identity matrix. $\otimes$ denotes the Kronecker product of matrices [3].

We explain the definition of computation time complexity. In this paper, q is a power of 2. Positive integers $n, k_{\mathrm{A}}, k_{\mathrm{B}}, l$ satisfy $2 \leq k_{\mathrm{A}} < n, 2 \leq k_{\mathrm{B}}$ and $2 \leq l$.

In this paper, we consider schemes for computing a multiplication of two matrices, $\boldsymbol{A} \in \mathbb{F}_q^{k_{\mathrm{A}}\times l}$ and $\boldsymbol{B} \in \mathbb{F}_q^{l\times k_{\mathrm{B}}}$. Specifically, the main computer (master) wants to leak no information about the value of $\boldsymbol{A}$, while the value of $\boldsymbol{B}$ is public to all other computers (workers). In this paper, we consider schemes for computing a multiplication of two matrices, $\boldsymbol{A} \in \mathbb{F}_q^{k_{\mathrm{A}}\times l}$ and $\boldsymbol{B} \in \mathbb{F}_q^{l\times k_{\mathrm{B}}}$. One value of $\boldsymbol{A}$ is input to the master only once, and the master store this value. On the other hand, many values of $\boldsymbol{B}$ are input to the master many times, and each time the master attempts to compute the value of the matrix $\boldsymbol{AB}$.

The most simple scheme is as follows.

Definition 2.1: We define a *stand-alone scheme* (SA) as a scheme in which the master computes $\boldsymbol{AB}$ solely from the input $\boldsymbol{A}, \boldsymbol{B}$.

Definition 2.2: We define *computation time complexity* (CTC) of a process as the number of four arithmetic operations over $\mathbb{F}_q$ which the system performs in the process from input to output. A subtraction, an addiction, a multiplication, and an inversion are equally treated as one operation in the evaluation of the CTC. *CTC of the system* is the overall CTC from input $\boldsymbol{B}$ to output $\hat{\boldsymbol{AB}}$. [2]

Proposition 2.1: The CTC of the SA system is $k_{\mathrm{A}}k_{\mathrm{B}}(2l-1)$.

Remark 2.1: The CTCs of a *basic scheme* and an *improved scheme*, defined later, are both the sum of the number of the four arithmetic operations in the Computing Process of each worker and the Decoding Process of the master over $\mathbb{F}_q$.

We would like to compute $\boldsymbol{AB}$ more efficiently than the SA system, i.e. to construct a computing system whose CTC is less than that of the SA system. For this purpose, we focus on a distributed coded computation scheme. In this system, however, erasure may occur in computation. Moreover, information about the value of $\boldsymbol{A}$ may leak to workers. Thus we use a secure distributed coded computation scheme using a $(h_{\mathrm{A}}, n_{\mathrm{A}})$-threshold secret sharing system (SSS). The scheme such as [2] uses a SSS for this reason. We introduce the concept of "groups" into the proposed schemes to reduce the CTC. We call the proposed schemes *group-type distributed secure coded computations* (GDSCCs). We propose a *basic scheme* and an *improved scheme* of GDSCCs. In these schemes, workers are equally divided into multiple groups. In the computing process, matrices are processed by distributed computing in each group. In this process, the computation is performed while keeping the secrecy of the value of A against collusion among some workers of each group.

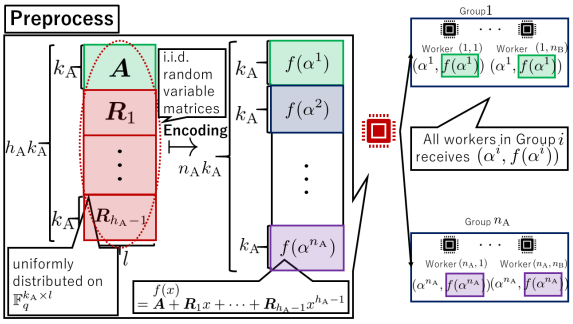

Fig. 1. Basic Scheme (Prepoocess)

III. A Proposition of A Basic Scheme

We propose a construction of a *basic scheme*. We assume $q > n_{\mathrm{A}} (> h_{\mathrm{A}})$. We define $\boldsymbol{G}_{\mathrm{A},i\cdot}$ as $(\boldsymbol{I}_{k_{\mathrm{A}}}, \alpha^i \boldsymbol{I}_{k_{\mathrm{B}}}, \ldots, \alpha^{i(h_{\mathrm{A}}-1)}\boldsymbol{I}_{k_{\mathrm{B}}})^\top$. The matrices $\boldsymbol{G}_{\mathrm{A},1\cdot}^\top, \ldots, \boldsymbol{G}_{\mathrm{A},n_{\mathrm{A}}\cdot}^\top \in \mathbb{F}_q^{k_{\mathrm{A}}\times h_{\mathrm{A}}k_{\mathrm{A}}}$ are stored in the master. Then it holds that $\boldsymbol{G}_{\mathrm{A}} := (\boldsymbol{G}_{\mathrm{A},1\cdot}, \ldots, \boldsymbol{G}_{\mathrm{A},n_{\mathrm{A}}\cdot})^\top = \mathbf{RS}_{n_{\mathrm{A}}\times h_{\mathrm{A}}} \otimes \boldsymbol{I}_{k_{\mathrm{A}}} \in \mathbb{F}_q^{n_{\mathrm{A}}k_{\mathrm{A}}\times h_{\mathrm{A}}k_{\mathrm{A}}}$, where $\mathbf{RS}_{n_{\mathrm{A}}\times h_{\mathrm{A}}}$ denotes a generator matrix of an $(n_{\mathrm{A}}, h_{\mathrm{A}})$ Reed Solomon code over $\mathbb{F}_q$.

The master and workers compute $\boldsymbol{AB}$ as follows [3].

$\langle$*Preprocess*$\rangle$(Figure 1) When the matrix $\boldsymbol{A}$ is input to the master, the master computes a matrix $\tilde{\boldsymbol{A}} = (\tilde{\boldsymbol{A}}_{1\cdot}^\top, \ldots, \tilde{\boldsymbol{A}}_{n_{\mathrm{A}}\cdot}^\top)^\top$,

[1] An inversion is an operation of computing a^{-1} from a. This indicates that an operation of computing ab^{-1} is a combination of an inversion b^{-1} and a multiplication ab^{-1}.

[2] $\hat{\boldsymbol{AB}}$ may not be $\boldsymbol{AB}$ by errors.

[3] This system is similar to the DSCC system of [2]. For example, in Preprocess, the encoding of [2] and that of the proposed scheme are the same. However, there is a difference in whether $\tilde{\boldsymbol{A}}_{i\cdot}$ is sent to each worker i or all workers of each Group i.

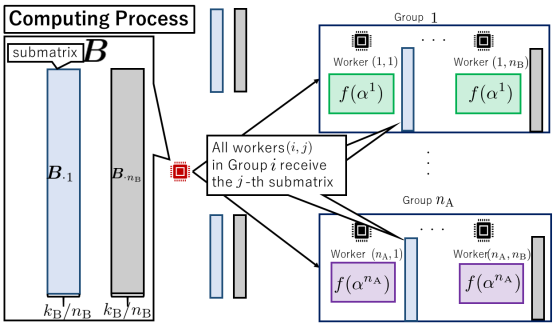

Fig. 2. Basic Scheme(Computing Process)

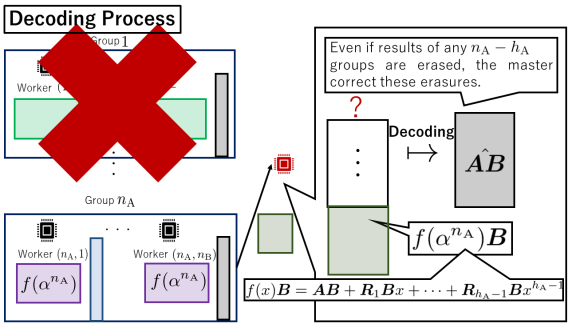

Fig. 3. Basic Scheme (Decoding Process)

where $\tilde{\boldsymbol{A}}_{i\cdot} := \boldsymbol{G}_{\mathrm{A},i\cdot}^{\top}\begin{pmatrix}\boldsymbol{A}\\ \boldsymbol{R}\end{pmatrix} = f(\alpha^i)$, where $f(x) := \boldsymbol{A}+x\boldsymbol{R}_1+\cdots+x^{h_{\mathrm{A}}-1}\boldsymbol{R}_{h_{\mathrm{A}}-1}$, the random variable matrix $\boldsymbol{R}\in\mathbb{F}_q^{(h_{\mathrm{A}}-1)k_{\mathrm{A}}\times l}$ is uniformly distributed on $\mathbb{F}_q^{(h_{\mathrm{A}}-1)k_{\mathrm{A}}\times l}$ and $\boldsymbol{R}_1,\ldots,\boldsymbol{R}_{h_{\mathrm{A}}-1}\in\mathbb{F}_q^{k_{\mathrm{A}}\times l}$ denotes submatrices of $\boldsymbol{R}$ such that $\boldsymbol{R}=(\boldsymbol{R}_1^{\top},\ldots,\boldsymbol{R}_{h_{\mathrm{A}}-1}^{\top})^{\top}$. a_{uv} denotes the (u,v)-th entry of $\boldsymbol{A}$ and r_{huv} denotes the (u,v)-th entry of $\boldsymbol{R}_h$ for any $h\in[h_{\mathrm{A}}-1]$ and $(u,v)\in[k_{\mathrm{A}}]\times[l]$. The master sends $(\alpha^i,\tilde{\boldsymbol{A}}_{i\cdot})$ to Group i. All workers in Group i share and receive this.

⟨*Computing Process*⟩ (Figure 2) At each time when $\boldsymbol{B}$ is input to the master, the master devides $\boldsymbol{B}$ into n_{B} submatrices of size $l\times(k_{\mathrm{B}}/n_{\mathrm{B}})$. $\boldsymbol{B}_{\cdot j}$ denotes j-th submatrix. The master sends $\boldsymbol{B}=(\boldsymbol{B}_{\cdot 1},\ldots,\boldsymbol{B}_{\cdot n_{\mathrm{B}}})$ to each Group. When each worker $(i,j)\in[n_{\mathrm{A}}]\times[n_{\mathrm{B}}]$ recieves $\boldsymbol{B}_{\cdot j}$, each worker (i,j) computes $\boldsymbol{G}_{\mathrm{A},i\cdot}^{\top}\begin{pmatrix}\boldsymbol{A}\\ \boldsymbol{R}\end{pmatrix}\boldsymbol{B}_{\cdot j}$ from the matrix $\boldsymbol{G}_{\mathrm{A},i\cdot}^{\top}\begin{pmatrix}\boldsymbol{A}\\ \boldsymbol{R}\end{pmatrix}$ and $\boldsymbol{B}_{\cdot j}$. Each Group i collects them to obtain $\boldsymbol{G}_{\mathrm{A},i\cdot}^{\top}\begin{pmatrix}\boldsymbol{A}\\ \boldsymbol{R}\end{pmatrix}\boldsymbol{B}=\boldsymbol{G}_{\mathrm{A},i\cdot}^{\top}\begin{pmatrix}\boldsymbol{AB}\\ \boldsymbol{RB}\end{pmatrix}$ and send $(\alpha^i,\boldsymbol{G}_{\mathrm{A},i\cdot}^{\top}\begin{pmatrix}\boldsymbol{AB}\\ \boldsymbol{RB}\end{pmatrix})$ to the master. Let $\{i_1,\ldots,i_h\}$ denote the set of all indices of groups from which the master receives [4], where $h\in[0,n_{\mathrm{A}}]$.

⟨*Decoding Process*⟩ (Figure 3) If $h\le h_{\mathrm{A}}-1$, the master outputs the symbol of computation failure. If $h\ge h_{\mathrm{A}}$, the master performs Algorithm 1 and obtain $\boldsymbol{AB}$ correctly.

[4]For example, it determines whether it receives or not after a certain time.

Algorithm 1 Decoding Algorithm

Require: $(\alpha^{i_h},\boldsymbol{G}_{\mathrm{A},i_h\cdot}^{\top}\begin{pmatrix}\boldsymbol{AB}\\ \boldsymbol{RB}\end{pmatrix})$ for $h\in[h_{\mathrm{A}}]$

Ensure: $\boldsymbol{A}'=\boldsymbol{AB}$

Compute $\prod_{h'=1}^{h_{\mathrm{A}}}\alpha^{i_{h'}}$.
for $h\in[h_{\mathrm{A}}]$ **do**
  Compute $\alpha^{i_h}\prod_{h'\in[h_{\mathrm{A}}]\setminus\{h\}}(\alpha^{i_{h'}}-\alpha^{i_h})$.
  Compute $\frac{\prod_{h'=1}^{h_{\mathrm{A}}}\alpha^{i_{h'}}}{\alpha^{i_h}\prod_{h'\in[h_{\mathrm{A}}]\setminus\{h\}}(\alpha^{i_{h'}}-\alpha^{i_h})}$.
end for
for $(u,v)\in[k_{\mathrm{A}}]\times[k_{\mathrm{B}}]$ **do**
  $a'_{uv}=0$
  for $h\in[h_{\mathrm{A}}]$ **do**
    $a'_{uv}\leftarrow a'_{uv}+f_{uv}(\alpha^{i_h})\frac{\prod_{h'=1}^{h_{\mathrm{A}}}\alpha^{i_{h'}}}{\alpha^{i_h}\prod_{h'\in[h_{\mathrm{A}}]\setminus\{h\}}(\alpha^{i_{h'}}-\alpha^{i_h})}$.
  end for
end for
$\boldsymbol{A}'$ denotes a matrix whose (u,v)-th entry is a'_{uv}.

IV. Performance Evaluation of the Basic Scheme

We evaluate (a) the CTC, (b) the error-correcting capability, and (c) the information security of the basic scheme. In the evaluation of (a), we compare the basic scheme with the SA and describe the conditions for the value of the parameter n_{B} and h_{A} that give the basic scheme an advantage.

We evaluate (c) the information security of this scheme. Let $\boldsymbol{A}':=\boldsymbol{AB}$ and $\boldsymbol{R}'_i:=\boldsymbol{R}_i\boldsymbol{B}$. Also, for each $(u,v)\in[k_{\mathrm{A}}]\times[k_{\mathrm{B}}]$, let a'_{uv} denote the (u,v)-th entry of $\boldsymbol{A}'$ and r'_{iuv} denote the (u,v)-th entry of $\boldsymbol{R}'_i$. Let $f_{uv}(x):=a'_{uv}+r'_{1uv}x+\cdots+r'_{h_{\mathrm{A}}-1,uv}x^{h_{\mathrm{A}}-1}$ be a polynomial of variable x.

Theorem 4.1 (Evaluation of (c) of the Basic Scheme): Let $\boldsymbol{A},\boldsymbol{R}_1,\ldots,\boldsymbol{R}_{h_{\mathrm{A}}-1}$ be i.i.d. random variable matrices that all independently are a uniform distributed over $\mathbb{F}_q^{k_{\mathrm{A}}\times l}$. For any set $\mathcal{P}\subset[n_{\mathrm{A}}]$ of groups which satisfies $|\mathcal{P}|\le h_{\mathrm{A}}-1$,

$$\mathrm{I}\left(\boldsymbol{A}\ ;\ \left(\tilde{\boldsymbol{A}}_{i\cdot}\right)_{i\in\mathcal{P}}\right)=\mathrm{I}\left(\boldsymbol{A}\ ;\ \left(\boldsymbol{G}_{\mathrm{A},i\cdot}^{\top}\begin{pmatrix}\boldsymbol{A}\\ \boldsymbol{R}\end{pmatrix}\right)_{i\in\mathcal{P}}\right)=0.$$

This is similar to those of SSSs.

Proof (Proof of Theorem 4.1): Let $\mathcal{P}\subset[n]$, $|\mathcal{P}|\le h_{\mathrm{A}}-1$.

$$\mathrm{I}\left(\boldsymbol{A}\ ;\ \left(\boldsymbol{G}_{\mathrm{A},i\cdot}^{\top}\begin{pmatrix}\boldsymbol{A}\\ \boldsymbol{R}\end{pmatrix}\right)_{i\in\mathcal{P}}\right)$$
$$=\mathrm{I}\left(\boldsymbol{A}\ ;\ \left(\boldsymbol{A}+\alpha^{i-1}\boldsymbol{R}_1+\ldots+\alpha^{(i-1)(h_{\mathrm{A}}-1)}\boldsymbol{R}_{h_{\mathrm{A}}-1}\right)_{i\in\mathcal{P}}\right)$$
$$\le\sum_{(u,v)\in[k_{\mathrm{A}}]\times[l]}\mathrm{I}\left(a_{uv};\left(\left(a_{uv}+\sum_{l\in[h_{\mathrm{A}}-1]}\alpha^{(i-1)l}r_{luv}\right)_{i\in\mathcal{P}}\right)\right).$$

Thus it is suffices to show that for any $(u,v)\in[k_{\mathrm{A}}]\times[l]$,

$$\mathrm{I}\left(a_{uv}\ ;\ \left(\left(a_{uv}+\sum_{l\in[h_{\mathrm{A}}-1]}\alpha^{(i-1)l}r_{luv}\right)_{i\in\mathcal{P}}\right)\right)=0.$$

Since the i.i.d random variables $a_{uv},r_{1uv},\ldots,r_{h_{\mathrm{A}}-1,uv}$ is uniformly distributed on $\mathbb{F}_q$, this equation holds similarly to the $(h_{\mathrm{A}},n_{\mathrm{A}})$-threshold SSSs. □

We evaluate (b) the erasure-correcting capability.

Theorem 4.2 (Evaluation of (b) of the Basic Scheme): For any set $\mathcal{P} \subset [n_{\rm A}]$ of groups which satisfies $|\mathcal{P}| \geq h_{\rm A}$,

$$\mathrm{H}\left(\boldsymbol{AB}|\left(\tilde{\boldsymbol{A}}_{i\cdot}\boldsymbol{B}\right)_{i\in\mathcal{P}}\right)=\mathrm{H}\left(\boldsymbol{AB}|\left(\boldsymbol{G}_{{\rm A},i\cdot}^{\top}\begin{pmatrix}\boldsymbol{AB}\\ \boldsymbol{RB}\end{pmatrix}\right)_{i\in\mathcal{P}}\right)=0.$$

Proof : Let $\{ i_1, \ldots, i_{h_{\rm A}} \}$ denote the set of groups of which the master receives the results , where $i_h \neq i_{h'}$ for any $h \neq h'$. It is sufficient to show the theorem that the master can decode by Algorithm 1. $\boldsymbol{A}' \coloneqq \boldsymbol{AB}$. Let a'_{uv} denote the (u,v) $(\in [k_{\rm A}] \times [k_{\rm B}])$-th entry. $\boldsymbol{R}'_i \coloneqq \boldsymbol{R}_i\boldsymbol{B}$. Let r'_{iuv} denote the (u,v)-th entry. Let $f_{uv}(x) \coloneqq a'_{uv} + r'_{1uv}x + \cdots + r'_{h_{\rm A}-1,uv}x^{h_{\rm A}-1}$ for any (u,v). Then a $k_{\rm A} \times k_{\rm B}$ matrix whose (u,v)-th entry is $f_{uv}(\alpha^i)$ is $\boldsymbol{G}_{{\rm A},i\cdot}^{\top}\begin{pmatrix}\boldsymbol{A}\\ \boldsymbol{R}\end{pmatrix}\boldsymbol{B}$, which is constructed from the results of Group i. It is because

$$\begin{aligned}&(a'_{uv}+r'_{1uv}\alpha^{i-1}+\ldots+r'_{h_{\rm A}-1,uv}\alpha^{(i-1)(h_{\rm A}-1)})_{(u,v)\in[k_{\rm A}]\times[k_{\rm B}]}\\ &=\boldsymbol{A}'+\boldsymbol{R}'_1\alpha^{i-1}+\cdots+\boldsymbol{R}'_{h_{\rm A}-1}\alpha^{(i-1)(h_{\rm A}-1)}\\ &=\boldsymbol{G}_{{\rm A},i\cdot}^{\top}\begin{pmatrix}\boldsymbol{A}'\\ \boldsymbol{R}'\end{pmatrix}=\boldsymbol{G}_{{\rm A},i\cdot}^{\top}\begin{pmatrix}\boldsymbol{AB}\\ \boldsymbol{RB}\end{pmatrix}=\boldsymbol{G}_{{\rm A},i\cdot}^{\top}\begin{pmatrix}\boldsymbol{A}\\ \boldsymbol{R}\end{pmatrix}\boldsymbol{B}.\end{aligned}$$

For any $(u,v) \in [k_{\rm A}] \times [k_{\rm B}]$, the (u,v)-th entry $a'_{uv} = f_{uv}(0)$ of $\boldsymbol{A}' = \boldsymbol{AB}$ is calculated from the value of $f_{uv}(\alpha^{i_1}), \ldots, f_{uv}(\alpha^{i_{h_{\rm A}-1}})$ by Eq.(1), derived by Lagrange interpolation polynomial.

$$f_{uv}(0)=\sum_{h\in[h_{\rm A}]}f_{uv}(\alpha^{i_h})\frac{\prod_{h'=1}^{h_{\rm A}}\alpha^{i_{h'}}}{\alpha^{i_h}\prod_{h'\in[h_{\rm A}]\setminus\{h\}}(\alpha^{i_{h'}}-\alpha^{i_h})}. \quad (1)$$

Thus the value of $\boldsymbol{A}' = \boldsymbol{AB}$, whose (u,v)-th entry is a'_{uv} , can be calculated by Algorithm 1. □

We evaluiate (a) the CTC of this scheme.

Theorem 4.3 (Evaluation of (a) of the Basic Scheme): $\frac{k_{\rm A}k_{\rm B}(2l-1)}{n_{\rm B}} + k_{\rm A}k_{\rm B}(2h_{\rm A}-1) + 2h_{\rm A}^2 - 1$ is an upper bound of the CTC of the system of the basic scheme.

Proof : The Computing Process of each worker $(i,j) \in [n_{\rm A}] \times [n_{\rm B}]$ can be done with $\frac{k_{\rm A}k_{\rm B}(l-1)}{n_{\rm B}}$ additions and $\frac{k_{\rm A}k_{\rm B}l}{n_{\rm B}}$ multiplications $\mathbb{F}_q$. It is because it can be done with $k_{\rm A}(k_{\rm B}/n_{,B})$ inner products of two vectors of dimension l to compute $\tilde{\boldsymbol{A}}_{i\cdot}\tilde{\boldsymbol{B}}_{\cdot j} \in \mathbb{F}_q^{k_{\rm A}\times\frac{k_{\rm B}}{n_{\rm B}}}$ from $\tilde{\boldsymbol{A}}_{i\cdot} \in \mathbb{F}_q^{k_{\rm A}\times l}$ and $\tilde{\boldsymbol{B}}_{\cdot j} \in \mathbb{F}_q^{l\times\frac{k_{\rm B}}{n_{\rm B}}}$, and an inner product of two vectors of dimension l can be done with $l-1$ additions and l multiplications.

The master performs Algorithm 1 in the Decoding Process. Thus this process can be done with $k_{\rm A}k_{\rm B}h_{\rm A}$ additions, $h_{\rm A}(h_{\rm A}-1)$ substractions, $h_{\rm A}^2 + k_{\rm A}k_{\rm B}h_{\rm A}$ multiplications, and $h_{\rm A}$ inversions over $\mathbb{F}_q$.

Thus the overall processes can be done with $\frac{k_{\rm A}k_{\rm B}(l-1)}{n_{\rm B}} + k_{\rm A}k_{\rm B}h_{\rm A}$ additions, $h_{\rm A}(h_{\rm A}-1)$ substractions, $\frac{k_{\rm A}k_{\rm B}l}{n_{\rm B}} + h_{\rm A}^2 + k_{\rm A}k_{\rm B}h_{\rm A}$ multiplications, and $h_{\rm A}$ inversions over $\mathbb{F}_q$. This indicates that they can be done with $\frac{k_{\rm A}k_{\rm B}(2l-1)}{n_{\rm B}} + 2k_{\rm A}k_{\rm B}h_{\rm A} + 2h_{\rm A}^2 - 1$ operations over $\mathbb{F}_q$. □

Corollary 4.1 (Comparison to (a) the CTC of the SA): The CTC of the basic scheme is less than the CTC of the SA if

$$(k_{\rm B} \geq)n_{\rm B} > \frac{2l-1}{2l-2h_{\rm A}-\frac{2h_{\rm A}^2-1}{k_{\rm A}k_{\rm B}}} \quad (2)$$

and the dominator of Eq.(2) is positive, i.e. $1 \leq h_{\rm A} < (-k_{\rm A}k_{\rm B} + \sqrt{k_{\rm A}^2k_{\rm B}^2 + 4k_{\rm A}k_{\rm B}l + 2})/2$.

Example 4.1: Let $(k_{\rm A}, k_{\rm B}, l) = (100, 293, 100000)$. When $h_{\rm A} \leq 41426$, the dominator is positive. For example, when $h_{\rm A} = 100$, then the CTC of the basic scheme is is less than the CTC of the SA if $2 \leq n_{\rm B} \leq k_{\rm B} = 293$.

For simplicity, we set $n_{\rm A} = n_{\rm B}$. If $k_{\rm A} = k_{\rm B}$, then positive real numbers $(h_{\rm A}^2 + h_{\rm A} - 1)/(k_{\rm A}k_{\rm B})$ and $(2h_{\rm A}-1)/(k_{\rm A}k_{\rm B})$ satisfy $\left|(h_{\rm A}^2 + h_{\rm A} - 1)/(k_{\rm A}k_{\rm B})\right|, |(2h_{\rm A}-1)/(k_{\rm A}k_{\rm B})| \leq 2$ since $h_{\rm A} \leq n_{\rm A} = n_{\rm B} \leq k_{\rm B} = k_{\rm A}$. This indicates that the dominator of the R.H.S of Eq.(2) is positive if $l \geq h_{\rm A} + 2$. Thus, the CTC of the basic scheme is less than that of the SA if $k_{\rm A} = k_{\rm B}$, $l \geq h_{\rm A} + 2$, and $n_{\rm B} \geq l/(l - h_{\rm A})$. Moreover, the CTC of the basic scheme is less than that of the SA if $k_{\rm A} = 1$, $l \geq 2h_{\rm A} + 1$ and $n_{\rm B} \geq l/(l - h_{\rm A})$.

V. A Proposition An Improved Scheme

We propose an *improved scheme*. This is similar to the basic scheme, however, in Computing Process, not only devide $\boldsymbol{B}$ but also encodes the submatrices with an $(n_{\rm B}, h_{\rm B})$ Reed Solomon-like encoder, where a positive number $h_{\rm B}$ $(\in [n_{\rm B}])$ devides $k_{\rm B}$. This causes the improvement of the erasure-correcting capability though the CTC is increased, i.e. in each group from which the master receives the result, the master recovers $\boldsymbol{AB}$ even if results of $n_{\rm B} - h_{\rm B}$ workers are erased. We assume $q > n_{\rm A} (> h_{\rm A})$ and $q > n_{\rm B} (> h_{\rm B})$.

$\boldsymbol{G}_{{\rm B},1\cdot}$,..., $\boldsymbol{G}_{{\rm B},n_{\rm B}\cdot}$ are stored in the master, where a $k_{\rm B} \times (k_{\rm B}/h_{\rm B})$ matrix $\boldsymbol{G}_{{\rm B},j\cdot}$ over $\mathbb{F}_q$ denotes $(\boldsymbol{I}_{k_{\rm B}/h_{\rm B}}, \alpha^j\boldsymbol{I}_{k_{\rm B}/h_{\rm B}}, \ldots, \alpha^{j(h_{\rm B}-1)}\boldsymbol{I}_{k_{\rm B}/h_{\rm B}})^{\top}$. Here it holds that Let $\boldsymbol{G}_{\rm B}$ denotes the matrix $(\boldsymbol{G}_{{\rm B},1\cdot}, \ldots, \boldsymbol{G}_{{\rm B},n_{\rm B}\cdot})^{\top}$. Then it holds that $\boldsymbol{G}_{\rm B} = \mathbf{RS}_{n_{\rm B}\times h_{\rm B}} \otimes \boldsymbol{I}_{k_{\rm B}/h_{\rm B}}$ and $\boldsymbol{BG}_{{\rm B},j\cdot} = \boldsymbol{B}_{\cdot 1} + \alpha^j\boldsymbol{B}_{\cdot 2} + \cdots + \alpha^{j(n_{\rm B}-1)}\boldsymbol{B}_{\cdot h_{\rm B}}$, where $\boldsymbol{B}_{\cdot j}$ is the j-th submatrix of the matrix $\boldsymbol{B}$ when $\boldsymbol{B}$ is equally devided into $h_{\rm B}$ submatrices in the column direction.

This system computes $\boldsymbol{AB}$ as follows.

⟨*Preprocess*⟩ When $\boldsymbol{A}$ is input to the master, the master performs the same process as that of the basic scheme.

⟨*Encoding Process*⟩ When $\boldsymbol{B}$ is input to the master, the master sends $\boldsymbol{B}$ to all workers. After $\boldsymbol{B}$ is input to each worker (i,j), it computes $\tilde{\boldsymbol{A}}_{i\cdot}\boldsymbol{BG}_{{\rm B},j\cdot} = \boldsymbol{G}_{{\rm A},i\cdot}^{\top}\begin{pmatrix}\boldsymbol{AB}\\ \boldsymbol{RB}\end{pmatrix}\boldsymbol{G}_{{\rm B},j\cdot}$.

⟨ *Decoding Process 1-i* ⟩ The master does the process from the process of each group $i \in [n_{\rm A}]$ as follows. By collecting the results of workers of Group i, the master obtains $\left(\alpha^i, \boldsymbol{G}_{{\rm A},i\cdot}^{\top}\begin{pmatrix}\boldsymbol{AB}\\ \boldsymbol{RB}\end{pmatrix}\boldsymbol{G}_{{\rm B},j\cdot}\right)_{j\in\mathcal{P}_{{\rm B},i}}$, where $\mathcal{P}_{{\rm B},i} \subset [n_{\rm B}]$ is the set of all workers from which the master receives the result in Group i. The algorithm for this process is clear from the proof of Theorem 6.1. By decoding this, the master obtains $\boldsymbol{G}_{{\rm A},i\cdot}^{\top}\boldsymbol{AB}$.

⟨ *Decoding Process 2* ⟩ Let $\mathcal{P}_{\rm A} \subset [n_{\rm A}]$ denote the set of all groups whose results are decoded successfully by the master. By collecting all results of groups in $\mathcal{P}_{\rm A}$, the master obtains $\left(\alpha^i, \boldsymbol{G}_{{\rm A},i\cdot}^{\top}\begin{pmatrix}\boldsymbol{AB}\\ \boldsymbol{RB}\end{pmatrix}\right)_{i\in\mathcal{P}_{\rm A}}$.The decoding algorithm is similar to Algorithm 1. Thus the master obtains $\boldsymbol{AB}$.

VI. Performance Evaluation of the Improved Scheme

We evaluate (a) the CTC, (b) the erasure-correcting capability, and (c) the information security of the improved scheme, similarily to Section IV. Since (c) is similar to Theorem 4.1, we only evaluate (a) and (b).

We evaluate (b) the erasure-correcting capability of the improved scheme.

Theorem 6.1 (Evaluation of (b) of the Improved Scheme): For any group set $\mathcal{P}_{\mathrm{A}} \subset [n_{\mathrm{A}}]$ which satisfies $|\mathcal{P}_{\mathrm{A}}| \geq h_{\mathrm{A}}$, and for any worker set $\mathcal{P}_{\mathrm{B},i} \subset [n_{\mathrm{B}}]$ which satisfies $|\mathcal{P}_{\mathrm{B},i}| \geq h_{\mathrm{B}}$ in each Group $i \in \mathcal{P}_{\mathrm{A}}$,

$$\mathrm{H}\left(\boldsymbol{AB} \;\middle|\; \left(\boldsymbol{G}_{\mathrm{A},i\cdot}^{\top}\begin{pmatrix}\boldsymbol{AB}\\ \boldsymbol{RB}\end{pmatrix}\boldsymbol{G}_{\mathrm{B},j\cdot}\right)_{i\in\mathcal{P}_{\mathrm{A}},j\in\mathcal{P}_{\mathrm{B},i}}\right)=0.$$

The proof is in Appendix A in [1].

We evaluate (a) the CTC of the improved scheme. For the evaluation, we assume Assumption 6.1.

Assumption 6.1: q is a prime number such that there exists a positive number b such that $q = 2^{2^b}+1$, and the codeword length is $n = q-1$.

Proposition 6.1: We assume that q and n satisfy Assumption 6.1. Then T_{RS} is an upper bound of the number of the arithmetic operations of the decoding algorithm of an (n,k) Reed Solomon code over $\mathbb{F}_q$ in [4].

$$T_{\mathrm{RS}} \coloneqq (9(\lceil\log_2 k\rceil)^2+9\lceil\log_2 k\rceil+19)2^{\lceil\log_2 k\rceil}-\frac{27}{2}\lceil\log_2 k\rceil + 4k+21n\log_2 n+\frac{31}{2}n+26. \tag{3}$$

The derivation is similar to the same as that of [4]. In this paper, we clarify the number of arithmetic operations. See Appendix B in [1].

The bounded Hamming-distance decoder of an $(n_{\mathrm{B}}, h_{\mathrm{B}})$ Reed Solomon code over $\mathbb{F}_q$ in this paper computes all coefficients of a Lagrange interpolation polynomial [5] of degree $h_{\mathrm{B}}-1$ over $\mathbb{F}_q$.

Lemma 6.1:

$$\frac{k_{\mathrm{B}}(k_{\mathrm{A}}(l-1)+l(k_{\mathrm{B}}-1))}{h_{\mathrm{B}}}+k_{\mathrm{A}}k_{\mathrm{B}}h_{\mathrm{A}}+h_{\mathrm{A}}(h_{\mathrm{A}}-1) + \frac{k_{\mathrm{B}}(k_{\mathrm{A}}+k_{\mathrm{B}})l}{h_{\mathrm{B}}}+h_{\mathrm{A}}^2+k_{\mathrm{A}}k_{\mathrm{B}}h_{\mathrm{A}}+h_{\mathrm{A}}+\frac{k_{\mathrm{A}}k_{\mathrm{B}}\mathrm{T}_{\mathrm{RS}}}{h_{\mathrm{B}}} \tag{4}$$

is an upper bound of the CTC of the improved scheme, where T_{RS} is defined in Eq.(3) if $(n_{\mathrm{B}}, h_{\mathrm{B}}) = (n,k)$.
The proof is in Appendix C in [1].

Theorem 6.2 (Evaluation of (a) of the Improved Scheme): We assume that q, n_{B} and h_{B} satisfy Assumption 6.1 if $n = n_{\mathrm{B}}, k = h_{\mathrm{B}}$. Then

$$\frac{2k_{\mathrm{A}}k_{\mathrm{B}}+2k_{\mathrm{B}}-1}{h_{\mathrm{B}}}l+\left(2k_{\mathrm{A}}k_{\mathrm{B}}h_{\mathrm{A}}+2h_{\mathrm{A}}^2-\frac{k_{\mathrm{A}}k_{\mathrm{B}}}{h_{\mathrm{B}}}+\frac{k_{\mathrm{A}}k_{\mathrm{B}}}{h_{\mathrm{B}}}T_{\mathrm{RS}}\right)$$

is an upper bound of the CTC of the improved scheme. T_{RS} is defined in Eq.(3).

This is clear from Theorem 6.2 and Proposition 6.1.

Corollary 6.1 (Comparison (a) the CTC of the improved scheme with (a) the CTC of the SA): Let $q, n_{\mathrm{B}}, h_{\mathrm{B}}$ satisfy Assumption 6.1 when $n = n_{\mathrm{B}}, k = h_{\mathrm{B}}$. The CTC of the improved scheme is less than that of the SA if

$$l \geq \frac{2k_{\mathrm{A}}k_{\mathrm{B}}h_{\mathrm{A}}+2h_{\mathrm{A}}^2+\frac{k_{\mathrm{A}}k_{\mathrm{B}}}{h_{\mathrm{B}}}(T_{\mathrm{RS}}-1)+1}{2k_{\mathrm{A}}k_{\mathrm{B}}-\frac{1}{h_{\mathrm{B}}}(2k_{\mathrm{A}}k_{\mathrm{B}}+2k_{\mathrm{B}}-1)} \tag{5}$$

$$= \frac{h_{\mathrm{A}}+\frac{1}{k_{\mathrm{A}}k_{\mathrm{B}}}h_{\mathrm{A}}^2+\frac{1}{2h_{\mathrm{B}}}(T_{\mathrm{RS}}-1)+\frac{1}{2k_{\mathrm{A}}k_{\mathrm{B}}}}{1-\frac{1}{h_{\mathrm{B}}}\left(1+\frac{1}{k_{\mathrm{A}}}-\frac{1}{2k_{\mathrm{A}}k_{\mathrm{B}}}\right)} \tag{6}$$

and the dominator of the R.H.S. of Eq.6 is positive, i.e. $h_{\mathrm{B}} > 1+\frac{1}{k_{\mathrm{A}}}-\frac{1}{2k_{\mathrm{A}}k_{\mathrm{B}}}$. T_{RS} is defined in Eq.(3) when $n = n_{\mathrm{B}}, k = h_{\mathrm{B}}$.

Example 6.1: Let $(h_{\mathrm{A}}, h_{\mathrm{B}}, k_{\mathrm{A}}, k_{\mathrm{B}}) = (10, 50, 100, 293)$. . If $h_{\mathrm{B}} \geq 2$, then the dominator of the R.H.S. of Eq.(6) is positive. For example, if $n_{\mathrm{B}} = 2^{2^3}, q = n_{\mathrm{B}}+1$,the minimum value of l such that the improved scheme is superior to the SA is 761.

VII. Conclusion and Future Works

In this paper, we proposed and evaluated new distributed coded computation schemes called the basic scheme and the improved scheme. First, we evaluated (a) their CTC and showed the parameter condition in which they are superior to the SA. Next, we evaluated (b) their erasure-correcting capability and (c) their information securities.

In future works, we would like to improve the proposed schemes. For example, we would like to propose new SDCCs considering not only (a), (b), and (c) but also communication load and the number of workers.

References

[1] K. Kazama and T Matsushima. A group-type distributed secure coded computation scheme based on a secret sharing. https://onl.la/92uCGgV, 2022.

[2] W. Chang and R. Tandon. On the capacity of secure distributed matrix multiplication. In *2018 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–6, 2018.

[3] Hiai F and Petz D. *Introduction to Matrix Analysis and Applications*. Springer. Springer International Publishing Switzerland, 2014.

[4] Alexandre Soro and Jerome Lacan. Fnt-based reed-solomon erasure codes. In *2010 7th IEEE Consumer Communications and Networking Conference*, pp. 1–5, 2010.

[5] H. Niederreiter. *Finite Fields*. Encyclopedia of Mathematics and its Applications. Cambridge University Press, 1984.

[6] Joachim von zur Gathen. *Modern computer algebra / Joachim von zur Gathen, J ́urgen Gerhard. [electronic resource]*. Cambridge University Press, Cambridge, third edition. edition, 2013.

Appendix

A. The Proof of Theorem 6.1

Proof : For any Group $i \in \mathcal{P}_{\mathrm{A}}$, in Decoding Process$1-i$, $\{j_1, \ldots, j_{h_{\mathrm{B}}}\}$ denotes the set of all workers which the master receives the result of. The values $j_1, \ldots, j_{h_{\mathrm{B}}} \in \mathcal{P}_{\mathrm{B},i}$ depend on i. It is sufficient to show that for any $i \in \mathcal{P}_{\mathrm{A}}$, the master can obtain the value of $\boldsymbol{G}_{\mathrm{A},i\cdot}^{\top}\begin{pmatrix}\boldsymbol{AB}\\ \boldsymbol{RB}\end{pmatrix}$ from the value of the matrix $\left(\boldsymbol{G}_{\mathrm{A},i\cdot}^{\top}\begin{pmatrix}\boldsymbol{AB}\\ \boldsymbol{RB}\end{pmatrix}\boldsymbol{G}_{\mathrm{B},j\cdot}\right)_{j\in\mathcal{P}_{\mathrm{B},i}}$ constructed from

the results of the workers $j_1, \ldots, j_{h_\mathrm{B}}$ by an algorithm. It is because it is clear that the master obtain the value of $\boldsymbol{AB}$ from the value of $\left(\boldsymbol{G}_{\mathrm{A},i}^\top \left(\begin{array}{c} \boldsymbol{AB} \\ \boldsymbol{RB} \end{array} \right)\right)_{i \in \mathcal{P}_\mathrm{A}}$, whose proof is similar to that of Theorem 4.2.

We define some notations. Let $\tilde{\boldsymbol{A}}'_i$ be $\boldsymbol{G}_{\mathrm{A},i\cdot}^\top \left(\begin{array}{c} \boldsymbol{AB} \\ \boldsymbol{RB} \end{array} \right) \in \mathbb{F}_q^{k_\mathrm{A} \times k_\mathrm{B}}$. Let $\tilde{a}'_{iuv}$ be the (u,v)-th element of $\tilde{\boldsymbol{A}}'_i$ for each $(u,v) \in [k_\mathrm{A}] \times [k_\mathrm{B}]$. Let $\tilde{\boldsymbol{a}}'_{iu}$ be the transpose of the u-th row of $\tilde{\boldsymbol{A}}'_i$ for each $u \in [k_\mathrm{A}]$, i.e. $(\tilde{a}'_{iu1}, \ldots, \tilde{a}'_{iuk_\mathrm{B}})^\top \in \mathbb{F}_q^{k_\mathrm{B}}$. Then, for each $u \in [k_\mathrm{B}]$,

$$\left((\tilde{\boldsymbol{a}}'_{iu})^\top \left(\begin{array}{ccc} \boldsymbol{G}_{\mathrm{B},j_1\cdot} & \ldots & \boldsymbol{G}_{\mathrm{B},j_{h_\mathrm{B}-1}\cdot} \end{array} \right)\right)^\top \tag{7}$$

$$= \left(\begin{array}{c} \boldsymbol{G}_{\mathrm{B},j_1\cdot}^\top \\ \vdots \\ \boldsymbol{G}_{\mathrm{B},j_{h_\mathrm{B}-1}\cdot}^\top \end{array} \right) \tilde{\boldsymbol{a}}'_{iu} \tag{8}$$

$$= \left(\begin{array}{c} \sum_{c=0}^{h_\mathrm{B}-1} \alpha^{cj_1} \left(\begin{array}{c} \tilde{a}'_{i,u,(h_\mathrm{B}-1)(k_\mathrm{B}/h_\mathrm{B})+1} \\ \vdots \\ \tilde{a}'_{i,u,k_\mathrm{B}} \end{array} \right) \\ \vdots \\ \sum_{c=0}^{h_\mathrm{B}-1} \alpha^{cj_{h_\mathrm{B}}} \left(\begin{array}{c} \tilde{a}'_{i,u,(h_\mathrm{B}-1)(k_\mathrm{B}/h_\mathrm{B})+1} \\ \vdots \\ \tilde{a}'_{i,u,k_\mathrm{B}} \end{array} \right) \end{array} \right) \tag{9}$$

$\in \mathbb{F}_q^{k_\mathrm{B}}$.

For each $w \in [k_\mathrm{B}/h_\mathrm{B}]$, a vector consisting of the $w, w + (k_\mathrm{B}/h_\mathrm{B}), \ldots, w + (k_\mathrm{B}/h_\mathrm{B})(n_\mathrm{B} - 1)$-th elements of the vector in Eq.(9) is

$$\left(\begin{array}{c} \sum_{c=0}^{h_\mathrm{B}-1} \alpha^{j_1 c} \tilde{a}'_{i,u,(h_\mathrm{B}-1)(k_\mathrm{B}/h_\mathrm{B})+w} \\ \vdots \\ \sum_{c=0}^{h_\mathrm{B}-1} \alpha^{j_{h_\mathrm{B}} c} \tilde{a}'_{i,u,(h_\mathrm{B}-1)(k_\mathrm{B}/h_\mathrm{B})+w} \end{array} \right) \tag{10}$$

$$= \left(\begin{array}{ccc} \alpha^{j_1} & \ldots & \alpha^{j_1(h_\mathrm{B}-1)} \\ \vdots & \ddots & \vdots \\ \alpha^{j_{h_\mathrm{B}}} & \ldots & \alpha^{j_{h_\mathrm{B}}(h_\mathrm{B}-1)} \end{array} \right) \left(\begin{array}{c} \tilde{a}'_{i,u,w} \\ \tilde{a}'_{i,u,w+(k_\mathrm{B}/h_\mathrm{B})} \\ \cdots \\ \tilde{a}'_{i,u,w+(h_\mathrm{B}-1)(k_\mathrm{B}/h_\mathrm{B})} \end{array} \right). \tag{11}$$

The vector in Eq.(11) consists of the $j_1, \ldots, j_{h_\mathrm{B}}$-th elements of a codeword (in Eq.(12)) of the $(n_\mathrm{B}, h_\mathrm{B})$ Reed Solomon code over $\mathbb{F}_q$.

$$\mathbf{RS}_{n_\mathrm{B} \times k_\mathrm{B}} \left(\begin{array}{c} \tilde{a}'_{i,u,w} \\ \tilde{a}'_{i,u,w+(k_\mathrm{B}/h_\mathrm{B})} \\ \cdots \\ \tilde{a}'_{i,u,w+(h_\mathrm{B}-1)(k_\mathrm{B}/h_\mathrm{B})} \end{array} \right). \tag{12}$$

This is a codeword of the $(n_\mathrm{B}, h_\mathrm{B})$ Reed Solomon code over $\mathbb{F}_q$ obtained by encoding the vector in Eq. (13).

$$\left(\begin{array}{c} \tilde{a}'_{i,u,w} \\ \tilde{a}'_{i,u,w+(k_\mathrm{B}/h_\mathrm{B})} \\ \cdots \\ \tilde{a}'_{i,u,w+(h_\mathrm{B}-1)(k_\mathrm{B}/h_\mathrm{B})} \end{array} \right). \tag{13}$$

Moreover,

$$\left((\tilde{\boldsymbol{a}}'_{iu})^\top \left(\begin{array}{ccc} \boldsymbol{G}_{\mathrm{B},1\cdot} & \ldots & \boldsymbol{G}_{\mathrm{B},n_\mathrm{B}\cdot} \end{array} \right)\right)^\top = \left((\tilde{\boldsymbol{a}}'_{iu})^\top \boldsymbol{G}_\mathrm{B}^\top\right)^\top \in \mathbb{F}_q^{(k_\mathrm{B}/h_\mathrm{B})n_\mathrm{B}}. \tag{14}$$

The u-th row of the matrix in Eq.(15) is the transpose of the vector in Eq.(14) for each $u \in [k_\mathrm{A}]$.

$$\left(\begin{array}{ccc} \tilde{a}'_{i,1,1} & \ldots & \tilde{a}'_{i,1,k_\mathrm{B}} \\ \vdots & \ddots & \vdots \\ \tilde{a}'_{i,k_\mathrm{A},1} & \ldots & \tilde{a}'_{i,k_\mathrm{A},k_\mathrm{B}} \end{array} \right) \boldsymbol{G}_\mathrm{B}^\top = \boldsymbol{G}_{\mathrm{A},i\cdot}^\top \left(\begin{array}{c} \boldsymbol{AB} \\ \boldsymbol{RB} \end{array} \right) \boldsymbol{G}_\mathrm{B}^\top \in \mathbb{F}_q^{k_\mathrm{A} \times (k_\mathrm{B}/h_\mathrm{B})n_\mathrm{B}} \tag{15}$$

For each $u \in [k_\mathrm{A}]$ and $w \in [k_\mathrm{B}/h_\mathrm{B}]$, using the bounded Hamming erasure decoder of the $(n_\mathrm{B}, h_\mathrm{B})$ Reed Solomon code over $\mathbb{F}_q$, the master obtains the vectors in Eq.(13) from the vector in Eq.(11). Then it obtains $\boldsymbol{G}_{\mathrm{A},i\cdot}^\top \left(\begin{array}{c} \boldsymbol{AB} \\ \boldsymbol{RB} \end{array} \right)$ by rearranging these vectors properly. Thus the proposition is proved. □

B. An Upper Bound of Four Arithmetic Operations of an Erasure-Decoding Algorithm of a Reed Solomon Code

We evaluate the number of (four arithmetic) operations over $\mathbb{F}_q$ in the erasure-decoding algorithm [4] of a Reed-Solomon code. The authors in [4] evaluate the orders of the number of (four arithmetic) operations. On the other hand, we evaluate the number of (four arithmetic) operations over $\mathbb{F}_q$ in each step. Specifically, in Steps 3, 5, and 6, our evaluations of the number of operations are different from those of [4].

We define some notations. Let q, n, k satisfies Assumption 6.1. The encoder of an (n, k) Reed Solomon code over $\mathbb{F}_q$ obtains a codeword $(f(\alpha^1), \ldots, f(\alpha^n))$ by encoding a vector consisting of all coefficients of a polynomial $f(x)$. A polynomial $f(x)$ whose degree is at most $k-1$ is called an information polynomial. We assume that at most $n-k$ symbols of this codeword are erased. Let the $i_1, \ldots, i_k$ $(\in [n])$-th entries are k symbols of all remaining symbols. Let $\boldsymbol{y} \in (\{?\} \cup \mathbb{F}_q)^n$ be a vector obtained by replacing all erasure symbols and ?, i.e. codeword symbols $y_{i_1} = f(\alpha^{i_1}), \ldots, y_{i_k} = f(\alpha^{i_k})$ are remaining. Then, by Lagrange polynomial interpolation,

$$\frac{1}{A(x)} f(x) = \frac{1}{A(x)} \sum_{h=1}^{k} y_{i_h} \frac{\prod_{h' \in [k] \setminus \{h\}} (x - x_{i_{h'}})}{\prod_{h' \in [k] \setminus \{h\}} (x_{i_h} - x_{i_{h'}})} \tag{16}$$

$$= \sum_{h=1}^{k} \frac{\frac{y_{i_h}}{A'(x_{i_h})}}{x - x_{i_h}}, \tag{17}$$

where $A(x) = \sum_{i \in [0,k]} a_i x^i$ is defined as a polynomial $\prod_{h' \in [k]} (x - x_{i_{h'}})$, and $A'(x)$ is defined as a polynomial obtained by formal derivation of $A(x)$ i.e. $A'(x) = \sum_{i \in [0,k-1]} (i+1) a_{i+1} x^i$. See [4] to understand how to derive Eq.(17) from Eq.(16).

We describe the decoding algorithm. The input is an vector $\boldsymbol{y} \in (\{?\} \cup \mathbb{F}_q)^n$. The output is an information polynomial $f(x)$. All of the (four arithmetic) operations (i.e. additions, subtractions, multiplications and inversions) are done over $\mathbb{F}_q$.

1) Compute $A(x)$.
2) Compute the polynomial $A'(x)$, which is a formal derivative of $A(x)$.
3) Compute $A'(\alpha^{i_1}), \ldots, A'(\alpha^{i_k}) \in \mathbb{F}_q$.
4) Compute $\frac{y_{i_1}}{A'(\alpha^{i_1})}, \ldots, \frac{y_{i_k}}{A'(\alpha^{i_k})}$.
5) For a function

$$g(x) = g_0 + g_1 x + \cdots + g_{n-1}x^{n-1} + g_n x^n + \ldots$$

, let $g(x) \mod x^n$ be a polynomial such that

$$g(x) \mod x^n \coloneqq g_0 + g_1 x + \cdots + g_{n-1}x^{n-1}.$$

Calculate $\sum_{h\in[k]} \frac{\frac{y_{i_h}}{A'(\alpha^{i_h})}}{x-\alpha^{i_h}} \mod x^n$.
6) Calculate $f(x) = A(x)\sum_{h\in[k]} \frac{\frac{y_{i_h}}{A'(\alpha^{i_h})}}{x-\alpha^{i_h}} \mod x^n$.

Step 1 can be done with at most $(9(\lceil\log_2 k\rceil)^2+9\lceil\log_2 k\rceil+19)2^{\lceil\log_2 k\rceil}-\frac{27}{2}\lceil\log_2 k\rceil+35$ operations. We explain this later.

Given $A(x)$, in Step 2, $A'(x)$ can be computed by computing $2,\ldots,k$ by $k-1$ additions and then computing $a_{i+1}(i+1)$ for each $i \in [0,k-1]$ with k multiplications. Therefore, Step 2 is can be done with $2k-1$ operations over $\mathbb{F}_q$.

Step 3 can be done with at most $\frac{3}{2}n\log_2 n$ operations. We prove this later.

In Step 4, $y_{i_h}(A'(\alpha^{i_h}))^{-1}$ can be computed for each $h \in [k]$ by k multiplications and k inversions. Thus Step 4 can be done with $2k$ operations.

Step 5 can be done with at most $2n+\frac{3}{2}n\log_2 n$ operations. We prove this later.

Step 6 can be done with $18n\log_2 n+\frac{27}{2}n-8$ operations. We prove this later.

Therefore, the decoding algorithm can be done with at most $(9(\lceil\log_2 k\rceil)^2+9\lceil\log_2 k\rceil+19)2^{\lceil\log_2 k\rceil}-\frac{27}{2}\lceil\log_2 k\rceil+4k+21n\log_2 n+\frac{29}{2}n+26$ operations.

1) The number of operations in Step 1,3,5,6: We count the number of operations in Steps 1,3,5 and 6. In this subsection, we use theorems and definitions in Chapter 8 of [6]. Many theorems in [6] can be applied to communicative rings. In this paper, we apply them to a finite field $\mathbb{F}_q$ and a positive integer n which satisfies Assumption 6.1.

Definition A.1 (Fast Fourier Transform (FFT) [6]): Let α be an element of order n. For an polynomial $f(x)$ of degree at most $n-1$ over $\mathbb{F}_q$, we define

$$\begin{aligned}\mathrm{DFT}_\alpha(f) &\coloneqq (f(\alpha),\ldots,f(\alpha^{n-1}),f(\alpha^n))\\ &= (f(\alpha),\ldots,f(\alpha^{n-1}),f(1)),\end{aligned} \tag{18}$$

The map DFT_α is called *(discrete) Fourier transformation*. A *fast Fourier transformation* (FFT) is a method for computing this map fast.

Proposition A.1 (The computation time of FFTs): A FFT of the polynomial $f(x)$ of degree at most $n-1$ over $\mathbb{F}_q$ can be done with $\frac{3}{2}n\log_2 n$ operartions over $\mathbb{F}_q$.
This is a specialization of Theorem 8.15 in [6].

Proposition A.2: A multiplication of two polynomials $f(x), g(x)$ of degree at most $n-1$ over $\mathbb{F}_q$ can be done with at most $18n\log_2 n+\frac{27}{2}n-8$ operations.
This is a specialization of Corollary 8.19 in [6]. This can be shown directly from the proof of Theorem 8.18 in [6].

We evaluate the number of (four arithemetic) operations in each Step.

Proposition A.3 (The number of operations in Step 1): Given $\alpha^{i_1},\ldots,\alpha^{i_k}$, a polynomial $A(x)$ can be computed with at most

$$(9(\lceil\log_2 k\rceil)^2+9\lceil\log_2 k\rceil+19)2^{\lceil\log_2 k\rceil}-\frac{27}{2}\lceil\log_2 k\rceil+35 \tag{19}$$

operations.
A divide-and-conquer method is used in this proof.

Proof : We prove this in case k is a power of 2. This can be shown similarly in the other cases.

$A(x)$ is computed as follows. The following steps are called Step1-1,1-2,1-2,....

1) Compute $(x-\alpha^{i_1})\ (x-\alpha^{i_2})$, $(x-\alpha^{i_3})\ (x-\alpha^{i_4})$, ... Next, compute $(x-\alpha^{i_1})\ (x-\alpha^{i_2})\ (x-\alpha^{i_3})\ (x-\alpha^{i_4})$, $(x-\alpha^{i_5})\ (x-\alpha^{i_6})\ (x-\alpha^{i_7})\ (x-\alpha^{i_8})$, ...
2) Then, compute $(x-\alpha^{i_1})\ \ldots\ (x-\alpha^{i_8})$, $(x-\alpha^{i_9})\ \ldots\ (x-\alpha^{i_{16}})$, ...
3) By doing similary, $A(x)$ is finally computed.

Let c be $\log_2 k$. For each $i \in [\log_2 k]$, in Step1-i above, 2^{-i+c} multiplications of two polynomials of degree 2^{i-1} are done. Since $2^{i-1} \le 2^i - 1$ for $i \in \mathbb{N}$, an upper bound of the number of operations are derived as follows.

$$\sum_{i=1}^{c} 2^{-i+c}\left(18\cdot i\cdot 2^i+\frac{27}{2}i-8\right) \tag{20}$$

$$= 18\frac{c(c+1)}{2}2^c+\frac{27}{2}\left(\sum_{i=0}^{c-1}2^i(c-i)\right)-8(2^c-1) \tag{21}$$

$$= (9c^2+9c)2^c-8\cdot 2^c+8+\frac{27}{2}\left(2^{c+1}-c-2\right) \tag{22}$$

$$= (9c^2+9c+19)2^c-\frac{27}{2}c+35.\square \tag{23}$$

In Step 3, $A'(\alpha^i)$ are computed for each $i \in [n]$ by FFT. Thus it is shown from Proposition A.1that Step 3 can be done with at most $\frac{3}{2}n\log_2 n$.

In Step 5, for each $h \in [k]$, after computing $n_h \coloneqq -\frac{y_{i_h}}{A'(\alpha^{i_h})}$, the values of $N(1),\ldots,N(\alpha^{n-1})$ are computed by FFT, where $N(x) \coloneqq \sum_{h=1}^{k} n_h x^{i_h}$. Thus it is shown from Proposition A.1 that Step 5 can be done with at most $2n+\frac{3}{2}n\log_2 n$.

It is shown that Step 6 can be done with at most $18n\log_2 n+\frac{27}{2}n-8$ from Proposition A.2.

C. The Proof of Lemma 6.1

Proof : In the evaluations of the number of four arithmetic operations in the decoding algorithm of a Reed Solomon code, let $\mathrm{T}_{\mathrm{RS},+}$, $\mathrm{T}_{\mathrm{RS},-}$, $\mathrm{T}_{\mathrm{RS},*}$, and $\mathrm{T}_{\mathrm{RS},/}$ be the number of additions, subtractions, multiplications, and inversions, respectively. It holds that $\mathrm{T}_{\mathrm{RS}} = \mathrm{T}_{\mathrm{RS},+}+\mathrm{T}_{\mathrm{RS},-}+\mathrm{T}_{\mathrm{RS},*}+\mathrm{T}_{\mathrm{RS},/}$. Each worker performs the Computing Process with $\frac{k_{\mathrm{B}}(k_{\mathrm{A}}(l-1)+l(k_{\mathrm{B}}-1))}{h_{\mathrm{B}}}$ additions, $\frac{k_{\mathrm{B}}(k_{\mathrm{A}}+k_{\mathrm{B}})l}{h_{\mathrm{B}}}$ multiplications. It

is because computing $\tilde{\boldsymbol{B}}_j \coloneqq \boldsymbol{B}\boldsymbol{G}_{\mathrm{B},j\cdot}$ from $\boldsymbol{B} \in \mathbb{F}_q^{l\times k_\mathrm{B}}$ and $\boldsymbol{G}_{\mathrm{B},j\cdot} \in \mathrm{F}_q^{k_\mathrm{B}\times(k_\mathrm{B}/h_\mathrm{B})}$ can be done with $\frac{k_\mathrm{B} l(k_\mathrm{B}-1)}{h_\mathrm{B}}$ additions and $\frac{k_\mathrm{B}^2 l}{h_\mathrm{B}}$ multiplications. Moreover, computing $\tilde{\boldsymbol{A}}_{i\cdot}\tilde{\boldsymbol{B}}_{\cdot j} \in \mathbb{F}_q^{k_\mathrm{A}\times(k_\mathrm{B}/h_\mathrm{B})}$ from $\tilde{\boldsymbol{A}}_{i\cdot} \in \mathbb{F}_q^{k_\mathrm{A}\times l}$ and $\tilde{\boldsymbol{B}}_j \in \mathbb{F}_q^{l\times(k_\mathrm{B}/h_\mathrm{B})}$ can be done with $\frac{k_\mathrm{A}k_\mathrm{B}(l-1)}{h_\mathrm{B}}$ additions and $\frac{k_\mathrm{A}k_\mathrm{B}l}{h_\mathrm{B}}$ multiplications. Thus it can be done with $\frac{k_\mathrm{B}(k_\mathrm{A}(l-1)+l(k_\mathrm{B}-1))}{h_\mathrm{B}}$ additions and $\frac{k_\mathrm{B}(k_\mathrm{A}+k_\mathrm{B})l}{h_\mathrm{B}}$ multiplications.

The master prforms Decoding Process1-i with $\frac{k_\mathrm{A}k_\mathrm{B}\mathrm{T}_{\mathrm{RS},+}}{h_\mathrm{B}}$ additions, $\frac{k_\mathrm{A}k_\mathrm{B}\mathrm{T}_{\mathrm{RS},-}}{h_\mathrm{B}}$ subtractions, $\frac{k_\mathrm{A}k_\mathrm{B}\mathrm{T}_{\mathrm{RS},*}}{h_\mathrm{B}}$ mu;ltiplications, and $\frac{k_\mathrm{A}k_\mathrm{B}\mathrm{T}_{\mathrm{RS},/}}{h_\mathrm{B}}$ inversions. It is because the master performs $k_\mathrm{A}k_\mathrm{B}/h_\mathrm{B}$ decoding algorithms of an $(n_\mathrm{B}, h_\mathrm{B})$ Reed Solomon code over $\mathbb{F}_q$.

The master performs Decoding Process2 (similarly as Algorithm 1) with $k_\mathrm{A}k_\mathrm{B}h_\mathrm{A}$ additions, $h_\mathrm{A}(h_\mathrm{A}-1)$ subtractions, $h_\mathrm{A}^2 + k_\mathrm{A}k_\mathrm{B}h_\mathrm{A}$ multiplications, and h_A inversions $\mathbb{F}_q$.

Therefore, this system needs at most $\frac{k_\mathrm{B}(k_\mathrm{A}(l-1)+l(k_\mathrm{B}-1))}{h_\mathrm{B}} + \frac{k_\mathrm{A}k_\mathrm{B}\mathrm{T}_{\mathrm{RS},+}}{h_\mathrm{B}} + k_\mathrm{A}k_\mathrm{B}h_\mathrm{A}$ additions, $\frac{k_\mathrm{A}k_\mathrm{B}\mathrm{T}_{\mathrm{RS},-}}{h_\mathrm{B}} + h_\mathrm{A}(h_\mathrm{A}-1)$ subtractions, $\frac{k_\mathrm{B}(k_\mathrm{A}+k_\mathrm{B})l}{h_\mathrm{B}} + \frac{k_\mathrm{A}k_\mathrm{B}\mathrm{T}_{\mathrm{RS},*}}{h_\mathrm{B}} + h_\mathrm{A}^2 + k_\mathrm{A}k_\mathrm{B}h_\mathrm{A}$ multiplications, and $\frac{k_\mathrm{A}k_\mathrm{B}\mathrm{T}_{\mathrm{RS},/}}{h_\mathrm{B}} + h_\mathrm{A}$ inversions $\mathbb{F}_q$. The sum of these numbers is the number of four arithmetic operations over $\mathbb{F}_q$ in this syetem. $\square$