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) [6] uses a 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. In addition, even if a part of the workers tries to know the value of the matrix $\boldsymbol{A}$, the value of $\boldsymbol{A}$ cannot be recovered even if some workers collude and collect the information that they have.

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 losses that occur between groups. In addition, the partitioning of the matrix $\boldsymbol{B}$ in 2 above reduces the computation time of each worker. 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 operations (an addition, a subtraction, a multiplication, and an inversion[1]) over $\mathbb{F}_q$ in parallel computing of each worker and decoding

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

[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}.

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. Notations

For two integers m and n which satisfies $m \leq n$, $[m,n]$ denotes $[m,n] := \{\, m, m+1, \ldots, n \,\}$. $[n]$ denotes $[1,n]$. All vectors are column vectors except specifically noted. $\boldsymbol{E}^{\top}$ is the transpotent of a matrix $\boldsymbol{A}$. $\mathbb{F}_q$ 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 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 transpotent of 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 set $A \subset [n]$, 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 the matrix $\boldsymbol{G}$. ? denotes the symbol representing an erasure or decoding failure. We formally define the sum and difference of a and ? as ? for any $a \in \mathbb{F}_q$. $\otimes$ denotes the Kronecker product of matrices.

III. Computation Time Complexity

We explain the problem setting, especially the definition of computation time complexity, of distributed coded computation schemes in this paper. 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 $2 \leq l$.

The schemes in this paper compute $\boldsymbol{AB} \in \mathbb{F}_q^{k_{\mathrm{A}}\times k_{\mathrm{B}}}$. 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 3.1: We define a *stand-alone scheme* (SA) as a scheme in which the master computes $\boldsymbol{AB}$ solely from the input $\boldsymbol{A}, \boldsymbol{B}$.

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.

Definition 3.2: 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 3.1: CTC of the SA system is $k_{\mathrm{A}}k_{\mathrm{B}}(2l-1)$.

The CTCs of the proposed schemes (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 $\mathbb{F}_q$.

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

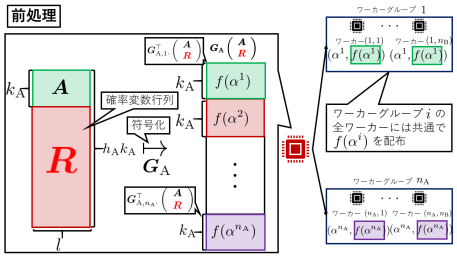

Fig. 1. Basic Scheme (Prepoocess)

IV. A Basic Scheme of the GDSCC

We propose a construction of a *basic scheme* of GDSCC using $(h_{\mathrm{A}}, n_{\mathrm{A}})$-secret sharing systems. 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 $\boldsymbol{G}_{\mathrm{A},1}^{\top}, \ldots, \boldsymbol{G}_{\mathrm{A},n_{\mathrm{A}}}^{\top} \in \mathbb{F}_q^{k_{\mathrm{A}}\times h_{\mathrm{A}}k_{\mathrm{A}}}$ are stored in the master. Then it holds that

$$\begin{pmatrix} \boldsymbol{G}_{\mathrm{A},1\cdot}^{\top} \\ \vdots \\ \boldsymbol{G}_{\mathrm{A},n_{\mathrm{A}}\cdot}^{\top} \end{pmatrix} = \mathbf{RS}_{n_{\mathrm{A}}\times h_{\mathrm{A}}} \otimes \boldsymbol{I}_{k_{\mathrm{A}}} \tag{1}$$

$$= \begin{pmatrix} \boldsymbol{I}_{k_{\mathrm{A}}} & \alpha\boldsymbol{I}_{k_{\mathrm{A}}} & \ldots & \alpha^{h_{\mathrm{A}}-1}\boldsymbol{I}_{k_{\mathrm{A}}} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{I}_{k_{\mathrm{A}}} & \alpha^{n_{\mathrm{A}}}\boldsymbol{I}_{k_{\mathrm{A}}} & \ldots & \alpha^{n_{\mathrm{A}}(h_{\mathrm{A}}-1)}\boldsymbol{I}_{k_{\mathrm{A}}} \end{pmatrix},]$$

where $\mathbf{RS}_{n_{\mathrm{A}}\times h_{\mathrm{A}}}$ is a generator matrix of an $(n_{\mathrm{A}}, h_{\mathrm{A}})$ Reed Solomon code over $\mathbb{F}_q$. $\boldsymbol{G}_{\mathrm{A}} \in \mathbb{F}_q^{n_{\mathrm{A}}k_{\mathrm{A}}\times h_{\mathrm{A}}k_{\mathrm{A}}}$ denotes the matrix in Eq.(1).

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

⟨*Preprocess*⟩(Figure 1)

When the matrix $\boldsymbol{A}$ is input to the master, the master computes $\tilde{\boldsymbol{A}}_{1\cdot}, \ldots, \tilde{\boldsymbol{A}}_{n_{\mathrm{A}}\cdot}$, where $\tilde{\boldsymbol{A}}_{i\cdot} := \boldsymbol{G}_{\mathrm{A},i\cdot}^{\top} \begin{pmatrix} \boldsymbol{A} \\ \boldsymbol{R} \end{pmatrix} = \boldsymbol{A} + \alpha^i \boldsymbol{R}_1 + \cdots + \alpha^{i(h_{\mathrm{A}}-1)}\boldsymbol{R}_{h_{\mathrm{A}}-1}$, $\boldsymbol{R} \in \mathbb{F}_q^{(h_{\mathrm{A}}-1)k_{\mathrm{A}}\times l}$ is a uniformly distributed random varaiable matrix 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 $\boldsymbol{R} = \begin{pmatrix} \boldsymbol{R}_1^{\top} & \ldots & \boldsymbol{R}_{h_{\mathrm{A}}-1}^{\top} \end{pmatrix}^{\top}$. a_{uv} denotes (u,v)-th entry of $\boldsymbol{A}$ and r_{huv} denotes (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 $\boldsymbol{G}_{\mathrm{A},i\cdot}^{\top} \begin{pmatrix} \boldsymbol{A} \\ \boldsymbol{R} \end{pmatrix} \boldsymbol{B}_{\cdot j} \in \mathbb{F}_q^{c_{\mathrm{A},i}\times c_{\mathrm{B},j}}$ 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

[3] This system is similar to the DSCC [6]. 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.

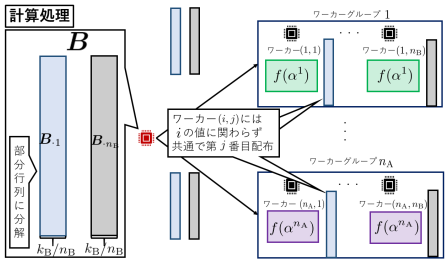

Fig. 2. Basic Scheme(Computing Process)

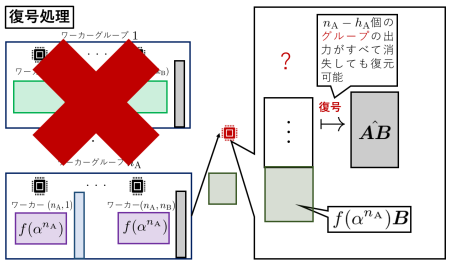

Fig. 3. Basic Scheme (Decoding Process)

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. We assume that the master receives results from h groups and does not receive from the others [4]. $\{i_1, \ldots, i_h\}$ denotes the set of all indices of worker groups from which the master receives.

⟨*Decoding Process*⟩ (Figure 3[5])

If $h \leq h_\mathrm{A} - 1$, then the master outputs the symbol of computation failure. If $h \geq h_\mathrm{A}$, then the master decode $\boldsymbol{G}_{\mathrm{A},i_1\cdot}^\top \begin{pmatrix} \boldsymbol{AB} \\ \boldsymbol{RB} \end{pmatrix}, \ldots, \boldsymbol{G}_{\mathrm{A},i_h\cdot}^\top \begin{pmatrix} \boldsymbol{AB} \\ \boldsymbol{RB} \end{pmatrix}$ by Algorithm 1 and obtain $\boldsymbol{AB}$ correctly.

V. 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 $\boldsymbol{A}' \coloneqq \boldsymbol{AB}$ and $\boldsymbol{R}_i' \coloneqq \boldsymbol{R}_i\boldsymbol{B}$. 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 $f_{uv}(x) \coloneqq a'_{uv} + r'_{1uv}x + \cdots + r'_{h_\mathrm{A}-1,uv}x^{h_\mathrm{A}-1}$ be a polynomial of variable x.

Theorem 5.1 (Evaluation of (c) of th Basic Scheme):

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

[5]$\times$ in the figures denotes the fact that the master did not recieve results from a worker group because of failure of some workers of that group in the figure.

Algorithm 1 Decoding Algorithm

Require: $(\alpha^{i_1}, \boldsymbol{G}_{\mathrm{A},i_1\cdot}^\top \begin{pmatrix} \boldsymbol{AB} \\ \boldsymbol{RB} \end{pmatrix}), \ldots, (\alpha^{i_{h_\mathrm{A}}}, \boldsymbol{G}_{\mathrm{A},i_{h_\mathrm{A}}\cdot}^\top \begin{pmatrix} \boldsymbol{AB} \\ \boldsymbol{RB} \end{pmatrix})$

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

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

Let $\boldsymbol{A}, \boldsymbol{R}_1, \ldots, \boldsymbol{R}_{h-1}$ 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 $|\mathcal{P}| \leq h_\mathrm{A} - 1$,

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

The proof is Appendix A in [1]. This is similar to those of SSSs.

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

Theorem 5.2 (Evaluation of (b) of the basic scheme): For any set $\mathcal{P} \subset [n_\mathrm{A}]$ of groups which satisfies $|\mathcal{P}| \geq h_\mathrm{A}$,

$$\mathrm{H}\left(\boldsymbol{AB} | \left(\tilde{\boldsymbol{A}}_{i\cdot}\boldsymbol{B}\right)_{i\in\mathcal{P}}\right) = \mathrm{H}\left(\boldsymbol{AB} | \left(\boldsymbol{G}_{\mathrm{A},i\cdot}^\top \begin{pmatrix} \boldsymbol{AB} \\ \boldsymbol{RB} \end{pmatrix}\right)_{i\in\mathcal{P}}\right) = 0. \tag{2}$$

The proof is in Appendix C in [1].

We evaluiate (a) the CTC of this scheme.

Theorem 5.3 (Evaluation of (a) of the Basic Scheme): $\frac{k_\mathrm{A}k_\mathrm{B}(2l-1)}{n_\mathrm{B}} + k_\mathrm{A}k_\mathrm{B}(2h_\mathrm{A}-1) + 2h_\mathrm{A}^2 - 1$ is an upper bound of the CTC of the system of the basic scheme.

The proof is in Appendix D in [1].

Corollary 5.1 (Comparison to (a) the CTC of the SA): The CTC of the basic scheme is less than the CTC of the SA if

$$(k_\mathrm{B} \geq) n_\mathrm{B} > \frac{2l-1}{2l - 2h_\mathrm{A} - \frac{2h_\mathrm{A}^2 - 1}{k_\mathrm{A}k_\mathrm{B}}} \tag{3}$$

and the dominator of Eq.(3) is positive, i.e. $1 \leq h_\mathrm{A} < (-k_\mathrm{A}k_\mathrm{B} + \sqrt{k_\mathrm{A}^2k_\mathrm{B}^2 + 4k_\mathrm{A}k_\mathrm{B}l + 2})/2$.

Example 5.1: Let

$$(k_\mathrm{A}, k_\mathrm{B}, l) = (100, 293, 100000). \tag{4}$$

When $h_\mathrm{A} \leq 41426$, the dominator is positive. For example, when $h_\mathrm{A} = 100$, then the CTC of the basic scheme is is less than the CTC of the SA if $2 \leq n_\mathrm{B} \leq k_\mathrm{B} = 293$.

For simplicity, we set $n_{\mathrm{A}} = n_{\mathrm{B}}$. If $k_{\mathrm{A}} = k_{\mathrm{B}}$, then $\left|\frac{h_{\mathrm{A}}^2+h_{\mathrm{A}}-1}{k_{\mathrm{A}}k_{\mathrm{B}}}\right|, \left|\frac{2h_{\mathrm{A}}-1}{k_{\mathrm{A}}k_{\mathrm{B}}}\right| \leq 2$ since $h_{\mathrm{A}} \leq n_{\mathrm{A}} = n_{\mathrm{B}} \leq k_{\mathrm{B}} = k_{\mathrm{A}}$. This indicates that the dominator of the R.H.S of Eq.(3) is positive if $l \geq h_{\mathrm{A}} + 2$. Thus, the CTC of the basic scheme is less than that of the SA if $k_{\mathrm{A}} = k_{\mathrm{B}}$, $l \geq h_{\mathrm{A}} + 2$, and $n_{\mathrm{B}} \geq \frac{l}{l-h_{\mathrm{A}}}$. Moreover, the CTC of the basic scheme is less than that of the SA if $k_{\mathrm{A}} = 1$, $l \geq 2h_{\mathrm{A}} + 1$ and $n_{\mathrm{B}} \geq \frac{l}{l-h_{\mathrm{A}}}$.

VI. An Improved Scheme of the GDSCC

We propose an *improved scheme* of the GDSCC. This is similar to the basic scheme, however, in Computing Process, not only devide $\boldsymbol{B}$ but also encodes the submatrices with an $(n_{\mathrm{B}}, h_{\mathrm{B}})$ Reed Solomon-like encoder, where a positive number h_{B} $(\in [n_{\mathrm{B}}])$ devides k_{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 the value of $\boldsymbol{AB}$ even if results of $n_{\mathrm{B}} - h_{\mathrm{B}}$ workers are erased. We assume $q > n_{\mathrm{A}} (> h_{\mathrm{A}})$ and $q > n_{\mathrm{B}} (> h_{\mathrm{B}})$ here.

$\boldsymbol{G}_{\mathrm{B},1\cdot}, \ldots, \boldsymbol{G}_{\mathrm{B},n_{\mathrm{B}}\cdot} \in \mathbb{F}_q^{k_{\mathrm{B}} \times (k_{\mathrm{B}}/h_{\mathrm{B}})}$ is stored in the master, where $\boldsymbol{G}_{\mathrm{B},j\cdot}$ denotes $(\boldsymbol{I}_{k_{\mathrm{B}}/h_{\mathrm{B}}}, \alpha^j \boldsymbol{I}_{k_{\mathrm{B}}/h_{\mathrm{B}}}, \ldots, \alpha^{j(h_{\mathrm{B}}-1)} \boldsymbol{I}_{k_{\mathrm{B}}/h_{\mathrm{B}}})^\top$. Here it holds that

$$\begin{pmatrix} \boldsymbol{G}_{\mathrm{B},1\cdot}^\top \\ \vdots \\ \boldsymbol{G}_{\mathrm{B},n_{\mathrm{B}}\cdot}^\top \end{pmatrix} = \mathbf{RS}_{n_{\mathrm{B}} \times h_{\mathrm{B}}} \otimes \boldsymbol{I}_{k_{\mathrm{B}}/h_{\mathrm{B}}} \tag{5}$$

$$= \begin{pmatrix} \boldsymbol{I}_{k_{\mathrm{B}}/h_{\mathrm{B}}} & \alpha \boldsymbol{I}_{k_{\mathrm{B}}/h_{\mathrm{B}}} & \ldots & \alpha^{h_{\mathrm{B}}-1} \boldsymbol{I}_{k_{\mathrm{B}}/h_{\mathrm{B}}} \\ \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{I}_{k_{\mathrm{B}}/h_{\mathrm{B}}} & \alpha^{n_{\mathrm{B}}} \boldsymbol{I}_{k_{\mathrm{B}}/h_{\mathrm{B}}} & \ldots & \alpha^{n_{\mathrm{B}}(h_{\mathrm{B}}-1)} \boldsymbol{I}_{k_{\mathrm{B}}/h_{\mathrm{B}}} \end{pmatrix}.$$

$\boldsymbol{G}_{\mathrm{B}} (\in \mathbb{F}_q^{n_{\mathrm{B}}(k_{\mathrm{B}}/h_{\mathrm{B}}) \times k_{\mathrm{B}}})$ denotes the matrix in the above equation. It holds that

$$\boldsymbol{B}\boldsymbol{G}_{\mathrm{B},j\cdot} = (\boldsymbol{B}_{\cdot 1}, \boldsymbol{B}_{\cdot 2}, \ldots, \boldsymbol{B}_{\cdot h_{\mathrm{B}}}) \begin{pmatrix} \boldsymbol{I}_{k_{\mathrm{B}}/h_{\mathrm{B}}} \\ \alpha^j \boldsymbol{I}_{k_{\mathrm{B}}/h_{\mathrm{B}}} \\ \vdots \\ \vdots \\ \alpha^{(h_{\mathrm{B}}-1)j} \boldsymbol{I}_{k_{\mathrm{B}}/h_{\mathrm{B}}} \end{pmatrix} \tag{6}$$

$$= \boldsymbol{B}_{\cdot 1} + \alpha^j \boldsymbol{B}_{\cdot 2} + \cdots + \alpha^{j(n_{\mathrm{B}}-1)} \boldsymbol{B}_{\cdot h_{\mathrm{B}}}, \tag{7}$$

where $\boldsymbol{B}_{\cdot j}$ is the j $(\in [n_{\mathrm{B}}])$-th submatrix of the matrix $\boldsymbol{B}$ when $\boldsymbol{B}$ is equally devided into h_{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 $\tilde{\boldsymbol{A}}_{i\cdot} \boldsymbol{B} \boldsymbol{G}_{\mathrm{B},j\cdot} = \boldsymbol{G}_{\mathrm{A},i\cdot}^\top \begin{pmatrix} \boldsymbol{AB} \\ \boldsymbol{RB} \end{pmatrix} \boldsymbol{G}_{\mathrm{B},j\cdot}$.

[⟨ *Decoding Process 1-i* ⟩] The master does the process from the process of each group $i \in [n_{\mathrm{A}}]$ as follows. By collecting the results of workers of Group i, the master obtains $\left(\alpha^i, \boldsymbol{G}_{\mathrm{A},i\cdot}^\top \begin{pmatrix} \boldsymbol{AB} \\ \boldsymbol{RB} \end{pmatrix} \boldsymbol{G}_{\mathrm{B},j\cdot}\right)_{j \in \mathcal{P}_{\mathrm{B},i}}$, where $\mathcal{P}_{\mathrm{B},i} \subset [n_{\mathrm{B}}]$ 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 7.1. By decoding this, the master obtains $\boldsymbol{G}_{\mathrm{A},i\cdot}^\top \boldsymbol{AB}$.

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

VII. 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 V. Since (c) is similar to Theorem 5.1, we only evaluate (a) and (b)

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

Theorem 7.1 (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 $i \in \mathcal{P}_{\mathrm{A}}$,

$$\mathrm{H}\left(\boldsymbol{AB} \;\middle|\; \left(\boldsymbol{G}_{\mathrm{A},i\cdot}^\top \begin{pmatrix} \boldsymbol{AB} \\ \boldsymbol{RB} \end{pmatrix} \boldsymbol{G}_{\mathrm{B},j\cdot}\right)_{i \in \mathcal{P}_{\mathrm{A}}, j \in \mathcal{P}_{\mathrm{B},i}}\right) = 0. \tag{8}$$

The proof is in Appendix E in [1].

We evaluate (a) the CTC of the improved scheme. For the evaluation, we assume Assumption 7.1.

Assumption 7.1: 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 7.1: We assume that q and n satisfy Assumption 7.1. 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 [7].

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

The derivation is similar to the same as that of [7]. In this paper, we clarify the number of arithmetic operations. See Appendix F in [1].

To compute the vector in Eq.(26) from Eq.(24),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 [8] of degree $h_{\mathrm{B}} - 1$ over $\mathbb{F}_q$.

Lemma 7.1:

$$\frac{k_{\mathrm{B}}(k_{\mathrm{A}}(l-1)+l(k_{\mathrm{B}}-1))}{h_{\mathrm{B}}}+k_{\mathrm{A}}k_{\mathrm{B}}h_{\mathrm{A}}+h_{\mathrm{A}}(h_{\mathrm{A}}-1)$$
$$+\frac{k_{\mathrm{B}}(k_{\mathrm{A}}+k_{\mathrm{B}})l}{h_{\mathrm{B}}}+h_{\mathrm{A}}^2+k_{\mathrm{A}}k_{\mathrm{B}}h_{\mathrm{A}}+h_{\mathrm{A}}+\frac{k_{\mathrm{A}}k_{\mathrm{B}}\mathrm{T}_{\mathrm{RS}}}{h_{\mathrm{B}}} \quad (10)$$

is an upper bound of the CTC of the improved scheme, where T_{RS} is defined in Eq.(9) if $(n_{\mathrm{B}}, h_{\mathrm{B}}) = (n, k)$.
The proof is in Appendix H in [1].

Theorem 7.2 (Evaluation of (a) of the Improved Scheme): We assume that q, n_{B} and h_{B} satisfy Assumption 7.1 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.(9).

This is clear from Theorem 7.2 and Proposition 7.1.

Corollary 7.1 (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 7.1 when $n = n_{\mathrm{B}}, k = h_{\mathrm{B}}$. The CTC of the improved scheme is less than that of the SA if

$$l \geq \frac{2k_{\mathrm{A}}k_{\mathrm{B}}h_{\mathrm{A}}+2h_{\mathrm{A}}^2+\frac{k_{\mathrm{A}}k_{\mathrm{B}}}{h_{\mathrm{B}}}(T_{\mathrm{RS}}-1)+1}{2k_{\mathrm{A}}k_{\mathrm{B}}-\frac{1}{h_{\mathrm{B}}}(2k_{\mathrm{A}}k_{\mathrm{B}}+2k_{\mathrm{B}}-1)} \quad (11)$$
$$= \frac{h_{\mathrm{A}}+\frac{1}{k_{\mathrm{A}}k_{\mathrm{B}}}h_{\mathrm{A}}^2+\frac{1}{2h_{\mathrm{B}}}(T_{\mathrm{RS}}-1)+\frac{1}{2k_{\mathrm{A}}k_{\mathrm{B}}}}{1-\frac{1}{h_{\mathrm{B}}}\left(1+\frac{1}{k_{\mathrm{A}}}-\frac{1}{2k_{\mathrm{A}}k_{\mathrm{B}}}\right)} \quad (12)$$

and the dominator of the R.H.S. of Eq.12 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.(9) when $n = n_{\mathrm{B}}, k = h_{\mathrm{B}}$.

Example 7.1: We set

$$(h_{\mathrm{A}}, h_{\mathrm{B}}, k_{\mathrm{A}}, k_{\mathrm{B}}) = (10, 50, 100, 293). \quad (13)$$

If $h_{\mathrm{B}} \geq 2$, then the dominator of the R.H.S. of Eq.(12) 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.

VIII. Conclusion and Future Works

In this paper, we proposed and evaluated new distributed coded computation schemes called the basic scheme and the improved scheme of the GSDCC. First, we evaluated (a) the CTC of the GSDCCs and showed the parameter condition in which the GSDCCs 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] Alexandre Soro and Jerome Lacan. Fnt-based reed-solomon erasure codes. In *2010 7th IEEE Consumer Communications and Networking Conference*, pp. 1–5, 2010.
[4] H. Niederreiter. *Finite Fields*. Encyclopedia of Mathematics and its Applications. Cambridge University Press, 1984.
[5] E. M. Gabidulin. Theory of codes with maximum rank distance. *Problems of Information Transmission (English translation of Problemy Peredachi Informatsii)*, Vol. 21, No. 1, pp. 1–12, 1985.
[6] Joachim von zur Gathen. Modern computer algebra / joachim von zur gathen, j´urgen gerhard. [electronic resource], 2013.

Appendix

A. Proof of Theorem 5.1

Proof (Proof of Theorem 5.1): Let $\mathcal{P} \subset [n]$, $|\mathcal{P}| \leq h_{\mathrm{A}}-1$.

$$\mathrm{I}\left(\boldsymbol{A}\ ;\ \left(\boldsymbol{G}_{\mathrm{A},i\cdot}^{\top}\begin{pmatrix}\boldsymbol{A}\\ \boldsymbol{R}\end{pmatrix}\right)_{i\in\mathcal{P}}\right)$$
$$=\mathrm{I}\left(\boldsymbol{A}\ ;\ \left(\boldsymbol{A}+\alpha^{i-1}\boldsymbol{R}_1+\cdots+\alpha^{(i-1)(h_{\mathrm{A}}-1)}\boldsymbol{R}_{h_{\mathrm{A}}-1}\right)_{i\in\mathcal{P}}\right).$$

From the chain rule of mutual informations,

$$\mathrm{I}\left(\boldsymbol{A}\ ;\ \left(\boldsymbol{A}+\alpha^{i-1}\boldsymbol{R}_1+\cdots+\alpha^{(i-1)(h_{\mathrm{A}}-1)}\boldsymbol{R}_{h_{\mathrm{A}}-1}\right)_{i\in\mathcal{P}}\right)$$
$$\leq \sum_{(u,v)\in[k_{\mathrm{A}}]\times[l]}\mathrm{I}\left(a_{uv};\left(\left(a_{uv}+\sum_{l\in[h_{\mathrm{A}}-1]}\alpha^{(i-1)l}r_{luv}\right)_{i\in\mathcal{P}}\right)\right).$$

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 is clear from the previous $(h_{\mathrm{A}}, n_{\mathrm{A}})$-SSSs. □

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

We explain a decoding algorithm of a Gabidulin code [10] Moreover, we derive an upper bound of four arithmetic operations over $\mathbb{F}_q$.

C. The Proof of Theorem 5.2

Proof : マスターは互いに相異なるワーカーグループ $i_1, \ldots, i_{h_{\mathrm{A}}}$ から出力を受信する ($\mathcal{P} \supset \{i_1, \ldots, i_{h_{\mathrm{A}}}\}$) として，以下の復号アルゴリズムで復号が可能なことを示せば，定理が示される．ここで，次のようにノーテーションを定める．

- $\boldsymbol{A}' \coloneqq \boldsymbol{A}\boldsymbol{B}$ とおき，その第 $(u,v)\in[k_{\mathrm{A}}]\times[k_{\mathrm{B}}]$ 成分を a'_{uv} とおく．
- $\boldsymbol{R}'_i \coloneqq \boldsymbol{R}_i\boldsymbol{B}$ とおき，その第 (u,v) 成分を r'_{iuv} とおく．
- 各 (u,v) に対し，$f_{uv}(x) \coloneqq a'_{uv}+r'_{1uv}x+\cdots+r'_{h_{\mathrm{A}}-1,uv}x^{h_{\mathrm{A}}-1}$ を変数 x の多項式とする．

このとき，第 (u,v) 成分が $f_{uv}(\alpha^i)$ である $k_\mathrm{A}\times k_\mathrm{B}$ 行列は，以下のようにして，ワーカーグループ i が出力した結果を集約した行列 $\boldsymbol{G}_{\mathrm{A},i\cdot}^\top\begin{pmatrix}\boldsymbol{A}\\ \boldsymbol{R}\end{pmatrix}\boldsymbol{B}$ であることが示される．

$$(a'_{uv}+r'_{1uv}\alpha^{i-1}+\cdots+r'_{h_\mathrm{A}-1,uv}\alpha^{(i-1)(h_\mathrm{A}-1)})_{(u,v)\in[k_\mathrm{A}]\times[k_\mathrm{B}]} \tag{14}$$

$$=\boldsymbol{A}'+\boldsymbol{R}'_1\alpha^{i-1}+\cdots+\boldsymbol{R}'_{h_\mathrm{A}-1}\alpha^{(i-1)(h_\mathrm{A}-1)} \tag{15}$$

$$=\boldsymbol{G}_{\mathrm{A},i\cdot}^\top\begin{pmatrix}\boldsymbol{A}'\\ \boldsymbol{R}'\end{pmatrix} \tag{16}$$

$$=\boldsymbol{G}_{\mathrm{A},i\cdot}^\top\begin{pmatrix}\boldsymbol{AB}\\ \boldsymbol{RB}\end{pmatrix} \tag{17}$$

$$=\boldsymbol{G}_{\mathrm{A},i\cdot}^\top\begin{pmatrix}\boldsymbol{A}\\ \boldsymbol{R}\end{pmatrix}\boldsymbol{B}. \tag{18}$$

また，各 $(u,v)\in[k_\mathrm{A}]\times[k_\mathrm{B}]$ に対して，$f_{uv}(\alpha^{i_1}),\ldots,f_{uv}(\alpha^{i_{h_\mathrm{A}-1}})$ の値が既知であれば，式 (19) によって，$\boldsymbol{A}'=\boldsymbol{AB}$ の第 (u,v) 成分 $a'_{uv}=f_{uv}(0)$ の値を計算できる．ラグランジュ多項式補間を利用することで，式 (19) の通り計算できる．この式は，ラグランジュ多項式補間から導出される式である．

$$f_{uv}(0)=\sum_{h\in[h_\mathrm{A}]}f_{uv}(\alpha^{i_h})\frac{\prod_{h'=1}^{h_\mathrm{A}}\alpha^{i_{h'}}}{\alpha^{i_h}\prod_{h'\in[h_\mathrm{A}]\setminus\{h\}}(\alpha^{i_{h'}}-\alpha^{i_h})}. \tag{19}$$

以上より，第 (u,v) 成分が a'_{uv}，すなわち式 (19) である行列 $\boldsymbol{A}'=\boldsymbol{AB}$ を Algorithm 1 によって計算することができることが示された．□

D. The Proof of Theorem 5.3

Proof : ワーカー $(i,j)\in[n_\mathrm{A}]\times[n_\mathrm{B}]$ の Computing Process の $\mathbb{F}_q$ 上の演算回数のうち，加算が $\frac{k_\mathrm{A}k_\mathrm{B}(l-1)}{n_\mathrm{B}}$ 回，乗算が $\frac{k_\mathrm{A}k_\mathrm{B}l}{n_\mathrm{B}}$ 回である．実際，$\tilde{\boldsymbol{A}}_{i\cdot}\in\mathbb{F}_q^{k_\mathrm{A}\times l}$ と $\tilde{\boldsymbol{B}}_{\cdot j}\in\mathbb{F}_q^{l\times\frac{k_\mathrm{B}}{n_\mathrm{B}}}$ の積行列 $\tilde{\boldsymbol{A}}_{i\cdot}\tilde{\boldsymbol{B}}_{\cdot j}\in\mathbb{F}_q^{k_\mathrm{A}\times\frac{k_\mathrm{B}}{n_\mathrm{B}}}$ を計算するには，l 次元ベクトル同士の内積を $k_\mathrm{A}(k_\mathrm{B}/n_{,B})$ 回行えばよいことと，l 次元ベクトル同士の内積の加算が $l-1$ 回，乗算が l 回であることから，上述の回数が算出できる．

マスターの Decoding Process は，Algorithm 1 によって行われるので，この処理における $\mathbb{F}_q$ 上の演算回数は，加算が $k_\mathrm{A}k_\mathrm{B}h_\mathrm{A}$ 回，減算が $h_\mathrm{A}(h_\mathrm{A}-1)$ 回，乗算が $h_\mathrm{A}^2+k_\mathrm{A}k_\mathrm{B}h_\mathrm{A}$ 回，逆元を求める演算が h_A 回である．

したがって，$\mathbb{F}_q$ 上の演算回数は，加算が $\frac{k_\mathrm{A}k_\mathrm{B}(l-1)}{n_\mathrm{B}}+k_\mathrm{A}k_\mathrm{B}h_\mathrm{A}$ 回，減算が $h_\mathrm{A}(h_\mathrm{A}-1)$ 回，乗算が $\frac{k_\mathrm{A}k_\mathrm{B}l}{n_\mathrm{B}}+h_\mathrm{A}^2+k_\mathrm{A}k_\mathrm{B}h_\mathrm{A}$ 回，逆元を求める演算が h_A 回である．

加算と減算をそれぞれ同じ 1 回と数えるときは，加算，減算が $\frac{k_\mathrm{A}k_\mathrm{B}(l-1)}{n_\mathrm{B}}+k_\mathrm{A}k_\mathrm{B}h_\mathrm{A}+h_\mathrm{A}(h_\mathrm{A}-1)$ 回，乗算が $\frac{k_\mathrm{A}k_\mathrm{B}l}{n_\mathrm{B}}+h_\mathrm{A}^2+k_\mathrm{A}k_\mathrm{B}h_\mathrm{A}$ 回，逆元を求める演算が h_A 回である．また，$\mathbb{F}_q$ 上の演算回数がすべて等しいときは，$\frac{k_\mathrm{A}k_\mathrm{B}(2l-1)}{n_\mathrm{B}}+2k_\mathrm{A}k_\mathrm{B}h_\mathrm{A}+2h_\mathrm{A}^2-1$ である．□

E. The Proof of Theorem 7.1

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 we can prove that the master obtain the value of $\boldsymbol{AB}$ from the value of $\left(\boldsymbol{G}_{\mathrm{A},i\cdot}^\top\begin{pmatrix}\boldsymbol{AB}\\ \boldsymbol{RB}\end{pmatrix}\right)_{i\in\mathcal{P}_\mathrm{A}}$, whose proof is similar to that of Theorem 5.2.

以下の復号アルゴリズムで復号が可能なことを示せば，定理が示される．ここで，次のようにノーテーションを定める．$\tilde{\boldsymbol{A}}'_i:=\boldsymbol{G}_{\mathrm{A},i\cdot}^\top\begin{pmatrix}\boldsymbol{AB}\\ \boldsymbol{RB}\end{pmatrix}\in\mathbb{F}_q^{k_\mathrm{A}\times k_\mathrm{B}}$．その第 $(u,v)\in[k_\mathrm{A}]\times[k_\mathrm{B}]$ 成分を $\tilde{a}'_{iuv}$ と定義する．さらに，$\tilde{\boldsymbol{A}}'_i$ の第 $u\in[k_\mathrm{A}]$ 行目の転置ベクトル $(\tilde{a}'_{iu1},\ldots,\tilde{a}'_{iuk_\mathrm{B}})^\top\in\mathbb{F}_q^{k_\mathrm{B}}$ を $\tilde{\boldsymbol{a}}'_{iu}$ と定義する．このとき，各 $u\in[k_\mathrm{B}]$ に対して，

$$\left((\tilde{\boldsymbol{a}}'_{iu})^\top\begin{pmatrix}\boldsymbol{G}_{\mathrm{B},j_1\cdot} & \cdots & \boldsymbol{G}_{\mathrm{B},j_{h_\mathrm{B}-1}\cdot}\end{pmatrix}\right)^\top \tag{20}$$

$$=\begin{pmatrix}\boldsymbol{G}_{\mathrm{B},j_1\cdot}^\top\\ \vdots\\ \boldsymbol{G}_{\mathrm{B},j_{h_\mathrm{B}-1}\cdot}^\top\end{pmatrix}\tilde{\boldsymbol{a}}'_{iu} \tag{21}$$

$$=\begin{pmatrix}\sum_{c=0}^{h_\mathrm{B}-1}\alpha^{cj_1}\begin{pmatrix}\tilde{a}'_{i,u,(h_\mathrm{B}-1)(k_\mathrm{B}/h_\mathrm{B})+1}\\ \vdots\\ \tilde{a}'_{i,u,k_\mathrm{B}}\end{pmatrix}\\ \vdots\\ \sum_{c=0}^{h_\mathrm{B}-1}\alpha^{cj_{h_\mathrm{B}}}\begin{pmatrix}\tilde{a}'_{i,u,(h_\mathrm{B}-1)(k_\mathrm{B}/h_\mathrm{B})+1}\\ \vdots\\ \tilde{a}'_{i,u,k_\mathrm{B}}\end{pmatrix}\end{pmatrix} \tag{22}$$

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

各 $w\in[k_\mathrm{B}/h_\mathrm{B}]$ に対して，式 (22) の第 $w,w+(k_\mathrm{B}/h_\mathrm{B}),\ldots,w+(k_\mathrm{B}/h_\mathrm{B})(n_\mathrm{B}-1)$ 成分を取り出してきたベクトルは，

$$\begin{pmatrix}\sum_{c=0}^{h_\mathrm{B}-1}\alpha^{j_1c}\tilde{a}'_{i,u,(h_\mathrm{B}-1)(k_\mathrm{B}/h_\mathrm{B})+w}\\ \vdots\\ \sum_{c=0}^{h_\mathrm{B}-1}\alpha^{j_{h_\mathrm{B}}c}\tilde{a}'_{i,u,(h_\mathrm{B}-1)(k_\mathrm{B}/h_\mathrm{B})+w}\end{pmatrix} \tag{23}$$

$$=\begin{pmatrix}\alpha^{j_1} & \cdots & \alpha^{j_1(h_\mathrm{B}-1)}\\ \vdots & \ddots & \vdots\\ \alpha^{j_{h_\mathrm{B}}} & \cdots & \alpha^{j_{h_\mathrm{B}}(h_\mathrm{B}-1)}\end{pmatrix}\begin{pmatrix}\tilde{a}'_{i,u,w}\\ \tilde{a}'_{i,u,w+(k_\mathrm{B}/h_\mathrm{B})}\\ \cdots\\ \tilde{a}'_{i,u,w+(h_\mathrm{B}-1)(k_\mathrm{B}/h_\mathrm{B})}\end{pmatrix}. \tag{24}$$

式 (24) のベクトルは，$\mathbb{F}_q$ 上 $(n_\mathrm{B},h_\mathrm{B})$ Reed Solomon 符号語（式 (25)）の第 $j_1,\ldots,j_{h_\mathrm{B}}$ 成分を取り出してきたベクトルである．

$$\mathbf{RS}_{n_\mathrm{B}\times k_\mathrm{B}}\begin{pmatrix}\tilde{a}'_{i,u,w}\\ \tilde{a}'_{i,u,w+(k_\mathrm{B}/h_\mathrm{B})}\\ \cdots\\ \tilde{a}'_{i,u,w+(h_\mathrm{B}-1)(k_\mathrm{B}/h_\mathrm{B})}\end{pmatrix}. \tag{25}$$

これは式 (26) のベクトルを $\mathbb{F}_q$ 上 $(n_{\mathrm{B}}, h_{\mathrm{B}})$ Reed Solomon 符号化したものである．

$$\begin{pmatrix} \tilde{a}'_{i,u,w} \\ \tilde{a}'_{i,u,w+(k_{\mathrm{B}}/h_{\mathrm{B}})} \\ \cdots \\ \tilde{a}'_{i,u,w+(h_{\mathrm{B}}-1)(k_{\mathrm{B}}/h_{\mathrm{B}})} \end{pmatrix}. \tag{26}$$

また，各 $w \in [k_{\mathrm{B}}/h_{\mathrm{B}}]$ に対して，式 (22) の第 $w, w+(k_{\mathrm{B}}/h_{\mathrm{B}}), \ldots, w+(k_{\mathrm{B}}/h_{\mathrm{B}})(n_{\mathrm{B}}-1)$ 成分を取り出すと式 (25) のベクトルになるようなベクトルは，次式のベクトルである．

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

これを転置した行ベクトルをすべての $u \in [k_{\mathrm{A}}]$ について並べた行列は，

$$\begin{pmatrix} \tilde{a}'_{i,1,1} & \cdots & \tilde{a}'_{i,1,k_{\mathrm{B}}} \\ \vdots & \ddots & \vdots \\ \tilde{a}'_{i,k_{\mathrm{A}},1} & \cdots & \tilde{a}'_{i,k_{\mathrm{A}},k_{\mathrm{B}}} \end{pmatrix} \boldsymbol{G}_{\mathrm{B}}^\top = \boldsymbol{G}_{\mathrm{A},i\cdot}^\top \begin{pmatrix} \boldsymbol{AB} \\ \boldsymbol{RB} \end{pmatrix} \boldsymbol{G}_{\mathrm{B}}^\top \in \mathbb{F}_q^{k_{\mathrm{A}} \times (k_{\mathrm{B}}/h_{\mathrm{B}})n_{\mathrm{B}}}$$

各 $u \in [k_{\mathrm{A}}]$，各 $w \in [k_{\mathrm{B}}/h_{\mathrm{B}}]$ に対して，$\mathbb{F}_q$ 上 $(n_{\mathrm{B}}, h_{\mathrm{B}})$ Reed Solomon 符号の限界ハミング消失距離復号を行うことで式 (24) から式 (26) のベクトルを得て，それらを並べなおすことで $\boldsymbol{G}_{\mathrm{A},i\cdot}^\top \begin{pmatrix} \boldsymbol{AB} \\ \boldsymbol{RB} \end{pmatrix}$ が得られる．以上より，命題が示された．□

F. Reed Solomon 符号の消失復号アルゴリズムにおける有限体上四則演算の回数

Reed-Solomon 符号の消失復号アルゴリズム [7] における有限体 $\mathbb{F}_q$ 上四則演算回数を論じる．[7] においては演算回数のオーダーのみが論じられているが，本論文では各 Step における回数を論じる．ただし，Step 3,5,6 においては，演算回数評価の考え方が [7] と異なる．

ノーテーションを定義する．q, n, k に関しては仮定 7.1 が成立するとする．$\mathbb{F}_q$ 上 (n, k) Reed Solomon 符号においては，$\mathbb{F}_q$ 上高々 $k-1$ 次多項式 $f(x)$（情報多項式）が符号語 $(f(\alpha^1), \ldots, f(\alpha^n))$ に符号化される．この符号語から高々 $n-k$ 個の符号語シンボルが消失したとする．残ったシンボルのうちの k 個の位置を $i_1, \ldots, i_k \in [n]$ とおく．消失したシンボルを ? と置いた時のベクトルを $\boldsymbol{y} \in (\{?\} \cup \mathbb{F}_q)^n$ とおく．すなわち，符号語シンボル $y_{i_1} = f(\alpha^{i_1}), \ldots, y_{i_k} = f(\alpha^{i_k})$ が残っている．このとき，ラグランジュの多項式補間から，

$$\frac{1}{A(x)} f(x) = \frac{1}{A(x)} \sum_{h=1}^{k} y_{i_h} \frac{\prod_{h' \in [k]\setminus\{h\}} (x - x_{i_{h'}})}{\prod_{h' \in [k]\setminus\{h\}} (x_{i_h} - x_{i_{h'}})} \tag{27}$$

$$= \sum_{h=1}^{k} \frac{\frac{y_{i_h}}{A'(x_{i_h})}}{x - x_{i_h}}. \tag{28}$$

ただし，$A(x) = \sum_{i \in [0,k]} a_i x^i$ を $\prod_{h' \in [k]} (x - x_{i_{h'}})$ と定義し，$A(x)$ を x で (形式的) 微分した多項式 $A'(x)$ を $\sum_{i \in [0,k-1]} (i+1) a_{i+1} x^i$ と定義する．ただし，$i+1$ は $\mathbb{F}_q$ の情報単位元 1 を $i+1$ 回足したものである．式 (27) から式 (28) への変形は [7] を参照せよ．

以上のノーテーションの下，復号アルゴリズムの概要を説明する．入力はベクトル $\boldsymbol{y} \in (\{?\} \cup \mathbb{F}_q)^n$ であり，出力は情報多項式 $f(x)$ である．

1) 多項式 $A(x)$ を計算する．
2) $A(x)$ を x で微分した多項式 $A'(x)$ を計算する．
3) $A'(\alpha^{i_1}), \ldots, A'(\alpha^{i_k}) \in \mathbb{F}_q$ を計算する．
4) $\frac{y_{i_1}}{A'(\alpha^{i_1})}, \ldots, \frac{y_{i_k}}{A'(\alpha^{i_k})}$ を計算する．
5) 高々 $n-1$ 次多項式 $\sum_{h \in [k]} \frac{\frac{y_{i_h}}{A'(\alpha^{i_h})}}{x - \alpha^{i_h}} \mod x^n$ を計算する．ここで，関数 $g(x)$ に対して，$g(x)$ をテイラー展開して得た冪級数のうち，高々 $n-1$ 次以下の部分を表す多項式を $g(x) \mod x^n$ と定義する．
6) $f(x) = A(x) \sum_{h \in [k]} \frac{\frac{y_{i_h}}{A'(\alpha^{i_h})}}{x - \alpha^{i_h}} \mod x^n$ を計算する．

Step 1 の演算回数は高々 $(9(\lceil \log_2 k \rceil)^2 + 9\lceil \log_2 k \rceil + 19)2^{\lceil \log_2 k \rceil} - \frac{27}{2}\lceil \log_2 k \rceil + 35$ 回である．証明は後述する．

Step 2 においては，$A(x)$ が与えられたもとで，$k-1$ 回の加算により $2, \ldots, k$ を求めてから，k 回の乗算により各 $i \in [0, k-1]$ に対して $a_{i+1}(i+1)$ を求めることで，$A'(x)$ を求める．よって，$2k-1$ 回の演算回数により Step 2 は行われる．

Step 3 の演算回数は高々 $\frac{3}{2} n \log_2 n$ 回である．証明は後述する．

Step 4 においては，k 回の乗算と k 回の (乗法的) 逆元を求める演算により，各 $h \in [k]$ に対して $y_{i_h}(A'(\alpha^{i_h}))^{-1}$ を求める．よって，$2k$ 回の演算回数により Step 4 は行われる．

Step 5 の演算回数は高々 $2n + \frac{3}{2} n \log_2 n$ 回である．証明は後述する．

Step 6 の演算回数は高々 $18n \log_2 n + \frac{27}{2} n - 8$ 回である．証明は後述する．

以上より，演算回数は高々 $(9(\lceil \log_2 k \rceil)^2 + 9\lceil \log_2 k \rceil + 19)2^{\lceil \log_2 k \rceil} - \frac{27}{2}\lceil \log_2 k \rceil + 4k + 21n \log_2 n + \frac{29}{2} n + 26$ である．

G. Step 1,3,5,6 の演算回数

Step 1,3,5 の演算回数について説明する．本小節では，[11](Chapter 8) の定義，定理を引用する．[11] に記載されている定理の多くは可換環に対して適用されているが，本論文では仮定 7.1 が成立する有限体 $\mathbb{F}_q$ と正整数 n に限定して論じる．

*Definition A.1 (*離散フーリエ変換 *[11]):* α を位数 n の元とする．$\mathbb{F}_q$ 上の高々 $n-1$ 次多項式 $f(x)$ に対し，

$$\begin{aligned}\mathrm{DFT}_\alpha(f) &\coloneqq (f(\alpha), \ldots, f(\alpha^{n-1}), f(\alpha^n)) \\ &= (f(\alpha), \ldots, f(\alpha^{n-1}), f(1))\end{aligned} \tag{29}$$

と定義する．写像 DFT_α を**離散フーリエ変換**と呼ぶ．
これを高速に計算する手法を**高速フーリエ変換**と呼ぶ．

*Proposition A.1 (*高速フーリエ変換の計算時間*):* $\mathbb{F}_q$ 上の高々 $n-1$ 次多項式 $f(x)$ の体 $\mathbb{F}_q$ 上フーリエ変換は，$\mathbb{F}_q$ 上の四則演算を $\frac{3}{2} n \log_2 n$ 回行うことによって求められる．
これは [11] の Theorem 8.15 のある特殊化である．

Proposition A.2: 2 個の $\mathbb{F}_q$ 上高々 $n-1$ 次多項式 $f(x), g(x)$ の積は高々 $18n \log_2 n + \frac{27}{2} n - 8$ 回で計算可能である．
これは [11] の Corollary 8.19 を特殊化した命題であり，[11] の Theorem 8.18 の証明から直ちに導かれる．

以降，各 Step の演算回数を論じる．

Proposition A.3 (Step 1 の演算回数): $\alpha^{i_1},\dots,\alpha^{i_k}$ が与えられたもと，多項式 $A(x)$ を高々

$$(9(\lceil\log_2 k\rceil)^2+9\lceil\log_2 k\rceil+19)2^{\lceil\log_2 k\rceil}-\frac{27}{2}\lceil\log_2 k\rceil+35 \tag{30}$$

回の演算で求めることができる．
証明には分割統治法を用いる．

Proof : k が 2 の冪である場合にのみ示す．そうでない場合でも同様に示すことが可能である．

$A(x)$ を以下の通りに計算する．以下の Step を Step1-1,1-2,1-2,... とおく．

1) $(x-\alpha^{i_1})\ (x-\alpha^{i_2})$，$(x-\alpha^{i_3})\ (x-\alpha^{i_4})$，... を計算する．次に，$(x-\alpha^{i_1})\ (x-\alpha^{i_2})\ (x-\alpha^{i_3})\ (x-\alpha^{i_4})$，$(x-\alpha^{i_5})\ (x-\alpha^{i_6})\ (x-\alpha^{i_7})\ (x-\alpha^{i_8})$，... を計算する．
2) $(x-\alpha^{i_1})\dots(x-\alpha^{i_8})$，$(x-\alpha^{i_9})\dots(x-\alpha^{i_{16}})$，... を計算する．
3) 以下同様に繰り返すことにより，最終的に $A(x)$ を求める．

以下では，$\log_2 k$ を c とおく．各 $i\in[\log_2 k]$ に対し，上記の Step1-i においては，2^{i-1} 次多項式の積を 2^{-i+c} 回行う．ここで，2^{i-1} 次多項式は $2^{i-1}-1$ 次以下の多項式であるため，以下のように演算回数の上界が求められる．

$$\sum_{i=1}^{c}2^{-i+c}\left(18\cdot i\cdot 2^i+\frac{27}{2}i-8\right) \tag{31}$$

$$=18\frac{c(c+1)}{2}2^c+\frac{27}{2}\left(\sum_{i=0}^{c-1}2^i(c-i)\right)-8(2^c-1) \tag{32}$$

$$=(9c^2+9c)2^c-8\cdot 2^c+8+\frac{27}{2}\left(2^{c+1}-c-2\right) \tag{33}$$

$$=(9c^2+9c+19)2^c-\frac{27}{2}c+35.\square \tag{34}$$

Step 3 では，高速フーリエ変換を用いて $A'(\alpha^i)$，$i\in[n]$ を計算することにより行われる．よって，その演算回数は，命題 A.1 より，高々$\frac{3}{2}n\log_2 n$ であると示される．

Step 5 では，各 $h\in[k]$ に対し $n_h:=-\frac{y_{i_h}}{A'(\alpha^{i_h})}$ を求めてから，$N(x):=\sum_{h=1}^{k}n_h x^{i_h}$ とおいたときの値 $N(1),\dots,N(\alpha^{n-1})$ の値を高速フーリエ変換によって求めることになる．よって，その演算回数は，命題 A.1 より，高々$2n+\frac{3}{2}n\log_2 n$ であると示される．

Step 6 の演算回数は，命題 A.2 より，高々$18n\log_2 n+\frac{27}{2}n-8$ であると示される．

H. The Proof of Lemma 7.1

Proof : $\mathbb{F}_q$ 上 $(n_{\mathrm{B}},h_{\mathrm{B}})$ Reed Solomon 符号の復号アルゴリズムにおいて，$\mathrm{T}_{\mathrm{RS},+}$ を加算の回数，$\mathrm{T}_{\mathrm{RS},-}$ を減算の回数，$\mathrm{T}_{\mathrm{RS},*}$ を乗算の回数，$\mathrm{T}_{\mathrm{RS},/}$ を逆元を求める演算の回数であると定義する．本論文においては，$\mathrm{T}_{\mathrm{RS}}=\mathrm{T}_{\mathrm{RS},+}+\mathrm{T}_{\mathrm{RS},-}+\mathrm{T}_{\mathrm{RS},*}+\mathrm{T}_{\mathrm{RS},/}$ が成立する．各ワーカーの Computing Process においては，加算が $\frac{k_{\mathrm{B}}(k_{\mathrm{A}}+k_{\mathrm{B}})l}{h_{\mathrm{B}}}$ 回，乗算が $\frac{k_{\mathrm{B}}(k_{\mathrm{A}}+k_{\mathrm{B}})l}{h_{\mathrm{B}}}$ 回，かかる．実際，$\boldsymbol{B}\in\mathbb{F}_q^{l\times k_{\mathrm{B}}}$ と $\boldsymbol{G}_{\mathrm{B},j\cdot}\in\mathrm{F}_q^{k_{\mathrm{B}}\times(k_{\mathrm{B}}/h_{\mathrm{B}})}$ の積 $\tilde{\boldsymbol{B}}_j:=\boldsymbol{B}\boldsymbol{G}_{\mathrm{B},j\cdot}$ の計算に加算が $\frac{k_{\mathrm{B}}l(k_{\mathrm{B}}-1)}{h_{\mathrm{B}}}$ 回，乗算が $\frac{k_{\mathrm{B}}^2 l}{h_{\mathrm{B}}}$ 回，また，$\tilde{\boldsymbol{A}}_{i\cdot}\in\mathbb{F}_q^{k_{\mathrm{A}}\times l}$ と $\tilde{\boldsymbol{B}}_j\in\mathbb{F}_q^{l\times(k_{\mathrm{B}}/h_{\mathrm{B}})}$ の積 $\tilde{\boldsymbol{A}}_{i\cdot}\tilde{\boldsymbol{B}}_{\cdot j}\in\mathbb{F}_q^{k_{\mathrm{A}}\times(k_{\mathrm{B}}/h_{\mathrm{B}})}$ の計算に加算が $\frac{k_{\mathrm{A}}k_{\mathrm{B}}(l-1)}{h_{\mathrm{B}}}$ 回，乗算が $\frac{k_{\mathrm{A}}k_{\mathrm{B}}l}{h_{\mathrm{B}}}$ 回かかる．したがって，合計で，加算が $\frac{k_{\mathrm{B}}(k_{\mathrm{A}}(l-1)+l(k_{\mathrm{B}}-1))}{h_{\mathrm{B}}}$ 回，乗算が $\frac{k_{\mathrm{B}}(k_{\mathrm{A}}+k_{\mathrm{B}})l}{h_{\mathrm{B}}}$ 回，かかる．

Decoding Process1-i においては，マスターの $\mathbb{F}_q$ 上演算回数は，加算は $\frac{k_{\mathrm{A}}k_{\mathrm{B}}\mathrm{T}_{\mathrm{RS},+}}{h_{\mathrm{B}}}$ 回，減算は $\frac{k_{\mathrm{A}}k_{\mathrm{B}}\mathrm{T}_{\mathrm{RS},-}}{h_{\mathrm{B}}}$ 回，乗算は $\frac{k_{\mathrm{A}}k_{\mathrm{B}}\mathrm{T}_{\mathrm{RS},*}}{h_{\mathrm{B}}}$ 回，逆元を求める演算は $\frac{k_{\mathrm{A}}k_{\mathrm{B}}\mathrm{T}_{\mathrm{RS},/}}{h_{\mathrm{B}}}$ 回，である．実際，$\mathbb{F}_q$ 上 $(n_{\mathrm{B}},h_{\mathrm{B}})$ Reed Solomon 符号の復号を $k_{\mathrm{A}}k_{\mathrm{B}}/h_{\mathrm{B}}$ 回行うからである．

Decoding Process2 においては，マスターが Algorithm 1 と同様の処理を行うので，$\mathbb{F}_q$ 上の演算回数は，加算が $k_{\mathrm{A}}k_{\mathrm{B}}h_{\mathrm{A}}$ 回，減算が $h_{\mathrm{A}}(h_{\mathrm{A}}-1)$ 回，乗算が $h_{\mathrm{A}}^2+k_{\mathrm{A}}k_{\mathrm{B}}h_{\mathrm{A}}$ 回，逆元を求める演算が h_{A} 回である．

したがって，$\mathbb{F}_q$ 上の演算回数は，システム全体の合計で，加算が $\frac{k_{\mathrm{B}}(k_{\mathrm{A}}(l-1)+l(k_{\mathrm{B}}-1))}{h_{\mathrm{B}}}+\frac{k_{\mathrm{A}}k_{\mathrm{B}}\mathrm{T}_{\mathrm{RS},+}}{h_{\mathrm{B}}}+k_{\mathrm{A}}k_{\mathrm{B}}h_{\mathrm{A}}$ 回，減算が $\frac{k_{\mathrm{A}}k_{\mathrm{B}}\mathrm{T}_{\mathrm{RS},-}}{h_{\mathrm{B}}}+h_{\mathrm{A}}(h_{\mathrm{A}}-1)$ 回，乗算が $\frac{k_{\mathrm{B}}(k_{\mathrm{A}}+k_{\mathrm{B}})l}{h_{\mathrm{B}}}+\frac{k_{\mathrm{A}}k_{\mathrm{B}}\mathrm{T}_{\mathrm{RS},*}}{h_{\mathrm{B}}}+h_{\mathrm{A}}^2+k_{\mathrm{A}}k_{\mathrm{B}}h_{\mathrm{A}}$ 回，逆元を求める演算が $\frac{k_{\mathrm{A}}k_{\mathrm{B}}\mathrm{T}_{\mathrm{RS},/}}{h_{\mathrm{B}}}+h_{\mathrm{A}}$ 回である．この合計が，$\mathbb{F}_q$ 上の演算回数の合計である．$\square$