平成 29 年度

学士学位論文

# 部分復元可能な秘密分散システムの分割方法による復元速度の評価

# Evaluation of Decoding Speed for Dividing Method of Confined Decodable Secret Sharing Scheme System

1180347　　竹中　壮磨

指導教員　　福本　昌弘

2018 年 3 月 16 日

高知工科大学 情報学群

# 要 旨

## 部分復元可能な秘密分散システムの分割方法による復元速度の評価

竹中　壮磨

災害時，バックアップした医療データを外部の医療従事者に提供できれば，被災地での医療行為を円滑に行うことができる．よって，バックアップした医療データから医療行為に必要な情報のみを提供できる仕組みがあればよい．そこで，部分復元可能な秘密分散システムが提案された．しかし，提案されたシステムは，演算量が大きく，復元速度が遅いため，緊急的に医療情報を必要とする災害時には適していない．

本研究では災害時に適した速度で医療データを部分復元できるように，データを一定のサイズごとに分割する方法を提案し，分割サイズによる復元速度の評価を行っている．提案方法では，分割サイズ $F$ に依存した $GF(2^F)$ 上で部分復元を行えるようにしている．そして，分割サイズを小さくするにつれて，部分復元にかかる演算量を小さくできる．しかし，分割サイズを小さくしすぎると，復元に必要なシェアファイル数が増えてしまい，ファイル入力処理が遅くなることを明らかにしている．

分割サイズによる復元速度の評価を行った結果，分割サイズを小さくすれば，並列処理の実装や CPU の性能によって，復元したいデータサイズが大きい場合でも十分高速に部分復元できることを明らかにしている．

**キーワード**　秘密分散法，部分復元

# Abstract

## Evaluation of Decoding Speed for Dividing Method of Confined Decodable Secret Sharing Scheme System

TAKENAKA Soma

Disaster Medical Assistance Team (DMAT) is a rescue team for disaster situations. In order to optimizing their rescue performance, an essential core of DMAT have to requests information of curing patients from data centers, such as, blood type and medicines information. Therefore, it is only necessary to have a mecahism that can provide only the necessary data, for curing victim, from data centers. In order to provide a selectively feature, dividing method of confined decodable secret sharing scheme system has been proposed. However, a common shortcoming of the pervious algorithm has been clearly recognized as large amount of computation and slow decoding speed. The pervious algorithm is not suitable for disasters requiring medical information urgently.

In this research, the novel and computationally efficient algorithm is proposed and the decoding speed by dividing size is evaluated. The proposed algorithm to divide data by a certain size, so that medical information can be confined decoded at an optimized speed for disaster situation usage. In proposed method, confined decode on the extension field $GF(2^F)$ depends on dividing size $F$. As a result, the computational amount to decode becomes smaller as the dividing size becomes smaller. However, as the dividing size becomes smaller, share files necessary for decoding increase and this causes share files to load slower.

As a result, the decoding speed by dividing speed it can be decoded at a sufficiety

high speed, by reducing dividing size, implementationing parallel processing and using high performance CPU even if the data is large.

***key words*** secret sharing scheme, confined decoding

# 目次

# 図目次

# 表目次

# 第 1 章

# はじめに

## 1.1　本研究の背景と目的

災害時，バックアップした医療データを外部の医療従事者に提供できれば，被災地での医療行為を円滑に行うことができる．よって，バックアップした医療データから医療行為に必要な情報のみを提供できる仕組みがあればよい．そこで，バックアップデータから必要な情報のみを復元できる部分復元可能な秘密分散システムが提案された [1]．しかし，提案されたシステムは，演算量が大きく，復元速度が遅いため，緊急的に医療情報を必要とする災害時には適していない．本研究では災害時に適した速度で，医療データを部分復元できるように，データを一定のサイズごとに分割する方法を提案する．そして, 部分復元可能な秘密分散システムの分割サイズによる復元速度の評価を行う．

## 1.2　本論文の構成

本節では本論文の構成について述べる．2 章では，提案された部分復元アルゴリズムについて述べたあと，部分復元可能な秘密分散システムの復元にかかる演算量について述べる．3 章では，サイズによる分割を行った部分復元可能な秘密分散システムを提案した後，提案した秘密分散システムの復元にかかる演算量について述べる．4 章では，提案システムの実装について述べる．5 章では，提案した秘密分散システムの評価実験を行い，分割サイズによる復元速度の評価を行う．6 章では，本研究をまとめ，今後の課題を述べる．

# 第 2 章

# 部分復元可能な秘密分散システム

東日本大震災の際に，患者の医療情報が流出し被災地での医療行為に支障が出た．そこで，高知県では広域災害に備えて電子カルテを遠隔地に分散バックアップする取り組みが行われている．

災害時は負傷者の増加と，被災地域での病院施設の機能低下が想定されるため，被災地域の病院だけでは負傷者を捌くことは困難である．そこで，被災地域外から派遣された外部の医師が，被災地での医療行為の手助けを行う．その際，外部の医師に患者の医療情報を提供できれば，適切な治療が行える．よって，バックアップした医療データから医療行為に必要な情報のみを提供できる仕組みがあればよい．そこで，部分復元可能な秘密分散システムが提案された [1]． しかし，提案されたシステムは演算量が大きいため，復元速度が遅く，緊急的に医療情報を必要とする災害時には適していない．本章では，提案された部分復元可能な秘密分散システムに用いられている部分復元アルゴリズムについて述べる．そして，部分復元可能な秘密分散システムの復元にかかる演算量について述べる．

## 2.1 部分復元アルゴリズム

秘密分散法は分散したデータを部分的に復元することはできない．そこで，(k,n) しきい値秘密分散法 [2](付録 A を参照) を用いた部分復元アルゴリズムが提案された．本節では，部分復元アルゴリズムについて述べる．部分復元アルゴリズムは分散と復元で構成される．部分復元アルゴリズムの流れを図 2.1 に示す．

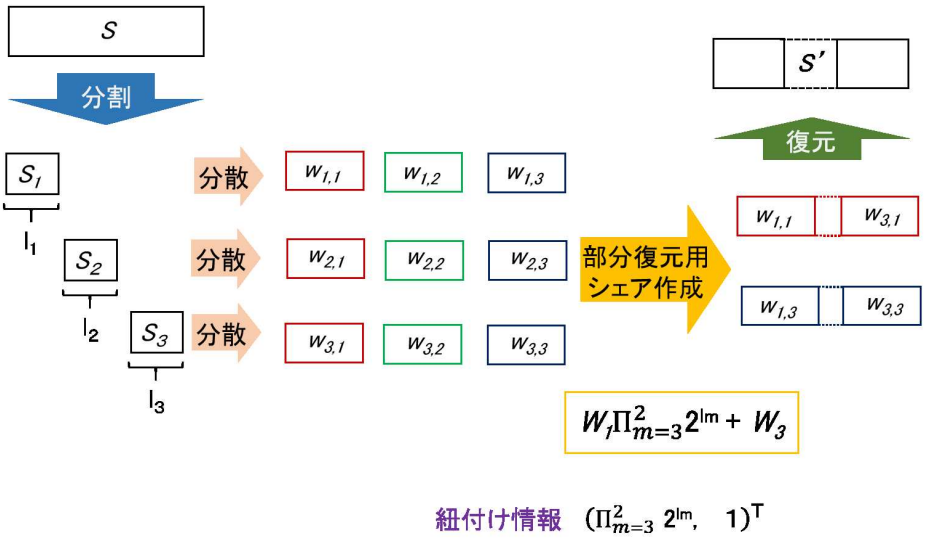


図 2.1　部分復元アルゴリズムの流れ

### 2.1.1　分散

まず，分散するデータ $S$ を意味のある項目ごとに $d$ 個に分け，分けたデータを $S_i(i = 1, 2, \cdots, d)$ とする．このようにデータを意味のある項目ごとに分けることを分割とする．$S_i$ をそれぞれ $(k, n)$ しきい値秘密分散法を用いてシェアを生成し分散する．$S_i$ のシェア集合を $W_i = \{w_{i,1}, \cdots, w_{i,n}\}$ とする．生成したシェアを正しい順番に並んだデータとして復元できるように，正しく紐付ける情報を作成し紐付け情報とする．紐付け情報は分割したデータの長さ $l_i$ を要素として対応する項目ごとに並べたものである．秘匿したいデータ内容にあたる部分を 0 となるように紐付け情報を作成する．

### 2.1.2　復元

シェア集合 $W_1, \cdots, W_d$ から復元したいシェアのみをしきい値以上集め，秘匿部分を選択した紐付け情報を入手する．集めたシェアと紐付け情報を用いて，部分復元用シェアを作成する．部分復元用シェアに復元操作を行うことで，部分復元データ $S'$ を得ることができる．

### 2.1.3 部分復元の手順

提案された部分復元アルゴリズムについて述べる．秘密分散するデータ $S$ を意味のある項目ごとに分割し，分割したデータを $S_i(i = 1, 2, \cdots, d)$, 分割データサイズを $l_i$ とする．分割したデータ $S$ は，

$$S = S_1 \prod_{m=2}^{d} 2^{l_m} + S_2 \prod_{m=3}^{d} 2^{l_m} + \cdots + S_{d-1} 2^{l_d} + S_d \tag{2.1}$$

と表すことができる．分割データ $S_i$ の集合を

$$S_{all} = \{S_1, S_2, \cdots, S_d\}$$

とおき，復元したいデータの集合を $S_{all}$ の部分集合として

$$S_{all} \supseteqq S_c = \{S_{c_1}, S_{c_2}, \cdots, S_{c_e}\}(1 \leqq e \leqq d)$$

としたとき，復元したいデータ $S'$ は

$$S' = S_{c_1} \prod_{m=2}^{e} 2^{l_m} + S_{c_2} \prod_{m=3}^{e} 2^{l_m} + \cdots + S_{c_e} \tag{2.2}$$

と表すことができる．$S_c$ の任意の要素である $S_{c_p}$ $(1 \leqq p \leqq e)$ のシェア集合を

$$W_{c_p} = \{w_{c_{p,1}}, w_{c_{p,2}}, \cdots, w_{c_{p,n}}\}(1 \leqq p \leqq e)$$

とおく．復元したいデータのシェア集合 $W_{c_p}$ を集めたものをシェア集合 $G_c$

$$G_c = \{W_{c_1}, W_{c_2}, \cdots, W_{c_e}\}$$

とおき，シェア集合 $G_c$ の要素 $W_{c_p}$ を部分復元シェアとなるように計算するための紐付け情報 $\boldsymbol{u_c}$

$$\boldsymbol{u_c} = \begin{pmatrix} \prod_{m=2}^{e} 2^{l_m} \\ \prod_{m=3}^{e} 2^{l_m} \\ \vdots \\ 2^{l_e} \\ 1 \end{pmatrix} \tag{2.3}$$

を作成する．部分復元の詳細な手順を以下に示す．

1. $S$ を式 (2.1) を用いて分割し，$S_i(i = 1, 2, \cdots, d)$ を作成する．$S_i$ のデータサイズを $l_i$ とする．
2. データ $S$ 以上のデータが表現できる $GF(2^m)$ を選択する．
3. $GF(2^m)$ の元 $-\{0\}$ の集合 $X = \{x_1, x_2, \cdots, x_n\}$ から $n * k$ の vandermonde 行列 $\boldsymbol{X}$ を作成する (ただし，$GF(2^m)$ 上で演算を行う)．

$$\boldsymbol{X} = \begin{pmatrix} 1 & x_1 & \dots & x_1^{k-1} \\ 1 & x_2 & \dots & x_2^{k-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & \dots & x_n^{k-1} \end{pmatrix} \tag{2.4}$$

また，分割データ $S_i$ と $GF(2^m)$ の元 $-\{0\}$ の集合 $R = \{r_{i,1}, r_{i,2}, \cdots, r_{i,k-1}\}$ からランダムに選択し，ベクトル $\boldsymbol{a_i}$ を作成する．

$$\boldsymbol{a_i} = \begin{pmatrix} S_i \\ r_{i,1} \\ \vdots \\ r_{i,k-1} \end{pmatrix} \tag{2.5}$$

4. $\boldsymbol{X}$ と $\boldsymbol{a_i}$ の乗算より (ただし，$GF(2^m)$ 上で演算を行う)，

$$\boldsymbol{X a_i} = \begin{pmatrix} w_{i,1} \\ w_{i,2} \\ \vdots \\ w_{i,n} \end{pmatrix} \tag{2.6}$$

となるような $w_{i,j}(j = 1, 2, \cdots, n)$ をシェアとよぶ．$w_{i,j}$ を分散する．

5. $G_c$ から各 $w_{c_p}$ のシェアが $k$ 個になるように集める．$k$ 個のシェアからシェア行列 $\boldsymbol{W_c}$

$$\boldsymbol{W_c} = \begin{pmatrix} w_{c_{1,1}} & w_{c_{2,1}} & \dots & w_{c_{e,1}} \\ w_{c_{1,2}} & w_{c_{2,2}} & \dots & w_{c_{e,2}} \\ \vdots & \vdots & \ddots & \vdots \\ w_{c_{1,k}} & w_{c_{2,k}} & \dots & w_{c_{e,k}} \end{pmatrix} \tag{2.7}$$

を作成する．

6. シェア行列 $\boldsymbol{W_c}$ と，紐付け情報 $\boldsymbol{u_c}$ を掛け合わせ部分復元シェア $\boldsymbol{W_c u_c}$ を作成する (ただし，$GF(2^m)$ 上で演算を行う).

$$
\begin{aligned}
\boldsymbol{W_c u_c} &= \begin{pmatrix} w_{c_{1,1}} & w_{c_{2,1}} & \cdots & w_{c_{e,1}} \\ w_{c_{1,2}} & w_{c_{2,2}} & \cdots & w_{c_{e,2}} \\ \vdots & \vdots & \ddots & \vdots \\ w_{c_{1,k}} & w_{c_{2,k}} & \cdots & w_{c_{e,k}} \end{pmatrix} \begin{pmatrix} \prod_{m=2}^{e} 2^{l_m} \\ \prod_{m=3}^{e} 2^{l_m} \\ \vdots \\ 2^{l_e} \\ 1 \end{pmatrix} \\
&= \begin{pmatrix} w_{c_{1,1}} \prod_{m=2}^{e} 2^{l_m} + w_{c_{2,1}} \prod_{m=3}^{e} 2^{l_m} + \cdots + w_{c_{e,1}} \\ w_{c_{1,2}} \prod_{m=2}^{e} 2^{l_m} + w_{c_{2,2}} \prod_{m=3}^{e} 2^{l_m} + \cdots + w_{c_{e,2}} \\ \vdots \\ w_{c_{1,k}} \prod_{m=2}^{e} 2^{l_m} + w_{c_{2,k}} \prod_{m=3}^{e} 2^{l_m} + \cdots + w_{c_{e,k}} \end{pmatrix}
\end{aligned} \tag{2.8}
$$

式 (2.8) を展開しまとめると，

$$
\boldsymbol{W_c u_c} = \boldsymbol{X} \begin{pmatrix} S_{c,1} \prod_{m=2}^{e} 2^{l_m} + S_{c_2} \prod_{m=3}^{e} 2^{l_m} + \cdots + S_{c_e} \\ r_{1,1} \prod_{m=2}^{e} 2^{l_m} + r_{2,1} \prod_{m=3}^{e} 2^{l_m} + \cdots + r_{e,1} \\ \vdots \\ r_{1,(k-1)} \prod_{m=2}^{e} 2^{l_m} + r_{2,(k-1)} \prod_{m=3}^{e} 2^{l_m} + \cdots + r_{e,(k-1)} \end{pmatrix} \tag{2.9}
$$

となる.

式 (2.9) に $X$ の逆行列 $X^{-1}$ を左からかけると (ただし，$GF(2^m)$ 上で演算を行う)

$$
\begin{aligned}
\boldsymbol{X^{-1} W_c u_c} &= S_{c_1} \prod_{m=2}^{e} 2^{l_m} + S_{c_2} \prod_{m=3}^{e} 2^{l_m} + \cdots + S_{c_e} \\
&= S'
\end{aligned} \tag{2.10}
$$

となり部分復元データ $S'$ を復元できる.

## 2.2　部分復元可能な秘密分散システムの復元にかかる演算量について

本節では，提案された部分復元アルゴリズムを用いた，部分復元可能な秘密分散システムの復元にかかる演算量について述べる．

### 2.2.1　復元にかかる演算量

提案された部分復元可能な秘密分散システムでは，分散したいデータ $S$ が表現できる $GF(2^m)$ で演算を行う．そのため，特に計算コストのかかる乗算演算に着目して復元にかかる演算量を求める．提案された部分復元可能な秘密分散システムの復元に必要な乗算回数を復元手順にそって示す．

1. $G_c$ から各 $w_{c_p}$ のシェアが $k$ 個になるように集める．$k$ 個のシェアからシェア行列 $\boldsymbol{W_c}$

$$\boldsymbol{W_c} = \begin{pmatrix} w_{c_{1,1}} & w_{c_{2,1}} & \cdots & w_{c_{e,1}} \\ w_{c_{1,2}} & w_{c_{2,2}} & \cdots & w_{c_{e,2}} \\ \vdots & \vdots & \ddots & \vdots \\ w_{c_{1,k}} & w_{c_{2,k}} & \cdots & w_{c_{e,k}} \end{pmatrix}$$

を作成する．

2. 集めたシェアに対応した $x_i(i = 1, \cdots, k)$ を用いて，$k * k$ の vandermonde 行列 $\boldsymbol{X}$

$$\boldsymbol{X} = \begin{pmatrix} 1 & x_1 & \cdots & x_1^{k-1} \\ 1 & x_2 & \cdots & x_2^{k-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_k & \cdots & x_k^{k-1} \end{pmatrix}$$

を作成する．vandermonde 行列 $\boldsymbol{X}$ の作成に必要な乗算回数 $\boldsymbol{V}$ は，$GF(2^m)$ 上で，

$$\boldsymbol{V} = \begin{cases} 0 & (k\text{<}3) \\ k\sum_{c=1}^{k-2} c & (otherwise) \end{cases}$$

回となる．

3. $k*k$ の $\boldsymbol{X}$ の逆行列 $\boldsymbol{X^{-1}}$ を掃き出し法を用いて作成する．逆行列 $\boldsymbol{X^{-1}}$ の作成に必要な乗算回数 $\boldsymbol{R}$ は $GF(2^m)$ 上で，

$$\boldsymbol{R} = k^3$$

回となる．

4. シェア行列 $\boldsymbol{W_c}$ と，紐付け情報 $\boldsymbol{u_c}$ を掛け合わせ部分復元シェア $\boldsymbol{W_c u_c}$

$$\begin{aligned}
\boldsymbol{W_c u_c} &= \begin{pmatrix} w_{c_{1,1}} & w_{c_{2,1}} & \cdots & w_{c_{e,1}} \\ w_{c_{1,2}} & w_{c_{2,2}} & \cdots & w_{c_{e,2}} \\ \vdots & \vdots & \ddots & \vdots \\ w_{c_{1,k}} & w_{c_{2,k}} & \cdots & w_{c_{e,k}} \end{pmatrix} \begin{pmatrix} \prod_{m=2}^{e} 2^{l_m} \\ \prod_{m=3}^{e} 2^{l_m} \\ \vdots \\ 2^{l_e} \\ 1 \end{pmatrix} \\
&= \begin{pmatrix} w_{c_{1,1}} \prod_{m=2}^{e} 2^{l_m} + w_{c_{2,1}} \prod_{m=3}^{e} 2^{l_m} + \cdots + w_{c_{e,1}} \\ w_{c_{1,2}} \prod_{m=2}^{e} 2^{l_m} + w_{c_{2,2}} \prod_{m=3}^{e} 2^{l_m} + \cdots + w_{c_{e,2}} \\ \vdots \\ w_{c_{1,k}} \prod_{m=2}^{e} 2^{l_m} + w_{c_{2,k}} \prod_{m=3}^{e} 2^{l_m} + \cdots + w_{c_{e,k}} \end{pmatrix} \\
&= \boldsymbol{X} \begin{pmatrix} S_{c_1} \prod_{m=2}^{e} 2^{l_m} + S_{c_2} \prod_{m=3}^{e} 2^{l_m} + \cdots + S_{c_e} \\ r_{1,1} \prod_{m=2}^{e} 2^{l_m} + r_{2,1} \prod_{m=3}^{e} 2^{l_m} + \cdots + r_{e,1} \\ \vdots \\ r_{1,(k-1)} \prod_{m=2}^{e} 2^{l_m} + r_{2,(k-1)} \prod_{m=3}^{e} 2^{l_m} + \cdots + r_{e,(k-1)} \end{pmatrix}
\end{aligned}$$

を作成する．部分復元用シェア $\boldsymbol{W_c u_c}$ の作成に必要な乗算回数 $\boldsymbol{L}$ は $GF(2^m)$ 上で，

$$\boldsymbol{L} = k * e$$

回となる．

5. $\boldsymbol{W_c u_c}$ に対して，$\boldsymbol{X}$ の逆行列 $\boldsymbol{X^{-1}}$ を左からかけると

$$\boldsymbol{X^{-1}W_c u_c} = \boldsymbol{X^{-1}X}\begin{pmatrix} S_{c_1}\prod_{m=2}^{e}2^{l_m} + S_{c_2}\prod_{m=3}^{e}2^{l_m} + \cdots + S_{c_e} \\ r_{1,1}\prod_{m=2}^{e}2^{l_m} + r_{2,1}\prod_{m=3}^{e}2^{l_m} + \cdots + r_{e,1} \\ \vdots \\ r_{1,(k-1)}\prod_{m=2}^{e}2^{l_m} + r_{2,(k-1)}\prod_{m=3}^{e}2^{l_m} + \cdots + r_{e,(k-1)} \end{pmatrix}$$

$$= \begin{pmatrix} S_{c_1}\prod_{m=2}^{e}2^{l_m} + S_{c_2}\prod_{m=3}^{e}2^{l_m} + \cdots + S_{c_e} \\ r_{1,1}\prod_{m=2}^{e}2^{l_m} + r_{2,1}\prod_{m=3}^{e}2^{l_m} + \cdots + r_{e,1} \\ \vdots \\ r_{1,(k-1)}\prod_{m=2}^{e}2^{l_m} + r_{2,(k-1)}\prod_{m=3}^{e}2^{l_m} + \cdots + r_{e,(k-1)} \end{pmatrix}$$

となり，復元すると

$$S_{c_1}\prod_{m=2}^{e}2^{l_m} + S_{c_2}\prod_{m=3}^{e}2^{l_m} + \cdots + S_{c_e} = S'$$

となり，$S$ の部分復元データ $S'$ を復元できる．

$\boldsymbol{X^{-1}W_c u_c}$ に必要な乗算回数 $\boldsymbol{M}$ は $GF(2^m)$ 上で，

$$\boldsymbol{M} = k^2$$

回となる．

以上の手順より復元にかかる合計乗算回数は，$GF(2^m)$ 上で，$\boldsymbol{V}+\boldsymbol{R}+\boldsymbol{L}+\boldsymbol{M}$ となり，すなわち，

$$\begin{cases} k^3 + k^2 + ke & (k\text{<}3) \\ k^3 + k^2 + k(\sum_{c=1}^{k-2} c + e) & (otherwise) \end{cases} \tag{2.11}$$

回の乗算が必要となる．

表 2.1　拡大次数 m の値による復元にかかる演算量

| データサイズ | 拡大次数 m | 乗算回数 | 復元にかかる演算量 |
|---|---|---|---|
| 2048Byte | 16,384 | $12+2e$ | $268,435,456(12+2e)$ |
| 3072Byte | 24,576 | $12+2e$ | $603,979,776(12+2e)$ |

1 回の乗算にかかる乗算の演算量を $m^2$ とすると，式 (2.11) より復元に必要な演算量は

$$\begin{cases} m^2(k^3+k^2+ke) & (k<3) \\ m^2(k^3+k^2+k(\sum_{c=1}^{k-2} c+e)) & (otherwise) \end{cases} \tag{2.12}$$

となる．

### 2.2.2　拡大次数 m の値による演算量の比較

分散したいデータ $S$ のサイズを 2048Byte,3072Byte として，部分復元アルゴリズムを用いて分散する．(k,n) のパラメータを (2,3)，分割数を $e$ とした場合の復元にかかる演算量を式 (2.12) を用いて，表 2.1 に示す．表 2.1 から，拡大次数 m の値が大きい場合，演算量が大きく復元まで何時間もかかってしまうとが予想できる．

# 第 3 章

# サイズによる分割を行った部分復元可能な秘密分散システム

部分復元可能な秘密分散システムが提案された．しかし，提案されたシステムは，演算量が大きく，復元速度が遅い．そこで，サイズによる分割を行った部分復元可能な秘密分散システムを提案する．本章では，まずサイズによる分割を提案する．そしてサイズによる分割を行った部分復元可能な秘密分散アルゴリズムと提案システムの復元にかかる演算量について述べる．

## 3.1 サイズによる分割

部分復元可能な秘密分散システムの復元速度を速くするために，サイズによる分割を提案する．

### 3.1.1 基本的な考え

提案された部分復元可能な秘密分散システムは分散したいデータ $S$ が表現できる $GF(2^m)$ 上で演算を行うため，2.2 で示したように 1 回の乗算に $m^2$ の演算量がかかる．そのため，演算量が大きく復元速度が遅い．拡大次数 $m$ の値を小さくすることができれば，1 回の乗算にかかる演算量を小さくし，復元速度を速くできるという考え方からサイズによる分割を提案する．具体的には，分割サイズを $F$ とすると，$GF(2^F)$ 上で部分復元を行えるという提案である．

### 3.1.2　分割方法

$GF(2^F)$ 上で部分復元できるようにするための分割方法について述べる．分散するデータ $S$ を一定のデータサイズ $F$ で $f$ 個に分け，分けたデータを $S_t(t = 1, 2, \cdots, f)$ とする．このように一定のデータサイズで分けることをサイズ分割とする．$S_t$ を秘匿部分を選択できるように $d$ 個に分け，$S_{t,i_t}(i_t = 1, 2, \cdots, d)$ を作成する．秘匿部分を選択できるように分けることを秘匿分割とする．$S_{t,i_t}$ のデータサイズを $l_{t,i_t}$ とすると，秘匿分割した $S_t$ は，

$$S_t = S_{t,1} \prod_{m=2}^{d} 2^{l_{t,m}} + S_{t,2} \prod_{m=3}^{d} 2^{l_{t,m}} + \cdots + S_{t,(d-1)} 2^{l_{t,d}} + S_{t,d} \tag{3.1}$$

と表され，サイズ分割したデータ $S$ は，

$$\begin{aligned} S =& S_1(2^F)^{f-1} + S_2(2^F)^{f-2} + \cdots + S_{f-1}(2^F) + S_f \\ =& S_{1,1}(2^F)^{f-1} \prod_{m=2}^{d} 2^{l_{1,m}} + S_{1,2}(2^F)^{f-1} \prod_{m=3}^{d} 2^{l_{1,m}} + \cdots \\ &+ S_{2,1}(2^F)^{f-2} \prod_{m=2}^{d} 2^{l_{2,m}} + \cdots + S_{f,(d-1)} 2^{l_{f,d}} + S_{f,d} \end{aligned} \tag{3.2}$$

と表すことができる．

このようにサイズによる分割をすることによって，$S_t$ を $GF(2^F)$ 上で復元できるようにしている．さらに，秘匿分割によって部分復元データ $S'_t$ を復元できるようにようにしている．$GF(2^F)$ 上で部分復元した $S'_t$ を用いて，データ $S$ の部分復元データ $S'$ を得ることができる．詳しい部分復元の手順は 3.2 で述べる．

## 3.2　サイズによる分割を行った部分復元アルゴリズム

本節では，サイズ分割を行った部分復元アルゴリズムについて述べる．提案するアルゴリズムは分散と復元で構成される．サイズによる分割を行った部分復元アルゴリズムの分散の流れを図 3.1 に復元の流れを図 3.2 に示す．

## 3.2　サイズによる分割を行った部分復元アルゴリズム

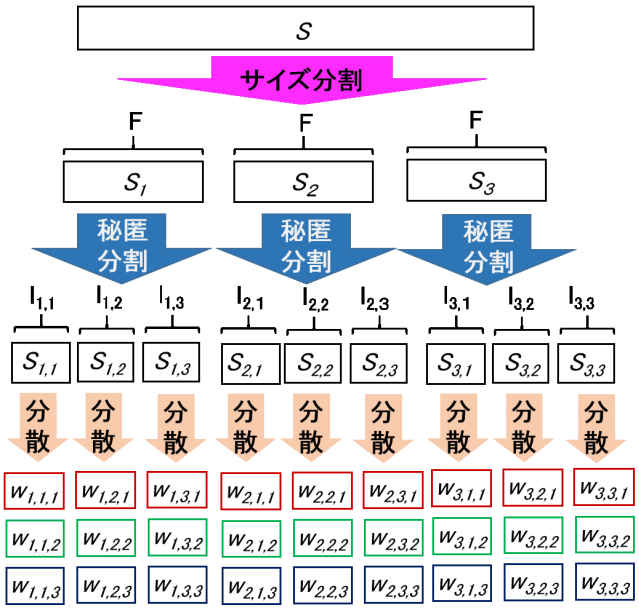


図 3.1　サイズによる分割を行った部分復元アルゴリズムの分散の流れ

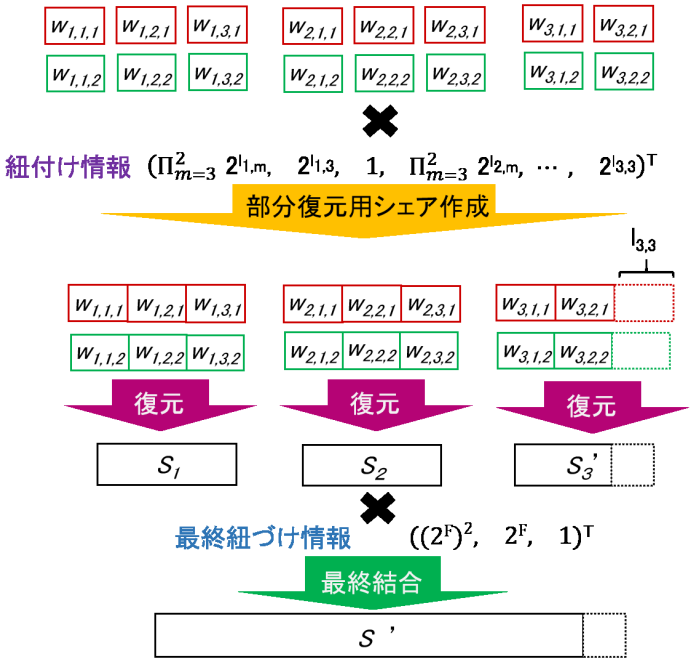


図 3.2　サイズによる分割を行った部分復元アルゴリズムの復元の流れ

### 3.2.1　分散

分散するデータ $S$ を一定のデータサイズ $F$ で $f$ 個にサイズ分割したデータを $S_t(t = 1, 2, \cdots, f)$ とする．$S_t$ を秘匿部分を選択できるように $d$ 個に秘匿分割し，$S_{t,i_t}(i_t = 1, 2, \cdots, d)$ を作成する．$S_{t,i_t}$ をそれぞれ $(k, n)$ しきい値秘密分散法を用いてシェアを作成し分散する．$S_{t,i_t}$ のシェア集合を $W_{t,i_t} = \{w_{t,i_t,1}, \cdots, w_{t,i_t,n}\}$ とする．秘匿分割し，生成したシェアを正しい順番に並んだデータとして復元できるように，正しく紐付ける情報を作成し紐付け情報とする．紐付け情報は秘匿分割したデータの長さ $l_{t,i_t}$ を要素として対応する順番に並べたものである．秘匿したいデータ内容にあたる部分を 0 となるように紐付け情報を作成する．また，サイズ分割によって作成した $S_t$ を正しい順番に並べるための情報を最終紐付け情報とする．

### 3.2.2　復元

復元したいシェアのみをしきい値以上集め，秘匿部分を選択した紐付け情報を入手する．集めたシェアと紐付け情報を用いて，部分復元用シェアを作成する．部分復元用シェアに復元操作を行うことで，$S_t$ の部分復元データ $S'_t$ を得ることができる．$S'_t$ と最終紐付け情報を用いて，$S$ の部分復元データ $S'$ を復元できる．

### 3.2.3　部分復元の手順

サイズ分割を行った部分復元アルゴリズムを述べる．秘密分散するデータを $S$，$S$ を一定のデータサイズ $F$ で $f$ 個にサイズ分割したものを $S_t(t = 1, 2, \cdots, f)$ とする．$S_t$ を秘匿分割したものを $S_{t,i_t}(i_t = 1, 2, \cdots, d)$，$S_{t,i_t}$ のデータサイズを $l_{t,i_t}$ とすると，秘匿分割した $S_t$ は，式 (3.1) のように表され，サイズ分割したデータ $S$ は，式 (3.2) のように表すことができる．

秘匿分割したデータ $S_{t,i_t}$ の集合を，

$$S_{all} = \{S_{1,1}, S_{1,2}, \cdots, S_{f,(d-1)}, S_{f,d}\}$$

とおき，復元したいデータの集合を $S_{\mathrm{all}}$ の部分集合として

$$S_{\mathrm{all}} \supseteqq S_c = \{S_{c_{1,1}}, S_{c_{1,2}}, \cdots, S_{c_{e,y_e}}\}(1 \leqq e \leqq f, 1 \leqq y_e \leqq d)$$

としたとき，復元したいデータ $S'$ は

$$S' = S_{c_{1,1}}(2^F)^{f-1}\prod_{m=2}^{y_1} 2^{l_{1,m}} + S_{c_{1,2}}(2^F)^{f-1}\prod_{m=3}^{y_1} 2^{l_{1,m}} + \cdots + S_{c_{e,y_e}} \tag{3.3}$$

と表すことができる．$S_c$ の任意の要素である $S_{c_{p,h_p}}$ $(1 \leqq p \leqq e, 1 \leqq h_p \leqq y_e)$ のシェア集合を

$$W_{c_{p,h_p}} = \{w_{c_{p,h_p,1}}, w_{c_{p,h_p,2}}, \cdots, w_{c_{p,h_p,n}}\}(1 \leqq p \leqq e, 1 \leqq h_p \leqq y_e)$$

とおく．復元したいデータのシェア集合 $W_{c_{p,h_p}}$ を集めたものをシェア集合 $G_c$

$$G_c = \{W_{c_{1,1}}, W_{c_{1,2}}, \cdots, W_{c_{e,y_e}}\}$$

とおき，シェア集合 $G_c$ の要素 $W_{c_{p,h_p}}$ を部分復元シェアとなるように計算するための紐付け情報 $\boldsymbol{u_c}$

$$\boldsymbol{u_c} = \begin{pmatrix} \prod_{m=2}^{y_1} 2^{l_{1,m}} \\ \prod_{m=3}^{y_1} 2^{l_{1,m}} \\ \vdots \\ 1 \\ \prod_{m=2}^{y_2} 2^{l_{2,m}} \\ \vdots \\ 2^{l_{e,y_e}} \\ 1 \end{pmatrix} \tag{3.4}$$

を作成する．また，復元した $S_t$ を紐付ける，最終紐付け情報 $\boldsymbol{b_c}$

$$\boldsymbol{b_c} = \begin{pmatrix} (2^F)^{e-1} \\ (2^F)^{e-2} \\ \vdots \\ (2^F) \\ 1 \end{pmatrix} \tag{3.5}$$

を作成する．部分復元の詳細な手順を以下に示す．

1. データ $S$ を一定のデータサイズ $F$ で $f$ 個に分割し，$S_t(t = 1, 2, \cdots, f)$ を作成する．
2. $S_t$ を式 (3.1) を用いて分割し，$S_{t,i_t}(i_t = 1, 2, \cdots, d)$ を作成する．$S_{t,i_t}$ のデータサイズを $l_{t,i_t}$ とする．
3. $GF(2^F)$ の元 $-\{0\}$ の集合 $X = \{x_1, x_2, \cdots, x_n\}$ から $n * k$ の vandermond 行列 $\boldsymbol{X}$ を作成する (ただし，$GF(2^F)$ 上で演算を行う)．

$$\boldsymbol{X} = \begin{pmatrix} 1 & x_1 & \dots & x_1^{k-1} \\ 1 & x_2 & \dots & x_2^{k-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & \dots & x_n^{k-1} \end{pmatrix} \tag{3.6}$$

また，分割データ $S_{t,i_t}$ と $GF(2^F)$ の元の集合 $R = \{r_{t,i_t,1}, r_{t,i_t,2}, \cdots, r_{t,i_t,k-1}\}$ からランダムに選択し，ベクトル $\boldsymbol{a_{t,i_t}}$ を作成する．

$$\boldsymbol{a_{t,i_t}} = \begin{pmatrix} S_{t,i_t} \\ r_{t,i_t,1} \\ \vdots \\ r_{t,i_t,k-1} \end{pmatrix} \tag{3.7}$$

4. $\boldsymbol{X}$ と $\boldsymbol{a_{t,i_t}}$ の乗算より (ただし，$GF(2^F)$ 上で演算を行う)，

$$\boldsymbol{X a_{t,i_t}} = \begin{pmatrix} w_{t,i_t,1} \\ w_{t,i_t,2} \\ \vdots \\ w_{t,i_t,n} \end{pmatrix} \tag{3.8}$$

となるような $w_{t,i_t,j}(j = 1, 2, \cdots, n)$ をシェアとよぶ．$w_{t,i_t,j}$ を分散する．

5. $G_c$ から各 $w_{c_{p,h_p}}$ のシェアが $k$ 個になるように集める．$k$ 個のシェアからシェア行列

$\boldsymbol{W_c}$

$$\boldsymbol{W_c} = \begin{pmatrix} w_{c_{1,1,1}} & w_{c_{1,2,1}} & \cdots & w_{c_{e,y_e,1}} \\ w_{c_{1,1,2}} & w_{c_{1,2,2}} & \cdots & w_{c_{e,y_e,2}} \\ \vdots & \vdots & \ddots & \vdots \\ w_{c_{1,1,k}} & w_{c_{1,2,k}} & \cdots & w_{c_{e,y_e,k}} \end{pmatrix} \tag{3.9}$$

を作成する.

6. シェア行列 $\boldsymbol{W_c}$ と，紐付け情報 $\boldsymbol{u_c}$ を掛け合わせ部分復元シェア $\boldsymbol{W_c u_c}$ を作成する (ただし，$GF(2^F)$ 上で演算を行う).

$$\begin{aligned} \boldsymbol{W_c u_c} &= \begin{pmatrix} w_{c_{1,1,1}} & w_{c_{1,2,1}} & \cdots & w_{c_{e,y_e,1}} \\ w_{c_{1,1,2}} & w_{c_{1,2,2}} & \cdots & w_{c_{e,y_e,2}} \\ \vdots & \vdots & \ddots & \vdots \\ w_{c_{1,1,k}} & w_{c_{1,2,k}} & \cdots & w_{c_{e,y_e,k}} \end{pmatrix} \begin{pmatrix} \prod_{m=2}^{y_1} 2^{l_{1,m}} \\ \prod_{m=3}^{y_1} 2^{l_{1,m}} \\ \vdots \\ 1 \\ \prod_{m=2}^{y_2} 2^{l_{2,m}} \\ \vdots \\ 2^{l_{e,y_e}} \\ 1 \end{pmatrix} \\ &= \begin{pmatrix} w_{c_{1,1,1}} \prod_{m=2}^{y_1} 2^{l_{1,m}} + w_{c_{1,2,1}} \prod_{m=3}^{y_1} 2^{l_{1,m}} + \cdots + w_{c_{e,y_e,1}} \\ w_{c_{1,1,2}} \prod_{m=2}^{y_1} 2^{l_{1,m}} + w_{c_{1,2,2}} \prod_{m=3}^{y_1} 2^{l_{1,m}} + \cdots + w_{c_{e,y_e,2}} \\ \vdots \\ w_{c_{1,1,k}} \prod_{m=2}^{y_1} 2^{l_{1,m}} + w_{c_{1,2,k}} \prod_{m=3}^{y_1} 2^{l_{1,m}} + \cdots + w_{c_{e,y_e,k}} \end{pmatrix} \end{aligned} \tag{3.10}$$

式 (3.10) を各 $S_t$ ごとのシェアを含む項となるように展開すると，$e$ 項の $(k * 1)$ 行列の

和となり，式 (3.11) のようになる．

$$\boldsymbol{W_c u_c} = \begin{pmatrix} w_{c_{1,1,1}} \prod_{m=2}^{y_1} 2^{l_{1,m}} + w_{c_{1,2,1}} \prod_{m=3}^{y_1} 2^{l_{1,m}} + \cdots + w_{c_{1,y_1,1}} \\ w_{c_{1,1,2}} \prod_{m=2}^{y_1} 2^{l_{1,m}} + w_{c_{1,2,2}} \prod_{m=3}^{y_1} 2^{l_{1,m}} + \cdots + w_{c_{1,y_1,2}} \\ \vdots \\ w_{c_{1,1,k}} \prod_{m=2}^{y_1} 2^{l_{1,m}} + w_{c_{1,2,k}} \prod_{m=3}^{y_1} 2^{l_{1,m}} + \cdots + w_{c_{1,y_1,k}} \end{pmatrix} + \cdots + \begin{pmatrix} w_{c_{e,1,1}} \prod_{m=2}^{y_e} 2^{l_{e,m}} + \ldots + w_{c_{e,y_e,1}} \\ w_{c_{e,1,2}} \prod_{m=2}^{y_e} 2^{l_{e,m}} + \ldots + w_{c_{e,y_e,2}} \\ \vdots \\ w_{c_{e,1,k}} \prod_{m=2}^{y_e} 2^{l_{e,m}} + \ldots + w_{c_{e,y_e,k}} \end{pmatrix} \tag{3.11}$$

式 (3.11) の各項を，$\boldsymbol{X}$ をくくりだし展開すると，

$$\boldsymbol{W_c u_c} = \boldsymbol{X} \begin{pmatrix} S_{c_{1,1}} \prod_{m=2}^{y_1} 2^{l_{1,m}} + S_{c_{1,2}} \prod_{m=3}^{y_1} 2^{l_{1,m}} + \cdots + S_{c_{1,y_1}} \\ r_{1,1,1} \prod_{m=2}^{y_1} 2^{l_{1,m}} + r_{1,2,1} \prod_{m=3}^{y_1} 2^{l_{1,m}} + \cdots + r_{1,y_1,1} \\ \vdots \\ r_{1,1,(k-1)} \prod_{m=2}^{y_1} 2^{l_{1,m}} + r_{1,2,(k-1)} \prod_{m=3}^{y_1} 2^{l_{1,m}} + \cdots + r_{1,y_1,(k-1)} \end{pmatrix} + \cdots + \boldsymbol{X} \begin{pmatrix} S_{c_{e,1}} \prod_{m=2}^{y_e} 2^{l_{e,m}} + \ldots + S_{c_{e,y_e}} \\ r_{e,1,1} \prod_{m=2}^{y_e} 2^{l_{e,m}} + \ldots + r_{e,y_e,1} \\ \vdots \\ r_{e,1,(k-1)} \prod_{m=2}^{y_e} 2^{l_{e,m}} + \ldots + r_{e,y_e,(k-1)} \end{pmatrix} \tag{3.12}$$

となる．

7. 式 (3.12) の各項を取り出すと，$e$ 個の $(k * 1)$ 行列を取り出すことができる．取り出し

た行列をそれぞれ，$\boldsymbol{I}_p(p=1,\cdots,e)$ とおく．

$$\boldsymbol{I}_1 = \boldsymbol{X}\begin{pmatrix} S_{c_{1,1}}\prod_{m=2}^{y_1} 2^{l_{1,m}} + S_{c_{1,2}}\prod_{m=3}^{y_1} 2^{l_{1,m}} + \cdots + S_{c_{1,y_1}} \\ r_{1,1,1}\prod_{m=2}^{y_1} 2^{l_{1,m}} + r_{1,2,1}\prod_{m=3}^{y_1} 2^{l_{1,m}} + \cdots + r_{1,y_1,1} \\ \vdots \\ r_{1,1,(k-1)}\prod_{m=2}^{y_1} 2^{l_{1,m}} + r_{1,2,(k-1)}\prod_{m=3}^{y_1} 2^{l_{1,m}} + \cdots + r_{1,y_1,(k-1)} \end{pmatrix}$$

$$\vdots \tag{3.13}$$

$$\boldsymbol{I}_e = \boldsymbol{X}\begin{pmatrix} S_{c_{e,1}}\prod_{m=2}^{y_e} 2^{l_{e,m}} + \ldots + S_{c_{e,y_e}} \\ r_{e,1,1}\prod_{m=2}^{y_e} 2^{l_{e,m}} + \ldots + r_{e,y_e,1} \\ \vdots \\ r_{e,1,(k-1)}\prod_{m=2}^{y_e} 2^{l_{e,m}} + \ldots + r_{e,y_e,(k-1)} \end{pmatrix}$$

8. $\boldsymbol{I}_p$ に対して，$\boldsymbol{X}$ の逆行列 $\boldsymbol{X^{-1}}$ を左からかけると (ただし，$GF(2^F)$ 上で演算を行う)，

$$\boldsymbol{X^{-1}I}_p = \boldsymbol{X^{-1}X}\begin{pmatrix} S_{c_{p,1}}\prod_{m=2}^{y_p} 2^{l_{p,m}} + \ldots + S_{c_{p,y_p}} \\ r_{p,1,1}\prod_{m=2}^{y_p} 2^{l_{p,m}} + \ldots + r_{p,y_p,1} \\ \vdots \\ r_{p,1,(k-1)}\prod_{m=2}^{y_p} 2^{l_{p,m}} + \ldots + r_{p,y_p,(k-1)} \end{pmatrix}$$

$$= \begin{pmatrix} S_{c_{p,1}}\prod_{m=2}^{y_p} 2^{l_{p,m}} + \ldots + S_{c_{p,y_p}} \\ r_{p,1,1}\prod_{m=2}^{y_p} 2^{l_{p,m}} + \ldots + r_{p,y_p,1} \\ \vdots \\ r_{p,1,(k-1)}\prod_{m=2}^{y_p} 2^{l_{p,m}} + \ldots + r_{p,y_p,(k-1)} \end{pmatrix}$$

となり，復元すると

$$S_{c_{p,1}} \prod_{m=2}^{y_p} 2^{l_{p,m}} + \ldots + S_{c_{p,y_p}} = S'_p \tag{3.14}$$

となり，$S_p$ の部分復元データ $S'_p$ を復元できる．

9. $S'_p$ をそれぞれ集め，$1 * e$ の行列 $\boldsymbol{C}$

$$\boldsymbol{C} = \begin{pmatrix} S'_1 & S'_2 & \cdots & S'_e \end{pmatrix} \tag{3.15}$$

を作成する．

10. $\boldsymbol{C}$ に対して，最終紐付け情報 $\boldsymbol{b_c}$ を掛け合わせると，

$$\begin{aligned} \boldsymbol{C}\boldsymbol{b_c} &= \begin{pmatrix} S'_1 & S'_2 & \cdots & S'_e \end{pmatrix} \begin{pmatrix} (2^F)^{e-1} \\ (2^F)^{e-2} \\ \vdots \\ 1 \end{pmatrix} \\ &= S'_1(2^F)^{e-1} + S'_2(2^F)^{e-2} + \cdots + S'_e \\ &= S_{c_{1,1}}(2^F)^{e-1} \prod_{m=2}^{y_1} 2^{l_{1,m}} + S_{c_{1,2}}(2^F)^{e-1} \prod_{m=3}^{y_1} 2^{l_{1,m}} + \cdots \\ &\quad + S_{c_{2,1}}(2^F)^{e-2} \prod_{m=2}^{y_2} 2^{l_{2,m}} + \cdots + S_{c_{e,y_e}} \\ &= S' \end{aligned} \tag{3.16}$$

となり，$S$ の部分的なデータ $S'$ を復元できる．

## 3.3　サイズによる分割を行った部分復元可能な秘密分散システムの復元にかかる演算量について

本節では，サイズによる分割を行った部分復元可能な秘密分散システムの復元にかかる演算量を示す．そして，分割サイズによる復元にかかる演算量の比較を行う．

### 3.3.1　復元にかかる演算量

サイズによる分割を行った部分復元可能な秘密分散システムでは，分散したいデータ $S$ が表現できる $GF(2^m)$ ではなく，分割サイズ $F$ に依存した $GF(2^F)$ 上で演算を行う．特に計算コストのかかる乗算演算に着目して復元にかかる演算量を求める．サイズ分割を行った部分復元可能な秘密分散システムの復元に必要な乗算回数を復元手順にそって示す．

1. $G_c$ から各 $w_{c_{p,h_p}}$ のシェアが $k$ 個になるように集める．$k$ 個のシェアからシェア行列 $\boldsymbol{W_c}$

$$\boldsymbol{W_c} = \begin{pmatrix} w_{c_{1,1,1}} & w_{c_{1,2,1}} & \cdots & w_{c_{e,y_e,1}} \\ w_{c_{1,1,2}} & w_{c_{1,2,2}} & \cdots & w_{c_{e,y_e,2}} \\ \vdots & \vdots & \ddots & \vdots \\ w_{c_{1,1,k}} & w_{c_{1,2,k}} & \cdots & w_{c_{e,y_e,k}} \end{pmatrix}$$

   を作成する．

2. 集めたシェアに対応した $x_i(i = 1, \cdots, k)$ を用いて，$k * k$ の vandermonde 行列 $\boldsymbol{X}$ を作成する．

$$\boldsymbol{X} = \begin{pmatrix} 1 & x_1 & \cdots & x_1^{k-1} \\ 1 & x_2 & \cdots & x_2^{k-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_k & \cdots & x_k^{k-1} \end{pmatrix}$$

   vandermonde 行列 $\boldsymbol{X}$ の作成に必要な乗算回数 $\boldsymbol{V}$ は，$GF(2^F)$ 上で，

$$\boldsymbol{V} = \begin{cases} 0 & (k\text{<}3) \\ k\sum_{c=1}^{k-2} c & (otherwise) \end{cases}$$

   回となる．

3. $k * k$ の $\boldsymbol{X}$ の逆行列 $\boldsymbol{X^{-1}}$ を掃き出し法を用いて作成する．逆行列 $\boldsymbol{X^{-1}}$ の作成に必要な乗算回数 $\boldsymbol{R}$ は $GF(2^F)$ 上で，

$$\boldsymbol{R} = k^3$$

   回となる．

4. シェア行列 $\boldsymbol{W_c}$ と，紐付け情報 $\boldsymbol{u_c}$ を掛け合わせ部分復元シェア $\boldsymbol{W_c u_c}$ を作成する．

$$
\begin{aligned}
\boldsymbol{W_c u_c} &= \begin{pmatrix} w_{c_{1,1,1}} & w_{c_{1,2,1}} & \cdots & w_{c_{e,y_e,1}} \\ w_{c_{1,1,2}} & w_{c_{1,2,2}} & \cdots & w_{c_{e,y_e,2}} \\ \vdots & \vdots & \ddots & \vdots \\ w_{c_{1,1,k}} & w_{c_{1,2,k}} & \cdots & w_{c_{e,y_e,k}} \end{pmatrix} \begin{pmatrix} \prod_{m=2}^{y_1} 2^{l_{1,m}} \\ \prod_{m=3}^{y_1} 2^{l_{1,m}} \\ \vdots \\ 1 \\ \prod_{m=2}^{y_2} 2^{l_{2,m}} \\ \vdots \\ 2^{l_{e,y_e}} \\ 1 \end{pmatrix} \\
&= \begin{pmatrix} w_{c_{1,1,1}} \prod_{m=2}^{y_1} 2^{l_{1,m}} + w_{c_{1,2,1}} \prod_{m=3}^{y_1} 2^{l_{1,m}} + \cdots + w_{c_{e,y_e,1}} \\ w_{c_{1,1,2}} \prod_{m=2}^{y_1} 2^{l_{1,m}} + w_{c_{1,2,2}} \prod_{m=3}^{y_1} 2^{l_{1,m}} + \cdots + w_{c_{e,y_e,2}} \\ \vdots \\ w_{c_{1,1,k}} \prod_{m=2}^{y_1} 2^{l_{1,m}} + w_{c_{1,2,k}} \prod_{m=3}^{y_1} 2^{l_{1,m}} + \cdots + w_{c_{e,y_e,k}} \end{pmatrix} \\
&= \begin{pmatrix} w_{c_{1,1,1}} \prod_{m=2}^{y_1} 2^{l_{1,m}} + w_{c_{1,2,1}} \prod_{m=3}^{y_1} 2^{l_{1,m}} + \cdots + w_{c_{1,y_1,1}} \\ w_{c_{1,1,2}} \prod_{m=2}^{y_1} 2^{l_{1,m}} + w_{c_{1,2,2}} \prod_{m=3}^{y_1} 2^{l_{1,m}} + \cdots + w_{c_{1,y_1,2}} \\ \vdots \\ w_{c_{1,1,k}} \prod_{m=2}^{y_1} 2^{l_{1,m}} + w_{c_{1,2,k}} \prod_{m=3}^{y_1} 2^{l_{1,m}} + \cdots + w_{c_{1,y_1,k}} \end{pmatrix} + \\
&\cdots + \begin{pmatrix} w_{c_{e,1,1}} \prod_{m=2}^{y_e} 2^{l_{e,m}} + \ldots + w_{c_{e,y_e,1}} \\ w_{c_{e,1,2}} \prod_{m=2}^{y_e} 2^{l_{e,m}} + \ldots + w_{c_{e,y_e,2}} \\ \vdots \\ w_{c_{e,1,k}} \prod_{m=2}^{y_e} 2^{l_{e,m}} + \ldots + w_{c_{e,y_e,k}} \end{pmatrix}
\end{aligned}
$$

$$
\begin{aligned}
=\boldsymbol{X}&\begin{pmatrix}
S_{c_{1,1}}\prod_{m=2}^{y_1}2^{l_{1,m}}+S_{c_{1,2}}\prod_{m=3}^{y_1}2^{l_{1,m}}+\cdots+S_{c_{1,y_1}}\\
r_{1,1,1}\prod_{m=2}^{y_1}2^{l_{1,m}}+r_{1,2,1}\prod_{m=3}^{y_1}2^{l_{1,m}}+\cdots+r_{1,y_1,1}\\
\vdots\\
r_{1,1,(k-1)}\prod_{m=2}^{y_1}2^{l_{1,m}}+r_{1,2,(k-1)}\prod_{m=3}^{y_1}2^{l_{1,m}}+\cdots+r_{1,y_1,(k-1)}
\end{pmatrix}+\\
\cdots+\boldsymbol{X}&\begin{pmatrix}
S_{c_{e,1}}\prod_{m=2}^{y_e}2^{l_{e,m}}+\ldots+S_{c_{e,y_e}}\\
r_{e,1,1}\prod_{m=2}^{y_e}2^{l_{e,m}}+\ldots+r_{e,y_e,1}\\
\vdots\\
r_{e,1,(k-1)}\prod_{m=2}^{y_e}2^{l_{e,m}}+\ldots+r_{e,y_e,(k-1)}
\end{pmatrix}
\end{aligned}
$$

部分復元用シェア $\boldsymbol{W_c u_c}$ の作成に必要な乗算回数 $\boldsymbol{L}$ は $GF(2^F)$ 上で,

$$\boldsymbol{L}=k*e*y_e$$

回となる.

5. $\boldsymbol{I}_p(p=1,\cdots,e)$ に対して，$\boldsymbol{X}$ の逆行列 $\boldsymbol{X^{-1}}$ を左からかけると

$$\boldsymbol{X^{-1}I}_p = \boldsymbol{X^{-1}X}\begin{pmatrix} S_{c_{p,1}}\prod_{m=2}^{y_p}2^{l_{p,m}}+\ldots+S_{c_{p,y_p}} \\ r_{p,1,1}\prod_{m=2}^{y_p}2^{l_{p,m}}+\ldots+r_{p,y_p,1} \\ \vdots \\ r_{p,1,(k-1)}\prod_{m=2}^{y_p}2^{l_{p,m}}+\ldots+r_{p,y_p,(k-1)} \end{pmatrix}$$

$$= \begin{pmatrix} S_{c_{p,1}}\prod_{m=2}^{y_p}2^{l_{p,m}}+\ldots+S_{c_{p,y_p}} \\ r_{p,1,1}\prod_{m=2}^{y_p}2^{l_{p,m}}+\ldots+r_{p,y_p,1} \\ \vdots \\ r_{p,1,(k-1)}\prod_{m=2}^{y_p}2^{l_{p,m}}+\ldots+r_{p,y_p,(k-1)} \end{pmatrix}$$

となり，復元すると

$$S_{c_{p,1}}\prod_{m=2}^{y_p}2^{l_{p,m}}+\ldots+S_{c_{p,y_p}} = S'_p$$

となり，$S_p$ の部分復元データ $S'_p$ を復元できる．

$\boldsymbol{X^{-1}I}_p$ に必要な乗算回数 $\boldsymbol{M}$ は $GF(2^F)$ 上で，

$$\boldsymbol{M} = e * k^2$$

回となる．

6. $\boldsymbol{C}$ に対して，最終紐付け情報 $\boldsymbol{b_c}$ を掛け合わせると，

$$
\begin{aligned}
\boldsymbol{Cb_c} &= \begin{pmatrix} S'_1 & S'_2 & \cdots & S'_e \end{pmatrix} \begin{pmatrix} (2^F)^{e-1} \\ (2^F)^{e-2} \\ \vdots \\ 1 \end{pmatrix} \\
&= S'_1(2^F)^{e-1} + S'_2(2^F)^{e-2} + \cdots + S'_e \\
&= S_{c_{1,1}}(2^F)^{e-1} \prod_{m=2}^{y_1} 2^{l_{1,m}} + S_{c_{1,2}}(2^F)^{e-1} \prod_{m=3}^{y_1} 2^{l_{1,m}} + \cdots \\
&\quad + S_{c_{2,1}}(2^F)^{e-2} \prod_{m=2}^{y_2} 2^{l_{2,m}} + \cdots + S_{c_{e,y_e}} \\
&= S'
\end{aligned}
$$

となり，部分復元データ $S'$ を復元できる．

以上の手順より復元にかかる合計乗算回数は，$GF(2^F)$ 上で，$\boldsymbol{V} + \boldsymbol{R} + \boldsymbol{L} + \boldsymbol{M}$ となり，すなわち，

$$
\begin{cases}
k^3 + k^2e + key_e & (k\text{<}3) \\
k^3 + k^2e + k(\sum_{c=1}^{k-2} c + ey_e) & (otherwise)
\end{cases}
\tag{3.17}
$$

回の乗算が必要となる．

1 回の乗算にかかる乗算の演算量を $F^2$ とすると，式 (3.17) より復元に必要な演算量は

$$
\begin{cases}
F^2(k^3 + k^2e + key_e) & (k\text{<}3) \\
F^2(k^3 + k^2e + k(\sum_{c=1}^{k-2} c + ey_e) & (otherwise)
\end{cases}
\tag{3.18}
$$

となる．

### 3.3.2　分割サイズによる復元にかかる演算量の比較

分散したいデータ $S$ のサイズを 2048Byte としてサイズ分割を行った部分復元アルゴリズムを用いて分散する．復元にかかる演算量に実際の値を入れてみた結果を式 (3.18) を用

3.3　サイズによる分割を行った部分復元可能な秘密分散システムの復元にかかる演算量について

表 3.1　2048B のデータの復元にかかる演算量

| 分割サイズ | 秘匿単位 | 一回の乗算にかかる演算量 | 乗算回数 | 復元にかかる演算量 |
|---|---|---|---|---|
| 8bit | 4bit | $8^2$ | 16,392 | 1,049,088 |
| 16bit | 8bit | $16^2$ | 8,200 | 2,099,200 |
| 32bit | 8bit | $32^2$ | 6,152 | 6,299,648 |
| 64bit | 8bit | $64^2$ | 5,128 | 21,004,288 |

いて，表 3.1 に表す. 表 3.1 より，分割サイズが小さければ小さいほど，復元にかかる演算量が小さいことが分かる.

# 第 4 章

# 提案システムの実装

提案したサイズによる分割を行った部分復元可能な秘密分散システムを実装した．本章では実装したシステムの仕様と設計について述べる．

## 4.1　仕様

本節では，提案システムを実装する際に立てた方針と仕様について説明する．

### 4.1.1　基本方針

今回の実装は，部分復元可能な秘密分散システムの分割サイズによる復元速度の評価を行うことを目的とした． 使用するサイズ分割を行った部分復元アルゴリズムは，3 章で提案したものである．

### 4.1.2　仕様

提案システムは分散と復元で構成されているので，それぞれの動作を以下に示す．

#### 分散

分散したいデータ $S$ が記述されたファイルとパラメータに従い，シェアファイルを出力する．

復元

シェアや紐付け情報が記述されたファイルとパラメータに従い，部分復元データ $S'$ を復元し標準出力する．

## 4.2 設計

提案システムの機能を分散と復元に分けて設計した．分散と復元に必要なパラメータはプログラムに記述した．また，シェアや紐付け情報はファイルに保存するように設計した．

### 4.2.1 パラメータ

$(k, n)$ しきい値，分割数 $f$，分割サイズ $F$ などのパラメータはプログラムに記述しておく．

### 4.2.2 シェアファイル

シェアファイルには 3 章で述べたアルゴリズムで作成したシェア情報の他に，シェアに対応した $x_i$ と $i$ を記述している．

### 4.2.3 紐付け情報ファイルと最終紐付け情報ファイル

紐付け情報 $\boldsymbol{u_c}$ と，最終紐付け情報 $\boldsymbol{b_c}$ をそれぞれ 1 つのファイルに記述している．

### 4.2.4 分散の流れ

実装したシステムの分散の流れを示す．

1. ファイルから秘匿単位で分散したいデータ $S$ を読み込む．
2. $x_i$ を用いて，vandermonde 行列 $X$ を作成する．
3. 乱数行列 $R$ を関数によって作成する．
4. 式 (3.8) の行列計算によりシェアデータ $w_{t,i}$ を求める．

5. シェアファイルにシェアデータ $w_{t,i}$，$x_i$，$i$ を書き込む．

### 4.2.5 復元の流れ

実装したシステムの復元の流れを示す．

1. 紐付け情報ファイルから紐付け情報 $\boldsymbol{u_c}$ を読み込む．
2. 最終紐付け情報ファイルから最終紐付け情報 $\boldsymbol{b_c}$ を読み込む．
3. シェアファイルからシェアデータ $w_{t,i}$，$x_i$，$i$ を読み込む．
4. $x_i$ を用いて，vandermonde 行列 $X$ を作成する．
5. 逆行列 $X^{-1}$ を作成する．
6. シェア行列 $\boldsymbol{W_c}$ に紐付け情報 $\boldsymbol{u_c}$ を掛け，行列 $\boldsymbol{I}_p$ を取り出す．
7. $\boldsymbol{I}_p$ に逆行列 $X^{-1}$ を掛け合わせ，$\boldsymbol{S}'_p$ を復元する．
8. $\boldsymbol{S}'_p$ の集合から $\boldsymbol{C}$ を作成し，$\boldsymbol{C}$ に最終紐付け情報 $\boldsymbol{b_c}$ を掛け合わせ $S'$ を復元する．
9. 部分復元データ $S'$ を標準出力する．

# 第 5 章

# 分割サイズによる復元速度の評価

本章では本研究で実装したサイズ分割を行った部分復元可能な秘密分散システムの実行結果について述べる．そして，考察を行い，分割サイズによる復元速度の評価を述べる．

## 5.1 評価実験

測定に用いたデータは，2048Byte であり，(2,3) しきい値秘密分散を用いて分散した．分割サイズをかえ，復元にかかる時間を計測した．実行時間は計りたい部分の最初と終わりに *clock*() 関数を用いて時間経過を取得し，その差分を計算することにより得た．

### 5.1.1 測定環境

測定に用いた環境を次に示す．

**OS** freebsd 11.1

**CPU** Intel(R) Core(TM) i5-4460 3.20GHz

メモリ 4GB

**HDD** 500GB

多倍長演算ライブラリ GMP-6.1.2

**Compiler** gcc version 5.4.0

コンパイラオプション -O

表 5.1　2048B のデータの復元にかかった時間 (s)

| 分割サイズ | 秘匿単位 | シェア数 | シェア読み込み時間 | 復元演算時間 | 復元時間 |
|---|---|---|---|---|---|
| 8bit | 4bit | 8192 | 0.1148435 | 0.0242191 | 0.1390624 |
| 16bit | 8bit | 4096 | 0.0554687 | 0.0195315 | 0.0749998 |
| 32bit | 8bit | 4096 | 0.0585939 | 0.0250003 | 0.0835935 |
| 64bit | 8bit | 4096 | 0.0554687 | 0.0515627 | 0.1070312 |

### 5.1.2　結果

分割サイズを 8bit,16bit,32bit,64bit として計測した．復元にかかった時間を表 5.1 に表す．シェア読み込み時間は，4.2.5 の処理 3 にかかった時間である．復元演算時間は，4.2.5 の処理 4 から処理 9 にかかった時間である．復元時間は 4.2.5 の処理 1 から処理 9 にかかった時間である．

### 5.1.3　考察

今回のプログラムでは，表 3.1 に示した理論値どおり，分割サイズ 64bit,32bit,16bi と小さくすることで復元演算時間を小さくできることを確認できた．しかし，分割サイズが 8bit の場合，理論値とは異なり，復元演算時間は分割サイズ 16bit より大きい結果となった．これは，今回作成した拡大体上での乗算プログラムが原因であり，乗算一回あたりの実行時間が表 3.1 とは異なったからであると考えられる．

分割サイズにしたがって秘匿単位を小さくすると，復元に必要なシェアファイル数が増加しシェア読み込み時間が増えていることが確認できる．シェア読み込み時間が復元演算時間より大きいため，分割サイズを小さくしすぎると復元時間が大きくなることが予想できる．

## 5.2　分割サイズによる復元速度の評価

復元したいデータサイズが大きい場合でも，分割サイズを小さくすれば，十分高速に復元できると考えられる．また，復元に必要な行列演算を並列処理で実行したり，クロック周波数の高い CPU を用いることでより高速に復元できると考えられる．また，シェアファイルの読み込み時間も復元時間として無視できないため，SSD などを使用すれば，復元速度を上げることが期待できる．

# 第 6 章

# まとめ

## 6.1 本研究のまとめ

本研究では部分復元可能な秘密分散システムの復元時間を速くすることを目的として，サイズによる分割を行った部分復元可能な秘密分散システムを提案した．そして，提案システムの実装と分割サイズによる復元速度の評価を行った．

提案方法では，データを一定のサイズ $F$ で分割し，サイズ分割したデータを秘匿部分を選択できるように秘匿分割し，それぞれの秘匿分割データを同じ $(k, n)$ しきい値秘密分散法で分散した．復元時に復元したいデータのシェアのみを $k$ 個以上集め，紐付け情報と最終紐付け情報を用いて部分復元を行った．この結果，復元処理を $GF(2^F)$ 上で行えるようになり，分割サイズ $F$ を小さくするに従って復元にかかる演算量を小さくできた．しかし，分割サイズを小さくしすぎると，復元に必要なシェアファイル数が増えてしまい，シェアファイル読み込み処理が遅くなってしまう．そのため，分割サイズとシェアファイル数のバランスが大切であるといえる．

分割サイズによる復元速度の評価を行った結果，分割サイズを小さくすれば，並列処理の実装や CPU の性能によって，復元したいデータサイズが大きい場合でも十分高速に部分復元できると推測できた．また，シェアファイルの読み込み時間も復元時間として無視できないため，SSD などを使用すれば復元速度を上げることが期待できる．

## 6.2　今後の課題

今後の課題として，部分復元システムの実用化に向けて，ネットワーク上のストレージからシェアを入手するなど，より詳細に復元速度を評価する必要がある．

# 謝辞

本研究を行うにあたり，御指導を頂いた福本昌弘教授に謹んで感謝致します．計画性がなく，考えを持たないまま行動する私に対して最後までご指導していただきました．研究や就職活動は，先生のお声がけで，乗りきれたと思っています．週次報告などで，少しでも研究が進んだことを実感できた時はとてもうれしい気持ちでした．

本研究の副査をしていただいた情報学群横山和俊教授，鵜川始陽准教授のお二人にも謹んで感謝致します．また，本研究で用いたデータの提供をいただいた高知県医療センターの情報システム室北村和之氏に謹んで感謝致します．

NOC の職員であり研究室の OB でもある福冨英次氏にも謹んで感謝致します．研究のアドバイスをいただき，お食事や遊びにも誘っていただきました．研究に関する質問に対して，十分な返答ができずご迷惑をかけました．

Bandhit.Suksiri 氏にも謹んで感謝致します．私がつらそうに作業していることを気にかけてくださり，優しく笑顔でお声がけていただきました．

同期の佐藤文也氏，沼尚樹氏にも謹んで感謝致します．研究室で泊まり込みで作業したり，カラオケに行ったりと楽しい思い出ができました．

福本研究室 19 期生の皆さん，何ごとも前向きに頑張ってください．

最後に，高知工科大学で過ごした 4 年間を支えてくださった皆様，親族の皆様，全ての皆様に感謝致します．

# 参考文献

[1] 田中麻実, 福冨英次, 福本昌弘, “秘密分散バックアップした医療データの部分復元,” 信学技報 IA2015-74, pp.31-36, Dec.2015.

[2] A.Shamir, “How to Share a Secret,” Communication of the ACM, Vol.22, No.11, pp.612-613, Nov.1979.

# 付録 A

# $(k, n)$ しきい値秘密分散法

$(k, n)$ しきい値秘密分散法 [2] は，データを $n$ 個のシェアに分散し，シェアを $k$ 個以上集めることでデータを復元できる方法である．$(k, n)$ しきい値秘密分散法を用いて作成したシェアは，単一のシェアからデータの情報を全く得ることができないためデータの秘匿化が可能である．また，$n$ 個のシェアのうち $n - k$ 個紛失しても元データを復元できるため冗長化ができる．

$(k, n)$ しきい値秘密分散法を用いたデータの分散，復元を以下に示す．

データを $S$，データ $S$ を表現できる 2 の拡大体を $GF(2^m)$ とし，$n$ 個のシェアを $w(i = 1, \cdots, n)$ を作成する．

**分散**

1. $GF(2^m)$ から異なる $n$ 個の $x_i$ と $k - 1$ 個の乱数 $r_j(j = 1, \cdots, k - 1)$ を選択する．
2. シェア $w_i$ を以下の式 (A.1) にて作成する (ただし，$GF(2^m)$ 上で演算を行う)．

$$w_i = S + r_1 x_1 + \cdots + r_{k-1} x_i^{k-1} \tag{A.1}$$

**復元**

1. $w_i$ と $x_i$ のセットを $k$ 個以上集める．
2. シェア $w_i$ と $x_i$ を式 (A.1) に代入し，$k$ 個の線形方程式を求める (ただし，$GF(2^m)$ 上で演算を行う)．
3. $k$ 個の方程式から $S$ と $r_j$ を求めることで，元データを得ることができる．