信州大学大学院
総合理工学研究科

修士論文

秘密情報とシェアのサイズが異なる
ランプ型秘密分散法に関する基礎的検討

指導教員　西新 幹彦 准教授

専攻　工学専攻
分野　電子情報システム学分野
学籍番号　16W2080K
氏名　吉森 祐希

2018 年 2 月 22 日

目次

1　はじめに

今日の情報社会では重要な情報は紛失と漏洩の危険にさらされているため, 管理方法が大切になる．例えば，情報の紛失を防ぎたいならば，大量のコピーを取ればいいが数が増える分，漏洩する危険が高くなる．では，コピーを取らずに管理をした場合はというと，そのデータを紛失すると復元することができなくなってしまう．このように紛失と漏洩に対する対策は一見互いに矛盾しており，両方同時に対策することは難しいように思われる．しかし，この問題を解決する方法として秘密分散法という手法がある．秘密分散法には $(k.L, n)$ ランプ型秘密分散法や完全型 (k, n) しきい値法という方法等がある．完全型 (k, n) しきい値法では，守りたい秘密情報をもとに，ディーラーと呼ばれる者が n 個のシェアと呼ばれるものを作成する．作成されたシェアは n 人のユーザーに一つずつ配布される．もし，秘密情報が必要になった場合には n 人のうち k 人のユーザーが集まることで元の秘密情報が復元できる．また，集まったユーザーが k 人未満の場合は秘密情報を復元することはできない仕組みになっている．さらに，$(k.L, n)$ ランプ型秘密分散法では集まったユーザーが k 人未満でも $k - L$ 人以上が集まることによって秘密に関する部分的な情報が手に入る仕組みになっている．この完全型 (k, n) しきい値法の実現法としては，有限体を用いた Shamir の多項式補間法 [2] が有名である．さらに，文献 [4] によれば，有限体を用いることなくラテン方陣を用いて (2,3) しきい値法を表現できる場合があることから，有限体を定義できないサイズの有限集合でも (k, n) しきい値法が構成できる可能性が指摘されている．本研究では 3 個のラテン方陣が完全型 (2,3) しきい値法の復号表となるための条件を整合性として定義し，直交配列の概念を通して整合性が直交性と等価な概念であることを示した．そして，そのことを利用して有限体の存在しないサイズ 10 の (2,3) しきい値法を構成できることが分かった．一方，文献 [8] により直交配列と強いランプ型秘密分散法の関係が明らかにされており，そのことからランプ型秘密分散法の復号表を直交配列で表現できることが知られている．本研究ではこの研究を一般化してシェアサイズと秘密情報サイズが異なる場合のランプ型秘密分散法について検討し，復号器の構成方法やその復号器の持つ安全性や復元性について既存のランプ型の定義 [3] を用いて評価を行った．その結果，秘密情報サイズがシェアサイズのべき乗の場合は既存のランプ型秘密分散法の定義を満足するような構成方法が分かった．しかし，秘密情報サイズがシェアサイズのべき乗でない場合は既存のランプ型秘密分散法の定義を満足するような構成方法は存在しなかったため様々な観点からの評価基準に関する研究を行った．

2　確率的な秘密分散法

はじめに，確率的な秘密分散法の定義について確認する．

2.1 完全型 (k,n) しきい値法の定義

文献 [1] に基づき完全型 (k,n) しきい値型秘密分散法の定義を導入する．

定義 1 $\mathcal{X}$ をシンボルの集合とする．$S, W_j(j=1,\cdots,n)$ を集合 $\mathcal{X}$ 上に値をとる確率変数とする．確率変数 S とシェア $(W_1, W_2, \ldots, W_n)$ が次の 2 条件を満たすとき，これらの確率変数は完全型 (k,n) しきい値型秘密分散法をなすという [1].

1. 任意の相違なる k 個の確率変数 $(W_{j_1}, W_{j_2}, \ldots, W_{j_k})$ から S が正しく復号できる．つまり，

$$H(S|W_{j_1}, W_{j_2}, \ldots, W_{j_k}) = 0$$

   が成り立つ．
2. 任意の $k-1$ 個の確率変数 $(W_{j_1}, W_{j_2}, \ldots, W_{j_{k-1}})$ から秘密情報 S の値は全く得られない．つまり，

$$H(S|W_{j_1}, W_{j_2}, \ldots, W_{j_{k-1}}) = H(S)$$

   が成り立つ．

このとき，S, W_j をそれぞれ秘密情報，シェアと呼ぶ．

上式で定義される完全型 (k,n) しきい値型秘密分散法のことを本論文では確率的な (k,n) 秘密分散法と呼ぶ．

2.2 (k,L,n) ランプ型秘密分散法の定義

文献 [3] に基づき (k,L,n) ランプ型秘密分散法の定義を導入する．

定義 2 $S_i, W_j(i=1,\ldots,L,\ j=1,\ldots,n)$ を集合 $\mathcal{X}$ 上に値をとる確率変数とする．確率変数 $S_i(i=1,\ldots,L)$ と $W_j(j=1,\cdots,n)$ が次の条件 1 を満たすとき，これらの確率変数は (k,L,n) ランプ型秘密分散法をなすといい，条件 1,2 を満たすときは特に，強い (k,L,n) ランプ型秘密分散法をなすという [3].

1. 任意の $k-t(0 \leqq t \leqq L)$ 個の確率変数からなる組 $(W_{j_1}, W_{j_2}, \ldots, W_{j_{k-t}})$ に対して

$$H(S^L|W_{j_1}, W_{j_2}, \ldots, W_{j_{k-t}}) = \frac{t}{L}H(X^L) = t\log q \tag{1}$$

   が成り立つ．
2. 任意の $k-t$ 個 $(1 \leqq t \leqq L)$ の確率変数からなる組 $(W_{j_1}, W_{j_2}, \ldots, W_{j_{k-t}})$ と $S^L =$

$(S_1, S_2, \ldots, S_L)$ の任意の t 個の確率変数の組 $(S_{s_1}, S_{s_2}, \ldots, S_{s_t})$ に対して

$$H(S_{s_1}, S_{s_2}, \ldots, S_{s_t} | W_{j_1}, W_{j_2}, \ldots, W_{j_{k-t}}) = \frac{t}{L} H(X^L) = t \log q$$

が成り立つ．

このとき，S_i, W_j をそれぞれ秘密情報，シェアと呼ぶ．

上式で定義されるランプ型秘密分散法のことを本論文では確率的な (k, L, n) ランプ型秘密分散法と呼ぶ．

2.3 Shamir の多項式補間法

確率的な秘密分散法の有名な実現例として Shamir の多項式補間法 [2] という手法がある．この方法による秘密情報からシェアを作る手順と秘密情報を復元する手順を説明する．

はじめに，秘密情報やシェアは有限体 $\mathbb{F}$ 上に値をとるとする．秘密情報 S の分布は任意でよい．係数 $a_1, a_2, \cdots, a_{k-1}$ をそれぞれ $\mathbb{F}$ 上に一様に分布する独立な確率変数とし，$k-1$ 次の多項式

$$f(x) = a_{k-1} x^{k-1} + a_{k-2} x^{k-2} + \cdots + a_1 x + S \tag{2}$$

を定める．これを元に，ディーラーが互いに異なる非零の $x_i \in \mathbb{F} (i = 1, 2, \ldots, n)$ に対し，

$$W_i = f(x_i) \tag{3}$$

を作る．この W_i をシェアとし，ユーザーにそれぞれ秘密裏に配布する．$S, a_1, \ldots, a_{k-1}$ については非公開とする．

シェア保有者が協力して k 個のシェアを集めることによって，式 (2) をもとめることができ，秘密情報 S を復元できる．つまり，定義 1.1 を満たしている．さらに，$k-1$ 個のシェアからでは未知数が k 個ある式 (2) を求められず，秘密情報 S を一意に定めることができないため，復元できない．つまり，定義 1.2 も満たしている．

3 非確率的な秘密分散法

文献 [4] によれば，確率的な秘密分散法は非確率的な秘密分散法の一部である．そのため，非確率的な秘密分散法を用いて議論することで，確率的な秘密分散法では実現できないものを見つけられる可能性がある．この章では非確率的な秘密分散法の定義について確認する．

3.1 非確率的な秘密分散法の定義

文献 [4] に基づき非確率的な秘密分散法の定義を導入する．

定義 3　s と w_j をそれぞれ，ある有限集合 $\mathbb{F}$ 上の値をとる変数とする．変数 s とシェア $(w_1, w_2, \cdots, w_n)$ が次の 2 条件を満たすときこれらの変数は非確率的な完全型 (k, n) しきい値秘密分散法をなすという．

1. ある写像 $\psi : (\mathbb{N} \times \mathbb{F})^k \to \mathbb{F}$ が存在し，

$$s = \psi(j_1, w_{j_1}, j_2, w_{j_2}, \ldots, j_k, w_{j_k}) \tag{4}$$

が成り立つ．ここで $\mathbb{N} \triangleq \{1, \cdots, n\}$ とする．ただし，$j_1, j_2, \ldots, j_k$ は互いに異なる．この時の ψ は復号器としての役割を果たす．

2. $\mathbb{F} = \psi(j_1, w_{j_1}, j_2, w_{j_2}, \ldots, j_{k-1}, w_{j_{k-1}}, j_k, \mathbb{F})$ となる．すなわち，

$$g(\cdot) = \psi(j_1, w_{j_1}, j_2, w_{j_2}, \ldots, j_{k-1}, w_{j_{k-1}}, j_k, \cdot) \tag{5}$$

が全単射となる．

このとき，s, w_j をそれぞれ秘密情報，シェアと呼ぶ．

式 (4) は k 個のシェアから秘密情報を復元できることを意味する．また，式 (5) が全単射の関係にあるということは k 個目の値を取り替えると，得られる値が $\mathbb{F}$ 上で重複することなく取り変わるということを意味する．つまり，$k-1$ 個のシェアが集まったとしても，秘密情報について情報が得られないということである．また，秘密情報 s とシェア $(w_1, w_2, \cdots, w_n)$ の値は非公開情報だが，復号器 ψ は非公開情報を含んでいないため公開してもかまわない．

3.2　非確率的な秘密分散法の例

2.3 節で説明した Shamir の多項式補間法は非確率的な秘密分散法でもある．そのことを以下の通り示す．シェアの作り方は基本的に 2.3 節と同じで，秘密情報 s と係数 $a_1, a_2, \cdots, a_{k-1}$ の値は非公開であるが，これらは確率変数ではない．

秘密を復元するための復号器は次のようにして導くことができる．まず，式 (2)(3) で定まる k 個のシェアを行列で表現すると

$$\begin{bmatrix} w_{j_1} \\ w_{j_2} \\ \vdots \\ w_{j_k} \end{bmatrix} = \begin{bmatrix} x_{j_1}^{k-1} & x_{j_2}^{k-2} & \ldots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ x_{j_k}^{k-1} & x_{j_k}^{k-2} & \ldots & 1 \end{bmatrix} \begin{bmatrix} a_{k-1} \\ a_{k-2} \\ \vdots \\ s \end{bmatrix} \tag{6}$$

となる．ここで，右辺の正方行列を [X] と置く．[X] はヴァンデルモンド行列となっているた

め逆行列が必ず存在する．式 (6) の両辺に左から $[X]^{-1}$ をかけると

$$\begin{bmatrix} a_{k-1} \\ a_{k-2} \\ \vdots \\ s \end{bmatrix} = \begin{bmatrix} x_{j_1}^{k-1} & x_{j_2}^{k-2} & \dots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ x_{j_k}^{k-1} & x_{j_k}^{k-2} & \dots & 1 \end{bmatrix}^{-1} \begin{bmatrix} w_{j_1} \\ w_{j_2} \\ \vdots \\ w_{j_k} \end{bmatrix}$$

となる．$[X]^{-1}$ の k 行目部分を $[\cdots]$ と置くと

$$\begin{bmatrix} \dots \end{bmatrix} \begin{bmatrix} w_{j_1} \\ w_{j_2} \\ \vdots \\ w_{j_k} \end{bmatrix} = s$$

となる．これは $j_1, w_{j_1}, j_2, w_{j_2}, \dots, j_k, w_{j_k}$ の値から秘密情報 s を復号する方法を示しており，式 (4) で表される復号器の構造に他ならない．つまり

$$s = \begin{bmatrix} \dots \end{bmatrix} \begin{bmatrix} w_{j_1} \\ w_{j_2} \\ \vdots \\ w_{j_k} \end{bmatrix} = \psi(j_1, w_{j_1}, j_2, w_{j_2}, \dots, j_k, w_{j_k})$$

となり，定義 3 の式 (4) を満たした復号器であることが確認できる．

また，x_i が相異なる非零の値であることから $[\cdots]$ の各要素も非零であることがいえる．$[\cdots]$ の各要素が非零であることから，s の値は $w_{j_1}, \cdots, w_{j_k}$ のすべてに依存し，有限体の演算の性質から s と w_{j_k} が 1 対 1 であることがわかる．このことから定義 3 の式 (5) が満たされることがわかり，$k-1$ 個のシェアから秘密情報 S は復元できない．以上のことから Shamir の多項式補間法が非確率的な秘密分散法となっていることが示された．

4　ラテン方陣を用いた秘密分散法

4 章では，特殊な非確率的な秘密分散法を実現するのに用いるラテン方陣について紹介する．ラテン方陣を用いることで有限体に縛られずに秘密分散法を構成することができる．本研究では，この方法を用いることで有限体を用いた Shamir の多項式補間法では実現できないサイズの秘密分散法について検討した．

4.1　ラテン方陣

シェアのサイズと秘密情報 S のサイズが等しいと仮定した場合，復号器 ψ は複数のラテン方陣で表現できる．n 行 n 列の表の各行各列に，それぞれ n 種類の記号が重複しないように現

れる表のことを n 次のラテン方陣またはサイズ n のラテン方陣と言う．例として，図 1 にサイズ 5 のラテン方陣の例を示す．図 1 に示したように，ラテン方陣には各行各列に同じ数字が表れない性質がある．この性質の特徴から，シェアをラテン方陣のマスの座標とすることで，任意の k 個のシェアで秘密情報の一点が定まる．つまり，秘密情報が復元される．また，$k-1$ 個のシェアではラテン方陣内の取りうる値に全ての数字が現れるような復号器を実現できる．

0	1	2	3	4
1	2	3	4	0
2	3	4	0	1
3	4	0	1	2
4	0	1	2	3

図 1　サイズ 5 のラテン方陣の例

4.2　ラテン方陣を用いた (2,2) しきい値法

図 1 のラテン方陣に座標を与えたものを図 2 に示す．図 2 において，座標部分がシェアを，ラテン方陣のマスの値が復号すべき値を表す．また，シェアは図 2 のようにそれぞれ w_1, w_2 とする．図 2 を用いてシェアを作ったり秘密情報を復号できることを説明する．

まず，秘密情報が s であると仮定する．次に，一方のシェア w_1 の値を他人に知られないようにディーラーが勝手に選ぶ (確率的に選択するという意味ではない)．また，他方のシェア w_2 の値はラテン方陣の w_1 行 w_2 列の値が秘密情報 s と等しくなるように選ぶ．例として，秘密情報 s が 3 だと仮定する．次に，ディーラーがシェア w_1 の値を 2 に決めたとする．ここで図 2 を見ると，$w_1 = 2$ で秘密情報が 3 の場合もう一方のシェア w_2 の値は 1 であることが分かる．つまり，シェアの値は，$w_1 = 2, w_2 = 1$ と設定される．

次に，秘密情報がどのように復号されるかを考える．まず，シェア w_1 のみを入手した場合を考える．シェア w_1 から得られる情報は，図 2 よりシェア w_2 のとりうる値と秘密情報のと

		w_2				
		0	1	2	3	4
w_1	0	0	1	2	3	4
	1	1	2	3	4	0
	2	2	3	4	0	1
	3	3	4	0	1	2
	4	4	0	1	2	3

図 2　非確率的な (2,2) しきい値法の復号表の例

りうる値は全単射となっているため，w_2 の値に対して同じ数の秘密情報が出現し，w_1 のみでは秘密情報を復号できない．シェア w_2 のみを入手した場合も同様に秘密情報は全く分からない．2 つのシェア $w_1 = 2, w_2 = 1$ を入手した場合は図 2 より，2 つのシェアの値から 1 マスの座標が求まり，そのマスの情報が秘密情報であるため，2 つのシェアから秘密情報が得られる．この場合は秘密情報として 3 が得られる．

このようにして，指定されたサイズの任意のラテン方陣を用いて非確率的な (2,2) 秘密分散法を表現することができる．

4.3 ラテン方陣の整合性を用いた (2,3) しきい値法

(2,3) しきい値法は，3 個のシェアのうち 2 個のシェアを集めることで秘密情報が復元できるという方法である．4.2 節で説明した方法を拡張することで，(2,3) しきい値法も複数個のラテン方陣を用いて表現できる．そのためにはラテン方陣の整合性という概念が必要となる．以下でこの概念について定義する．

まず，これらの概念を定義するために必要な記号や表を図 3 のように設定する．図 3 のようにサイズ n の 3 個のラテン方陣を準備し，図のように座標を取る．ここで，それぞれの f はラテン方陣を表し，2 個の座標の値からその座標のマスの値への写像である．これらを用いてラテン方陣の整合性を以下のように定義する．

定義 4 同じサイズの 3 個のラテン方陣が整合性を持つとは，任意の w_1,w_2 に対し, w_{31}, w_{32} をそれぞれ

$$f_2(w_2, w_{31}) = f_1(w_1, w_2),$$
$$f_3(w_1, w_{32}) = f_1(w_1, w_2)$$

により一意に定まる値とするとき，$w_{31} = w_{32}$ が成り立つことである．

次に，ラテン方陣の整合性を用いた (2,3) しきい値法の構成について説明する．サイズ 5 の整合性を満たした 3 個のラテン方陣の例を図 4 に示す．ここで，座標部分はシェアの値，ラテン方陣のマスの値は秘密情報の値を表すものとする．図 4 を復号表として用いた時の (2,3) しきい値法の 3 個のシェアの作り方を例を用いて説明する．

	w_2		
	0	$\cdots$	$n-1$
0			
w_1 $\vdots$	$f_1(w_1, w_2)$		
$n-1$			

	w_3		
	0	$\cdots$	$n-1$
0			
w_2 $\vdots$	$f_2(w_2, w_3)$		
$n-1$			

	w_3		
	0	$\cdots$	$n-1$
0			
w_1 $\vdots$	$f_3(w_1, w_3)$		
$n-1$			

図 3　任意の 3 個のラテン方陣

s		w_2				
		0	1	2	3	4
	0	0	4	3	2	1
	1	2	1	0	4	3
w_1	2	4	3	2	1	0
	3	1	0	4	3	2
	4	3	2	1	0	4

s		w_3				
		0	1	2	3	4
	0	0	3	1	4	2
	1	3	1	4	2	0
w_2	2	1	4	2	0	3
	3	4	2	0	3	1
	4	2	0	3	1	4

s		w_3				
		0	1	2	3	4
	0	0	2	4	1	3
	1	4	1	3	0	2
w_1	2	3	0	2	4	1
	3	2	4	1	3	0
	4	1	3	0	2	4

図 4　ラテン方陣の整合性を用いた (2,3) しきい値法の復号表の例

秘密情報の値を 3 と仮定する．そして，ディーラーが 4.2 節と同様に w_1 の値を勝手に決める．ここで，$w_1 = 2$ に決定したと仮定する．この場合，1 個目の表 (w_1-w_2 の表) では $w_1 = 2, w_2 = 1$ のマスにあたる 3 が守りたい秘密情報となる．したがって，シェア w_2 の値は $w_2 = 1$ となる．次に，2 個目の表 (w_2-w_3 の表) において $w_2 = 1$ の行に注目すると，その行における 3 にあたるマスの w_3 の値は 0 であるため，$w_3 = 0$ と求まる．以上より，3 個のシェアの値は $w_1 = 2, w_2 = 1, w_3 = 0$ となる．

では，例で作成した 3 個のシェアを用いて復元性と安全性の確認をする．(2,3) しきい値法は 3 個のシェアのうち任意の 2 個のシェアを入手すると元の情報が求まり，1 個以下では元の情報について何も分からないような仕組みである．1 個のシェアを入手した場合について考える．復号器にはラテン方陣を用いているので元の情報については何も分からない．よって，安全性については満たされていることが分かる．次に 2 個のシェアを入手した場合について考える．例として w_1, w_3 の 2 個のシェアを入手したとすると，3 個目の表 (w_1-w_3 の表) をもちいて $w_1 = 2, w_3 = 0$ のマスを確認すると 3 が求まり，元の秘密情報が復元されていることが分かる．また，整合性を満たした 3 個のラテン方陣を復号器に用いているため，どの 2 個のシェアの組み合わせからでも 3 が秘密情報として復元される．したがって，復元性についても満たされていることが確認できる．同様にして，どの値を秘密情報としてもすべてのラテン方陣で正しい秘密情報が復号できることがわかる．従って，整合性を満たした 3 個のラテン方陣を復号器に用いることで (2,3) しきい値法を構成することができる．

4.4　ラテン方陣の直交性を用いた (2,3) しきい値法

複数個のラテン方陣には整合性とは別に直交性という概念が存在する．(2,3) しきい値法は直交性を用いることでも構成することができる．以下でラテン方陣の直交性を定義する．

定義 5　同じサイズの 2 個のラテン方陣が直交性を持つとは，図 5 のようなサイズ n の任意の 2 個のラテン方陣 f_1, f_2 を選び，同じ座標におけるマスの値 $f_1(w_1, w_2), f_2(w_1, w_2)$ をペアとしてみた時に n^2 通りの値のペアが各々一回ずつ出現するような性質を持つことである．

	w_2	
	0 ⋯	n-1
	0	
w_1	⋮	$f_1(w_1, w_2)$
	n-1	

	w_2	
	0 ⋯	n-1
	0	
w_1	⋮	$f_2(w_1, w_2)$
	n-1	

図 5　任意の 2 個のラテン方陣

次に，ラテン方陣の直交性を用いた (2,3) しきい値法の構成について例を用いて説明する．サイズ 5 の直交ラテン方陣の例を図 6 に示す．直交ラテン方陣を復号表に用いる場合，図 6 のように行の座標は秘密情報の値，列の座標は w_1 の値，直交ラテン方陣のマスの値は w_2, w_3 の値を表すものとする．

はじめに，シェアの作り方について説明する．例として，守りたい秘密情報の値が 3 であると仮定する．そして，ディーラーが 4.2 節と同様に w_1 の値を勝手に決める．ここで，$w_1 = 2$ に決定したと仮定する．すると，復号表の $s = 3, w_1 = 2$ のマスの値から $w_2 = 1, w_3 = 0$ と求まり，3 個のシェアの値が確定する．

次に，作ったシェアの値を用いて復元性について説明する．まず，w_1 と w_2 の値を入手した場合について考える．2 個のシェアの値 $w_1 = 2, w_2 = 1$ より，図 6 においてある 1 個のマス ($s = 3, w_1 = 2$ のマス) が特定される．したがって，そのマスの座標を求めるように s の値を確認すると $s = 3$ となり，元の秘密情報が復元されることが分かる．w_1 と w_3 を入手した場合についても同様である．次に，w_2 と w_3 を入手した場合について考える．直交ラテン方陣の性質により，図 6 の表のマス部分には $w_2 = 1, w_3 = 0$ の値の組み合わせは 1 個しかないので，その値をとるマスが一意に定まる．ここで，図 6 において $w_2 = 1, w_3 = 0$ のマスの座標を確認すると，$s = 3, w_1 = 2$ と求まる．したがって，元の秘密情報である $s = 3$ が復元されていること確認できる．

最後に安全性について確認をする．シェアが 1 個も入手できない場合は秘密情報に関して何も分からないのは明らかである．$w_1 = 2$ を入手した場合，図 6 より $w_1 = 2$ の 1 列が求まるが，ラテン方陣の持つ性質から w_2 および w_3 の値には 5 種類の値が出現するため，行の座標を一意に特定することができない．したがって，$w_1 = 2$ だけでは秘密情報は分からない．次に，$w_2 = 1$ を入手した場合について考える．ラテン方陣の性質から，$w_2 = 1$ は全ての行に 1 回ずつ出現するため，行の座標を一意に特定することができない．したがって，$w_2 = 1$ だけでは秘密情報は分からない．w_3 についても同様である．以上より，直交ラテン方陣を用いることによって (2,3) しきい値法を構成することができる．

w_2, w_3		w_1				
		0	1	2	3	4
	0	0,0	2,3	4,1	1,4	3,2
	1	4,3	1,1	3,4	0,2	2,0
s	2	3,1	0,4	2,2	4,0	1,3
	3	2,4	4,2	1,0	3,3	0,1
	4	1,2	3,0	0,3	2,1	4,4

図 6　直交ラテン方陣を用いた (2,3) しきい値法の復号表の例

4.5　有限体の定義できない (2,3) しきい値法

Shamir の多項式補間法ではサイズ 6 や 10 の (2,3) しきい値法は構成することができない．その理由は位数 6,10 の有限体が存在しないからである．しかし，サイズ 10 の直交ラテン方陣は存在する．その一例を図 7 に示す．つまり，4.4 節で紹介した方法を用いることでサイズ 10 の (2,3) しきい値法を構成することができる．ただし，サイズ 6 の直交ラテン方陣は存在しない [7] ので，サイズ 6 の (2,3) しきい値法は構成することができない．

0,0	1,1	2,9	3,2	4,3	5,8	6,4	7,6	8,5	9,7
3,6	4,7	0,8	1,9	2,5	7,2	9,3	8,1	6,0	5,4
4,9	3,3	1,7	2,4	0,6	9,5	7,8	6,2	5,1	8,0
1,3	2,8	4,2	0,5	7,4	8,7	5,9	3,0	9,6	6,1
2,1	0,4	3,5	7,0	5,7	6,3	8,6	9,9	4,8	1,2
5,2	7,5	6,6	9,1	8,9	3,4	4,0	1,8	2,7	0,3
8,4	9,0	7,1	5,3	6,8	1,6	2,2	0,7	3,9	4,5
6,5	5,6	9,4	8,8	1,0	4,1	3,7	2,3	0,2	7,9
9,8	8,2	5,0	6,7	3,1	0,9	1,5	4,4	7,3	2,6
7,7	6,9	8,3	4,6	9,2	2,0	0,1	5,5	1,4	3,8

図 7　サイズ 10 の直交ラテン方陣 [5]

4.6　直交配列

この節ではラテン方陣の別表現である直交配列という概念を導入する．この概念は 4.7 節で整合性と直交性の等価性を証明するときに必要となる．文献 [6] に基づき直交配列の定義を導入する．

定義 6　N 行 k 列の配列を A, アルファベットを $\mathcal{S}$, $t(1 \leq t \leq k)$ を整数とする．配列 A にお

ける k 列の中から任意に t 列を選び，N 行 t 列の部分配列を作る．このとき，その部分配列の行において $|\mathcal{S}|^t$ 種類の文字列が各々 λ 回現れる場合，その配列 A をサイズ s, 強さ t, インデックス λ の直交配列という [6].

直交配列は上記のように定義されるが，$\lambda = 1$ の場合が最も重要であり，本研究で取り扱う直交配列は全て $\lambda = 1$ として取り扱う．よって，以降の直交配列は全て $\lambda = 1$ のものとする．また $\lambda = 1$ の時，直交配列に関わるパラメータは N, k, s, t の 4 個の値であるので，サイズ s, 強さ t の N 行 k 列の直交配列を $OA(N, k, s, t)$ と表す．次に，上記で定義した直交配列について例を用いて説明する．

直交配列 $OA(8, 4, 2, 3)$ の一例を図 8 に示す．図 8 はサイズ 2, 強さ 3 の 8 行 4 列の直交配列であるため，4 列の内どの 3 列を選んでも 2^3 通りの文字列が各々 1 回ずつ現れるような性質を持っている．確認のために図 8 の配列から 8 行 3 列の部分配列を作成する．図 9 を確認すると図 8 からできる 4 個の部分配列全てにおいて，2^3 通りの文字列が各々 1 回ずつ現れていることが確認できる．

列番号			
1	2	3	4
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

図 8　直交配列 $OA(8, 4, 2, 3)$

列番号		
1	2	3
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

列番号		
1	2	4
0	0	0
0	0	1
0	1	1
0	1	0
1	0	1
1	0	0
1	1	0
1	1	1

列番号		
1	3	4
0	0	0
0	1	1
0	0	1
0	1	0
1	0	1
1	1	0
1	0	0
1	1	1

列番号		
2	3	4
0	0	0
0	1	1
1	0	1
1	1	0
0	0	1
0	1	0
1	0	0
1	1	1

図 9　8 行 3 列の部分配列

4.7 等価性の証明

4.3 節と 4.4 節ではそれぞれラテン方陣に関する整合性と直交性の概念を説明した．しかし，研究を通してこの 2 個の概念に類似点が見られた．次の 2 つがその類似点の根拠である．1 つ目は，本研究で整合性を満たす 3 個のラテン方陣を探索するプログラムを作り，サイズ 2 からサイズ 7 の範囲でそのようなラテン方陣の組み合わせを探索したが，サイズが 2 と 6 の組み合わせを発見できなかったことである．2 つ目は，文献 [7] によるとサイズ 2 と 6 の直交ラテン方陣が存在しないことが知られていることである．これらの共通点から 2 個の概念は等価なものではないかという予測を立てた．そこで，この節では整合性と直交性が等価であることを証明する．以下でこの証明を行うが，証明は 2 つの部分からなっている．前半では整合性から直交性を導出し，後半では直交性から整合性を導出する．

まず証明の前半として，整合性のある 3 個のラテン方陣を用いて直交ラテン方陣を作成できることを導出する．サイズ n の整合性のある 3 個のラテン方陣を準備する．

		w_2		
		0	$\cdots$	$n-1$
	0	○	$\cdots$	○
w_1	$\vdots$	$\vdots$	$\ddots$	$\vdots$
	$n-1$	○	$\cdots$	○

		w_2		
		0	$\cdots$	$n-1$
	0	○	$\cdots$	○
w_1	$\vdots$	$\vdots$	$\ddots$	$\vdots$
	$n-1$	○	$\cdots$	○

		w_2		
		0	$\cdots$	$n-1$
	0	○	$\cdots$	○
w_1	$\vdots$	$\vdots$	$\ddots$	$\vdots$
	$n-1$	○	$\cdots$	○

図 10　整合性を満たす 3 個のラテン方陣

次に準備した 3 個のラテン方陣を用いて，秘密情報 s, シェア w_1, w_2, w_3 を座標に取る配列を以下のように作る．図 10 の中から任意に 2 個のラテン方陣を選ぶ．選んだ表の片方から 1 マス選び，そのマスに対応する座標を配列の行として並べて図 11 のような配列表を作る．ここで，3 個のラテン方陣のうちどの 2 個を選んでもラテン方陣の整合性から同じ配列ができる．このようにして作った図 11 のような配列はラテン方陣のマスの数だけ行数があるので，n^2 行 4 列の配列である．

s	w_1	w_2	w_3
○	○	○	○
○	○	○	○
$\vdots$	$\vdots$	$\vdots$	$\vdots$
○	○	○	○

図 11　図 10 から作った配列

次に，作られた配列が強さ 2 の直交配列であることを示す．強さ 2 の直交配列であることを示すためには，4 個の列の内，どの 2 個の列を選んでも n^2 通りの文字列が各々 1 回ずつ出現することを示せばよい．従って，次の 2 通りの場合に分けて考える．

1. 秘密情報 s と 3 個のシェア w_1, w_2, w_3 の中から 1 個選ぶ．このとき w_1 の値を固定した場合，ラテン性から秘密情報 s に n 通りの値が各々 1 回ずつ現れる．つまり，秘密情報 s とシェア w_1 の組み合わせは全部で n^2 通りある．また，w_1 を w_2, w_3 に置き換えても同様のことが言える．したがって，秘密情報 s と 3 個のシェア w_1, w_2, w_3 の中から 1 個選んだ場合，その組み合わせは全部で n^2 通り存在する．
2. 3 個のシェア w_1, w_2, w_3 の中から 2 個選ぶ．このとき表の座標の組み合わせであるから，その組み合わせは高々 n^2 通り存在する．(w_1, w_2) の組み合わせが n^2 通りあることは自明である．

以上より，得られた表は強さ 2 の直交配列であることが示された．

次に得られた直交配列を元に図 11 のような s と w_1 の値を座標に取り，その座標に対応するマスに w_2, w_3 の値を埋めて図 12 のような表を作る．このようにして作った図 12 のような表が直交ラテン方陣であることを示す．直交配列の性質から w_2, w_3 の値のとりうる組み合わせは n^2 通りである．それに加えて，s または w_1 の値を固定した場合，w_2 と w_3 は n 通りの値を取る．以上より，図 12 は直交ラテン方陣であることが言える．

	w_1		
	0	$\cdots$	$n-1$
0			
s $\vdots$		w_2, w_3	
$n-1$			

図 12　図 11 を用いて作った表

さて次に証明の後半として，直交ラテン方陣を用いて整合性を満たす 3 個のラテン方陣を作成できることを導出する．サイズ n の直交ラテン方陣を準備する．

	w_1		
	0	$\cdots$	$n-1$
0			
s $\vdots$		w_2, w_3	
$n-1$			

図 13　サイズ n の直交ラテン方陣

図 13 から図 11 のような配列を作成する．ここで，得られた配列が強さ 2 の直交配列であることを示す．整合性から直交性を導き出した証明と同様に 4 個の列の内，どの 2 個の列を選んでも n^2 通りの文字列が各々 1 回ずつ出現することを示せばよい．従って，次の 3 通りの場合に分けて考える．

1. w_2, w_3 を選ぶ．このとき，直交ラテン方陣の性質より n^2 通りの文字列が各々 1 回ずつ出現することは自明である．
2. s, w_1 を選ぶ．このとき座標の組み合わせであるから，n^2 通りの文字列が各々 1 回ずつ出現することは自明である．
3. s または w_1 から 1 個，w_2 または w_3 から 1 個選ぶ．このときラテン性より，n^2 通りの並びが出現する．

以上より，得られた配列が直交配列であることが言える．

次に，この直交配列を元に w_1, w_2, w_3 の値を座標に取り，その座標に対応する s の値を埋めて図 14 のような 3 個の表を作る．直交配列の性質からこの表は全てラテン方陣であり，直交配列の各行を見れば作った 3 個のラテン方陣は整合性を満たしていることが確認できる．

以上の証明により，ラテン方陣の整合性と直交性が等価なものであることが言えた．

		w_2		
		0	$\cdots$	$n-1$
	0	○	$\cdots$	○
w_1	$\vdots$	$\vdots$	$\ddots$	$\vdots$
	$n-1$	○	$\cdots$	○

		w_2		
		0	$\cdots$	$n-1$
	0	○	$\cdots$	○
w_1	$\vdots$	$\vdots$	$\ddots$	$\vdots$
	$n-1$	○	$\cdots$	○

		w_2		
		0	$\cdots$	$n-1$
	0	○	$\cdots$	○
w_1	$\vdots$	$\vdots$	$\ddots$	$\vdots$
	$n-1$	○	$\cdots$	○

図 14　得られた直交配列から作成した 3 個の表

5　直交配列を用いた秘密分散法

4.7 節の証明の中でも触れたように，ラテン方陣は強さ 2 の直交配列の別表現である．したがって，直交配列は秘密分散法の復号表として用いることができる．秘密分散法と直交配列に関する具体的な関係性は文献 [8] において，強いランプ型秘密分散法と MDS 符号が等価な関係であることを示すのに用いられている．そこで，本研究では文献 [8] で用いられていた強いランプ型秘密分散法と直交配列の関係に焦点を当て，特殊な直交配列の探索アルゴリズムについて研究した．

5.1　ランプ型秘密分散法と直交配列

文献 [8] で知られている強いランプ型と直交配列の関係性について具体的に紹介する．アルファベット $\mathcal{S}$ 上に値をとる L 個の秘密情報 $S_1, S_2, \cdots, S_L$ と n 個のシェア $w_1, w_2, \cdots, w_n$ を持つ (k, L, n) ランプ型秘密分散法の復号器は強さ k の $|\mathcal{S}|^k$ 行 $L+n$ 列の直交配列で表現でき，さらに直交配列と MDS 符号は等価関係である．つまり，求めたいランプ型の具体的な (k, L, n) の値を設定し，その値から必要な直交配列を発見できればその秘密分散法を構成することができる．

5.2　サイズ 4 の (2,2,3) ランプ型秘密分散法

多項式補間法による (2,2,3) ランプ型秘密分散法は次のように構成できる [9].

まず，多項式 $f(x) = a_0 + a_1 x + a_2 x^2$ を考える．この多項式を使い，秘密情報 s_1, s_2 をそれぞれ $f(0), f(1)$ とする．次に，w_1 の値をある有限体上の値から勝手に選ぶ．そして，s_1, s_2, w_1 から次式を満たす a_0, a_1, a_2 を求める．

$$f(0) = s_1, f(1) = s_2, f(2) = w_1$$

これらを元の多項式に代入すると次の 3 式が得られる．

$$f(0) = a_0 = s_1$$

$$f(1) = a_0 + a_1 + a_2 = s_2$$

$$f(2) = a_0 + 2a_1 + 4a_2 = w_1$$

この 3 式を用いて a_0, a_1, a_2 を求め，復号器 $f(x) = a_0 + a_1 x + a_2 x^2$ を作り，この復号器の x に異なる 2 種類の値 (0,1,2 以外を用いる) を代入することで，さらに 2 個のシェアを作り出せる．こうして (2,2,3) ランプ型秘密分散法が構成できるが，この方法では s_1, s_2, w_1 の作成に 3 種類の値を用い，これに加えて w_2, w_3 の作成に追加で 2 種類の値が必要であるため，最低 5 種類の値を用いる必要がある．つまり位数が 5 以上でなければならない．つまり，この方法ではサイズ 4 の (2,2,3) ランプ型秘密分散法は構成することができない．

そこで，本研究では多項式補間法で構成できないサイズ 4 の (2,2,3) ランプ型秘密分散法を探索した．探索の方法としては，この秘密分散法と等価な直交配列 $\mathrm{OA}(16, 5, 4, 2)$ が存在するかどうかを確認するための探索プログラムを作成した．この探索プログラムを用いて発見した直交配列 $\mathrm{OA}(16, 5, 4, 2)$ の一例を図 15 に示す．

s_1	s_2	w_1	w_2	w_3	s_1	s_2	w_1	w_2	w_3
0	0	0	0	0	2	0	2	3	1
0	1	1	1	1	2	1	3	2	0
0	2	2	2	2	2	2	0	1	3
0	3	3	3	3	2	3	1	0	2
1	0	1	2	3	3	0	3	1	2
1	1	0	3	2	3	1	2	0	3
1	2	3	0	1	3	2	1	3	0
1	3	2	1	0	3	3	0	2	1

図 15　直交配列 $(16, 5, 4, 2)$ の 1 例

また，この探索プログラムを付録に記す．本研究では探索プログラムを作成し，目的の直交配列を発見したが，発見した直交配列が有限体を用いて表現できるのか等の検証が課題として残っている．

6　秘密情報サイズとシェアサイズが異なるランプ型秘密分散法

5 章までは秘密情報サイズとシェアサイズが等しいものであると仮定していた．しかし，秘密情報サイズとシェアサイズが異なっている場合，2 章での数式による定義ができなかったり，直交配列やラテン方陣を用いた具体的な復号表の構成方法がない等の問題が知られている．そこで本章では，5 章までの研究結果を一般化し，秘密情報とシェアのサイズが異なるランプ型秘密分散法の実現法について検討する．ここで，本研究で目標とする安全性に対する評価基準について説明する．

本研究では (k, L, n) ランプ型の持つ，ある一定個数以上のシェアを集めた場合に k 個に近づくにつれ，秘密情報の候補が絞られるという特徴に焦点を当てる．この特徴に対して，確率的なランプ型の定義の条件 1 のように秘密情報のエントロピーが変化するようにしたい．このような性質を持つ復号表の構成とその構成における安全性についての考察を 2 通りの場合に分けて行った．

6.1　秘密情報サイズがシェアサイズのべき乗の場合

秘密情報サイズがシェアサイズのべき乗であるとき，直交配列を応用することで復号表を構成することができる．その復号表の一例を図 16 に示す．図 16 は秘密情報のサイズが $2^3 = 8$, シェアのサイズが 2 の (4,3,4) ランプ型の復号表の一例である．(4,3,4) ランプ型秘密分散法はシェアが 4 個あり，その 4 個全てを集めることで秘密情報が一意に定まる．また，シェアを 2

s	w_1	w_2	w_3	w_4
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1

s	w_1	w_2	w_3	w_4
7	1	0	0	0
6	1	0	0	1
5	1	0	1	0
4	1	0	1	1
3	1	1	0	0
2	1	1	0	1
1	1	1	1	0
0	1	1	1	1

図 16　(4,3,4) ランプ型秘密分散法の復号表の例

個以上集めることで秘密情報が徐々に絞られる性質を持つ．ここで，図 16 がこの性質を満たしているかを確認する．例えば，$w_3 = 0$ と $w_4 = 0$ の 2 個のシェアを集めた場合，秘密情報 s が $0, 3, 4, 7$ の 4 通りに絞られていることが確認できる．次に，$w_1 = 1, w_3 = 0, w_4 = 0$ の 3 個のシェアを集めた場合を考えると，秘密情報 s は $4, 7$ の 2 通りに絞られることが確認できる．最後に，$w_1 = 1, w_2 = 1, w_3 = 0, w_4 = 0$ の 4 個のシェアを集めた場合を考えると，秘密情報は $s = 4$ と一意に定まっていることが確認できる．この性質は直交配列が持っている性質を利用しているため，どのシェアをどんな値で選んでもこのような絞られ方になる．よって，図 16 の表は (4,3,4) ランプ型の復号表になっている．また，この復号表の構成方法については 5.2 節で紹介した有限体上での多項式補間法を応用することで作成することができる．

6.2 秘密情報サイズがシェアサイズのべき乗でない場合

秘密情報のサイズがシェアのサイズのべき乗でない場合，復号表のシェアの部分 (直交配列) の行数と秘密情報の数が一致しない．そこで，6.1 節の方法で作った復号表から必要分だけ行を削って秘密情報のサイズを調整した表を復号表として扱う方法を考える．

具体的な復号表の構成方法を秘密情報サイズが 8, シェアサイズが 3 の (4,2,4) ランプ型秘密

s	w_1	w_2	w_3	w_4
0	0	0	0	0
1	0	0	0	1
2	0	0	0	2
3	0	0	1	0
4	0	0	1	1
5	0	0	1	2
6	0	0	2	0
7	0	0	2	1
8	0	0	2	2
5	0	1	0	0
3	0	1	0	1
4	0	1	0	2
8	0	1	1	0
6	0	1	1	1
7	0	1	1	2
2	0	1	2	0
0	0	1	2	1
1	0	1	2	2
7	0	2	0	0
8	0	2	0	1
6	0	2	0	2
1	0	2	1	0
2	0	2	1	1
0	0	2	1	2
4	0	2	2	0
5	0	2	2	1
3	0	2	2	2

s	w_1	w_2	w_3	w_4
4	1	0	0	0
5	1	0	0	1
3	1	0	0	2
7	1	0	1	0
8	1	0	1	1
6	1	0	1	2
1	1	0	2	0
2	1	0	2	1
0	1	0	2	2
6	1	1	0	0
7	1	1	0	1
8	1	1	0	2
0	1	1	1	0
1	1	1	1	1
2	1	1	1	2
3	1	1	2	0
4	1	1	2	1
5	1	1	2	2
2	1	2	0	0
0	1	2	0	1
1	1	2	0	2
5	1	2	1	0
3	1	2	1	1
4	1	2	1	2
8	1	2	2	0
6	1	2	2	1
7	1	2	2	2

s	w_1	w_2	w_3	w_4
8	2	0	0	0
6	2	0	0	1
7	2	0	0	2
2	2	0	1	0
0	2	0	1	1
1	2	0	1	2
5	2	0	2	0
3	2	0	2	1
4	2	0	2	2
1	2	1	0	0
2	2	1	0	1
0	2	1	0	2
4	2	1	1	0
5	2	1	1	1
3	2	1	1	2
7	2	1	2	0
8	2	1	2	1
6	2	1	2	2
3	2	2	0	0
4	2	2	0	1
5	2	2	0	2
6	2	2	1	0
7	2	2	1	1
8	2	2	1	2
4	2	2	2	0
1	2	2	2	1
2	2	2	2	2

s	w_1	w_2	w_3	w_4
0	0	0	0	0
1	0	0	0	1
2	0	0	0	2
3	0	0	1	0
4	0	0	1	1
5	0	0	1	2
6	0	0	2	0
7	0	0	2	1
*	*	*	*	*
5	0	1	0	0
3	0	1	0	1
4	0	1	0	2
*	*	*	*	*
6	0	1	1	1
7	0	1	1	2
2	0	1	2	0
0	0	1	2	1
1	0	1	2	2
7	0	2	0	0
*	*	*	*	*
6	0	2	0	2
1	0	2	1	0
2	0	2	1	1
0	0	2	1	2
4	0	2	2	0
5	0	2	2	1
3	0	2	2	2

s	w_1	w_2	w_3	w_4
4	1	0	0	0
5	1	0	0	1
3	1	0	0	2
7	1	0	1	0
*	*	*	*	*
6	1	0	1	2
1	1	0	2	0
2	1	0	2	1
0	1	0	2	2
6	1	1	0	0
7	1	1	0	1
*	*	*	*	*
0	1	1	1	0
1	1	1	1	1
2	1	1	1	2
3	1	1	2	0
4	1	1	2	1
5	1	1	2	2
2	1	2	0	0
0	1	2	0	1
1	1	2	0	2
5	1	2	1	0
3	1	2	1	1
4	1	2	1	2
*	*	*	*	*
6	1	2	2	1
7	1	2	2	2

s	w_1	w_2	w_3	w_4
*	*	*	*	*
6	2	0	0	1
7	2	0	0	2
2	2	0	1	0
0	2	0	1	1
1	2	0	1	2
5	2	0	2	0
3	2	0	2	1
4	2	0	2	2
0	2	1	0	0
1	2	1	0	1
0	2	1	0	2
4	2	1	1	0
5	2	1	1	1
3	2	1	1	2
7	2	1	2	0
*	*	*	*	*
6	2	1	2	2
3	2	2	0	0
4	2	2	0	1
5	2	2	0	2
6	2	2	1	0
7	2	2	1	1
*	*	*	*	*
0	2	2	2	0
1	2	2	2	1
2	2	2	2	2

図 17　(4,2,4) ランプ型秘密分散法の復号表の例

分散法の復号表を例に説明する．まず，行を削るのに用いる表には秘密情報サイズがシェアサイズのべき乗の値の中で最も近い値の復号表を用いる．つまり，この場合は秘密情報サイズが $3^2 = 9$, シェアサイズが 3 の (4,2,4) ランプ型の復号表を作り，その表から 1 種類の秘密情報を全て消去する．ここで，図 17 の左図は秘密サイズ 9, シェアサイズ 3 の復号表の一例を表し，右図は左図から $s = 8$ の行を全て消去して作った秘密情報のサイズが 8, シェアのサイズが 3 の (4,2,4) ランプ型秘密分散法の復号表である．この場合，どの 2 個のシェアを集めても秘密情報については何も分からない．しかし，3 個のシェアを集めた時，行の消去の仕方によって秘密情報の漏れ方が変化する．本研究では図 17 の左の復号表をベースに行の消去の仕方を変化させて例を作り，3 個のシェアを集めた場合についてエントロピーの観点から考察を行った．

6.3 秘密情報のサイズと安全性

6.2 節の方法で作成した復号表における安全性を条件付きエントロピー [10] を用いて評価した．具体的には，4 個のシェアの中から幾つかのシェアを集めたという条件の下での秘密情報のエントロピーを用いた．また，以降は条件付きエントロピーのことを単にエントロピーという．

ここで，(4,2,4) ランプ型秘密分散法が満たすべき安全性を確認するため，確率的なランプ型の定義である式 (1) に基づいて，集めたシェアの数と秘密情報のエントロピーの関係を図 18 に示す．(4,2,4) ランプ型秘密分散法では図 18 のように集めたシェアが 2 個以下では秘密情報は全く分からないが，3 個の時にエントロピーの値が半分にならなければならない．本研究では集めたシェアの数が 1 個から 4 個のそれぞれの個数の場合における秘密情報のエントロピーを求め，その値が確率的なランプ型の定義である式 (1) を満たすような復号表の探索を

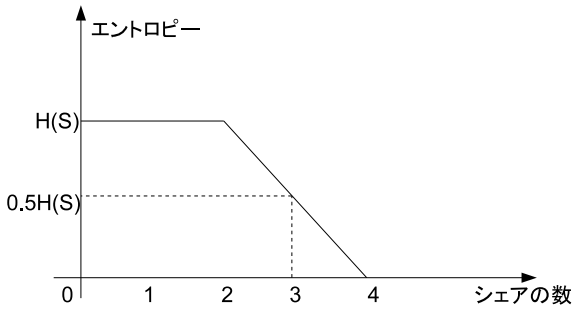


図 18　(4,2,4) ランプ型秘密分散法のエントロピーの変化

行った．具体的には，シェアサイズを 3 に固定して秘密情報のサイズが 8, 7, 6 の 3 つの場合に分けて考察を行った．その結果を以下で順に説明する．

はじめに，秘密情報サイズが 8 の場合の結果を説明する．秘密情報サイズを 8 にするために例として図 17 から秘密情報が 8 の行を消去した場合について考える．このとき，エントロピーの値は図 19 のようになった．図 19 より，3 個のシェアを集めた場合に確率的なランプ型の定義に基づいたエントロピーを満たすような値は得られなかった．また，8 以外に 0 から 7 のどの秘密情報を消去してもエントロピーの値に変化はなかった．

次に，秘密情報サイズが 7 の場合の結果を説明する．秘密サイズを 7 にするために例として図 17 から秘密情報が 0 と 8 の行を消去した場合の秘密情報のエントロピーの値は図 20 のようになった．図 20 の右のグラフは W_2, W_3, W_4 を選んだ場合のエントロピーのグラフで，左のグラフはそれ以外を選んだグラフである．この結果から，シェアの選び方によってエントロピーの値に変化が生じることが分かった．その原因としては秘密情報の消去の仕方によって 3 個のシェアを選んだ時の値の分布が変化したことが考えられる．実際に図 17 から 0 と 8 の行を消去した場合，W_1, W_2, W_3 を選んだ時には 000 の並びが 2 回出現しているが，W_2, W_3, W_4 を選んだ時には 000 の並びが 1 回のみの出現であることが確認できた．このように消去の仕方で 3 個のシェアの値の分布が変化し，それに伴ってエントロピーの値が変化することがわかった．また，秘密情報の中から 2 種類の値を選ぶ組み合わせを全て検証した結果，どの 2 種類の値を選んで消去しても 3 個のシェアの組み合わせである 4 通りのうち，3 通りは 1.25, 残りの 1 通りは 1.36 となることが分かった．

最後に，秘密情報サイズが 6 の場合の結果を説明する．秘密サイズを 6 にするために 3 種類の秘密情報を消去した場合，エントロピーの値のとり方が 2 通りの場合に分かれた．

1 つ目の場合は任意の 3 個のシェアを選んでも，秘密情報が一意に定まらないように行を消去した．例として図 17 から秘密情報が 6 と 7 と 8 の行を消去した場合を考える．このとき，エントロピーの値は図 21 のようになった．図 21 の右のグラフは W_1, W_2, W_3 を選んだ場合のエントロピーのグラフで，左のグラフはそれ以外を選んだグラフである．この場合，W_1, W_2, W_3 を選んだときにエントロピーが確率的なランプ型の定義に基づいた値を超えるものとなった．

2 つ目の場合ではある 3 個のシェアを選んだ場合に秘密情報が一意に定まってしまうように行を消去した．例として図 17 から秘密情報が 0 と 5 と 8 の行を消去した場合を考える．このとき，エントロピーの値は図 22 のようになった．図 22 の右のグラフは W_1, W_2, W_3 を選んだ場合のエントロピーのグラフで，左のグラフはそれ以外を選んだグラフである．この場合，どの 3 個のシェアを選んでも確率的なランプ型の定義に基づいたエントロピーの値を満たすものができなかった．以上のことから秘密サイズを 6 にしようとした結果，行の消し方によって上記のような 2 通りの場合に分けられることが分かった．

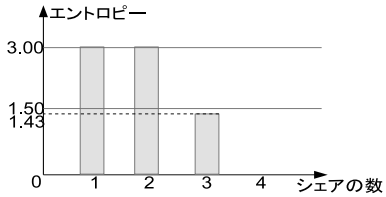


図 19　秘密情報サイズ:8, シェアサイズ:3 の場合

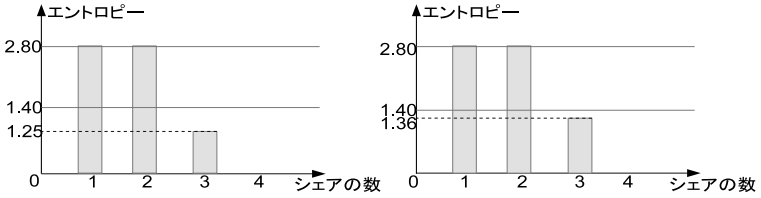


図 20　秘密情報サイズ:7, シェアサイズ:3 の場合

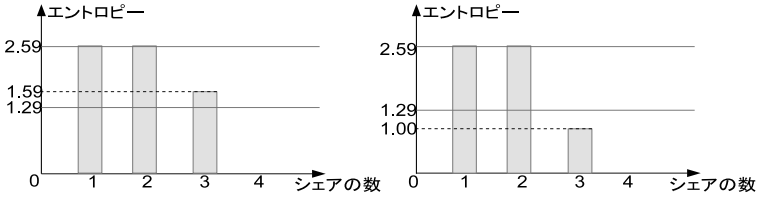


図 21　秘密情報サイズ:6, シェアサイズ:3 の場合 (1)

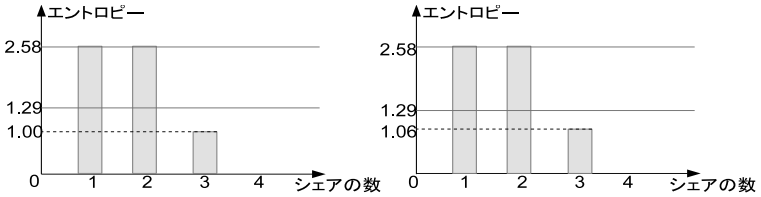


図 22　秘密情報サイズ:6, シェアサイズ:3 の場合 (2)

6.4　エントロピーの観点からの課題

本研究で検証した構成方法では任意の 3 個のシェアを集めた時の秘密情報のエントロピーが確率的なランプ型の定義に基づいた値を超えるものを構成できなかった．これは構成方法にお

いてオリジナルの復号表から行を削除したことにより，シェアの値の組合せによっては秘密情報の候補が少なくなることが原因である．また，秘密情報を消去する数によって 3 個シェアを集めた場合の秘密情報のエントロピーの値がシェアの選び方によって異なる場合があった．そのため，このような場合に対して何を持ってランプ型秘密分散法であるのかを定義する必要がある．今後の課題としては確率的なランプ型の定義に基づいたエントロピーの変化になるような構成方法が可能かどうかの検証や秘密情報サイズとシェアサイズが異なる場合のランプ型の定義についての検討があげられる．しかし，これらの課題の解決のためには様々な指標が必要である．そこで，その指標の 1 つとして，シェアの集め方によって変化するエントロピーに注目してみる．6.3 節の結果より，秘密サイズが 8 の時はエントロピーの値がどの 3 個のシェアを選んだときでも同じ値を取ったが，秘密サイズが 6,7 の時にはどの秘密情報を選んで消去しても 3 個のシェアの選び方によって秘密情報のエントロピーの値が異なっていた．このことから，図 17 の左図を用いて秘密サイズが 7 以下の復号表を作ろうとするとどの 3 個のシェアを選んだ場合でもエントロピーの値を一定にすることができないということを意味している．しかし，秘密サイズ 9, シェアサイズ 3 の (4,2,4) ランプ型の復号表は複数存在し，本研究では図 17 の復号表の一例のみを用いて検証したため，このことがすべての復号表に当てはまるかどうかは理論的には分からない．そこで本研究では，秘密サイズ 9, シェアサイズ 3 の (4,2,4) ランプ型の復号表から 2 種類以上の秘密を消去した場合に 3 個のシェアの選び方に関わらず秘密情報のエントロピーの値が一定となるような復号表が存在するかどうかを検証した．

6.5　シェアの分布の観点からの結果と考察

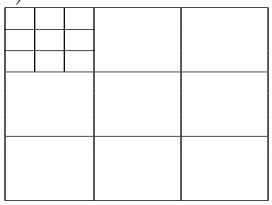
エントロピーの観点からの考察から，行の削除の仕方によって任意の 3 個のシェアを集めた時の秘密情報のエントロピーの値が異なることが分かった．そこで，どんな 3 個のシェアの集め方でも秘密情報のエントロピーの値を一定にするために，4 個のシェアの中からどの 3 個のシェアを選んでも秘密情報のエントロピーが一定になる例に焦点を当ててどのような法則があるかを考察した．その結果，4 個のシェアの中からどの 3 個のシェアを選んでも，秘密情報のエントロピーが同じ値になる例を作れるのは秘密情報のサイズが 8, シェアのサイズが 3 の場合のみであることが分かった．また，秘密情報のサイズが 7 以下の場合はどのような消し方をしても任意の 3 個のシェアを集めた場合の秘密情報のエントロピーの値が一定となるような例が作れないことが分かった．以下でその理由を説明する．

		$w_3,\ w_4$								
		0,0	0,1	0,2	1,0	1,1	1,2	2,0	2,1	2,2
w_1, w_2	0,0	0	1	2	3	4	5	6	7	8
	0,1	5	3	4	8	6	7	2	0	1
	0,2	7	8	6	1	2	0	4	5	3
	1,0	4	5	3	7	8	6	1	2	0
	1,1	6	7	8	0	1	2	3	4	5
	1,2	2	0	1	5	3	4	8	6	7
	2,0	8	6	7	2	0	1	5	3	4
	2,1	1	2	0	4	5	3	7	8	6
	2,2	3	4	5	6	7	8	0	1	2

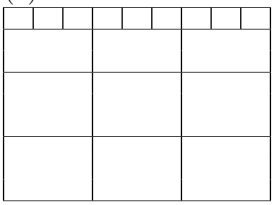
図 23　秘密サイズ 9, シェアサイズ 3 のランプ型の復号表の例

まず，図 17 の表現方法を図 23 のような別表現に変える．図 23 を 3×3 の区画ごとに区切ると特殊な制約を持った数独パズルとなる．その理由は元となった図 17 が持つ性質によるものである．図 17 のもつ性質はシェアを座標とする 4 列の中から任意の 2 列の値を固定しても 9 種類の文字が出現する．また，任意の 3 列の値を固定すると，3 種類の値が出現するという性質を持つ．これらの性質を数独パズルのようなルールとして図で表現すると図 24 の 6 パターンで表せる．また，これらのルールを用いることによって秘密サイズが 7 の場合にどの 3 個のシェアを集めても秘密情報のエントロピーの値が一定となるような復号表ができないことが論理的に示すことができる．その証明を次節で行う．

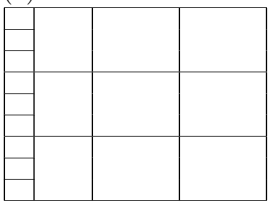
(1)1 区画中で 0〜8 が出現



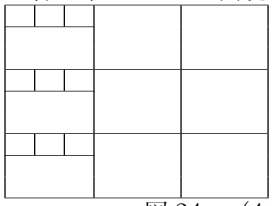
(2)1 行中で 0〜8 が出現



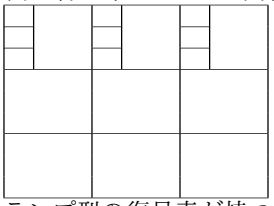
(3)1 列中で 0〜8 が出現



(4) 縦に並ぶ 3 区画での同じ行の中で 0〜8 が出現



(5) 横並びの 3 区画における同じ行の中で 0〜8 が出現



(6) 各区画における同じマスの中で 0〜8 が出現

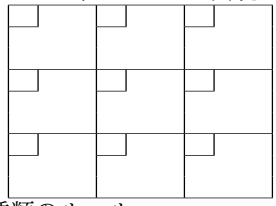


図 24　(4,2,4) ランプ型の復号表が持つ 6 種類のルール

6.6　秘密情報のエントロピーの変化を統一できないことの証明

証明の流れとしてはどの 3 個のシェアを集めても秘密情報のエントロピーの値が一定となるように秘密情報を消去できるような表が存在しないことを示す．図 25 のような 3 個のシェアを座標にした表を作成する．

次に図 25 の各々の表の左上の区画のマスと 4 座標の表におけるマスのそれぞれの対応関係を図 26,27 に示す．

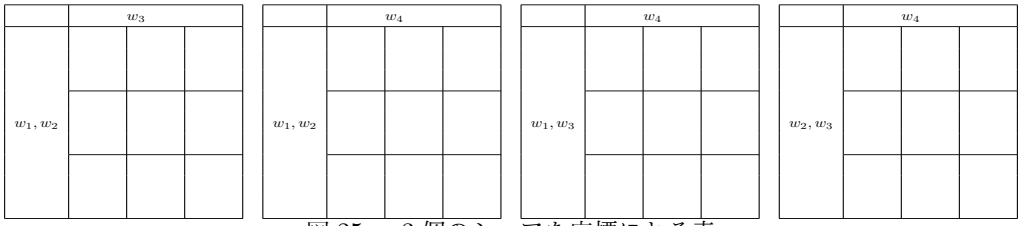


図 25　3 個のシェアを座標にとる表

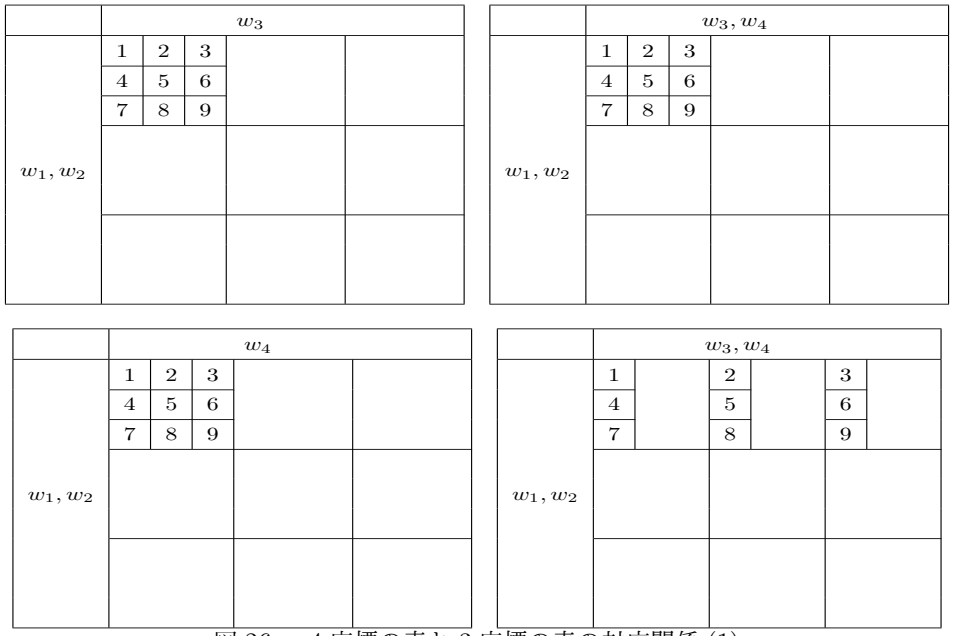


図 26　4 座標の表と 3 座標の表の対応関係 (1)

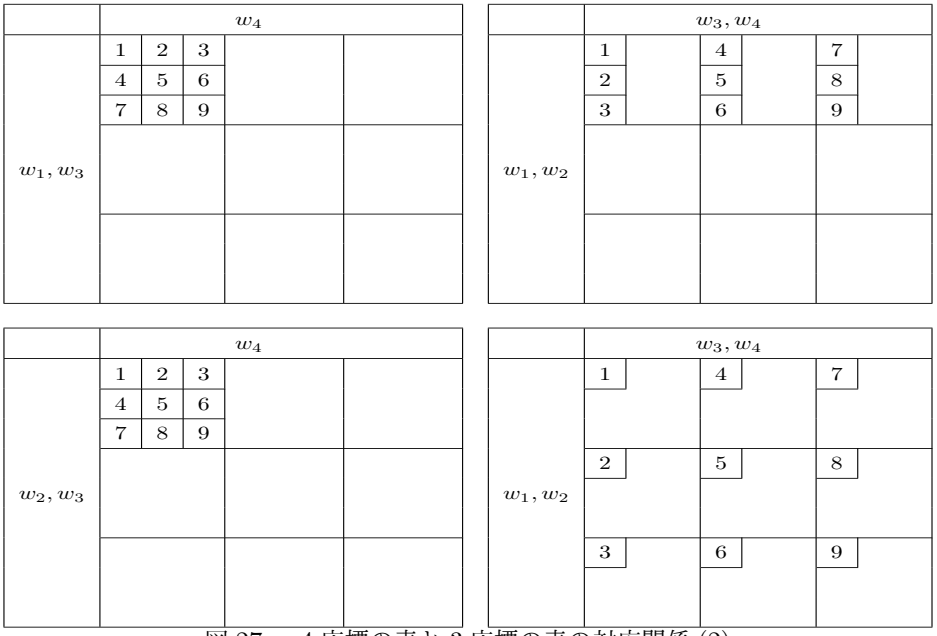


図 27　4 座標の表と 3 座標の表の対応関係 (2)

3 座標の表の左上の区画のマスは図 26,27 のように 4 座標の表の同じ番号のマスと対応関係にある．以上のことを用いて，秘密サイズ 7, シェアサイズ 3 の場合にどの 3 個のシェアを集めても秘密情報のエントロピーの値が一定になるような復号表が作れないことを示していく．それを示すためには，そのような復号表を作る過程で図 24 で示した 6 種類のルール全てを同時には満たせないことを示せばよい．

まず，左上の 1 区画目のみ記入された図 28 のような表を準備する．

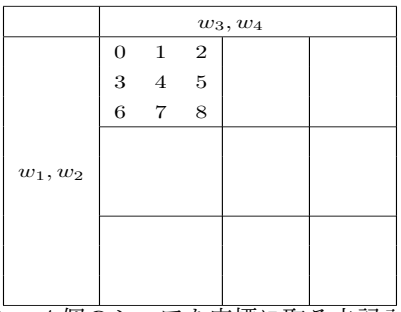


図 28　4 個のシェアを座標に取る未記入な表

秘密情報サイズを 7 にするためには図 28 の全体から 2 種類の秘密情報のマスを消去しなければならない．消去の仕方は大きく 2 通りに分けられる．

1 つ目の消去の仕方では「任意の 3 個のシェアを集めた場合に同区画中で消去するマスが同じ横列に存在しない」という条件を満たすように消去するマスを選ぶ．この条件を満たすよう

に消去する秘密情報を 0 と 8 に設定し，その 2 種類の値のあるマスを強調すると図 29 のような表になる．ここで，条件さえ満たせば消去したい秘密情報をどの 2 種類に設定しても良い．図 29 の右図は w_1, w_2, w_3 を座標とする表にその結果を反映したものである．

	w_3, w_4				
w_1, w_2	(0)	1	2		
	3	4	5		
	6	7	(8)		

	w_3				
w_1, w_2	(0)	1	2		
	3	4	5		
	6	7	(8)		

図 29　0 と 8 を消去した表

次に，w_1, w_2, w_4 を座標とする表において条件を満たすように図 28 の表から 8 の入る候補を探す．8 が入ってはいけないマスに対して×マークを記入すると，w_1, w_2, w_4 を座標とする表において条件を満たすためには図 30 のように×マークが入り，空白のある 2 マスが候補となる．

	w_3, w_4								
w_1, w_2	(0)			×			×		
	×								
	×		(8)	×			×		

図 30　マスの候補

また，図 24 のルール (5) により図 30 の残りの 2 マスのどちらかには 8 が入らなければいけない．そこで，図 31 のように 8 を入れたとする．

	w_3, w_4								
w_1, w_2	(0)			×			×		
	×			(8)					
	×		(8)	×			×		

図 31　現時点での復号表

次に，図 31 に対して図 24 のルール (2)(3) を適用してから，w_1, w_3, w_4 を座標とする表について考える．w_1, w_3, w_4 を座標とする表において条件を満たすためには図 32 のように×マークが入る．また，図 24 のルール (6) により図 32 の空白のマスのどちらかには 8 が入らなければいけない．

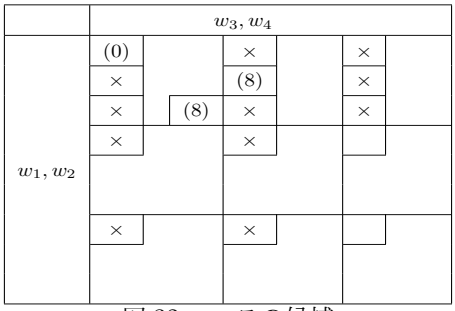


図 32　マスの候補

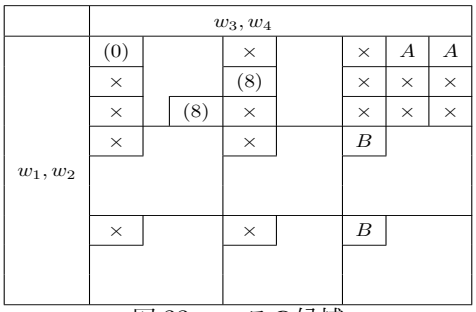


図 33　マスの候補

ここで，図 32 の右上の区画に対しても×マークを記入すると，図 33 のようになる．この図 33 に対して，4 個ある候補のマスを A, B に分ける．もし，A のどちらかのマスに 8 を入れた場合，図 24 のルール (4) の適用により図 34 の左の表のように×マークが入る．一方，B のどちらかのマスに 8 を入れた場合，図 24 のルール (4) の適用により図 34 の右の表のように×マークが入る．

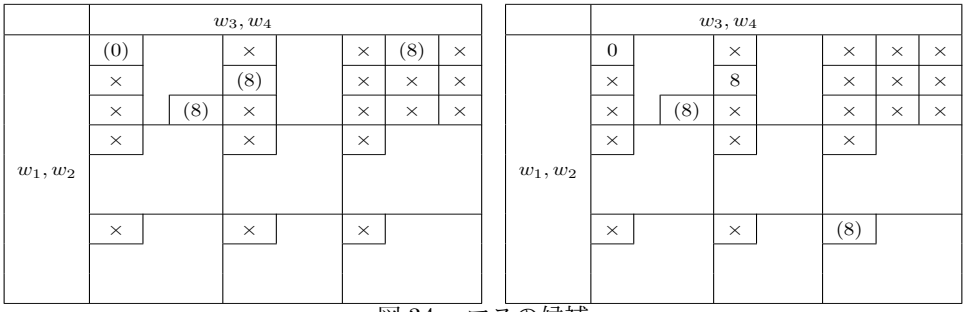


図 34　マスの候補

図 34 の左の表は図 24 のルール (6) を満たしておらず，右の表はルール (1) を満たしていない．つまり，この場合 (4,2,4) ランプ型の復号表が作れないことを意味する．また，この議論において候補が複数ある場合がいくつか存在したが，どの候補のマスを選んでも 6 種類のルールの適用により同じ結論が導き出される．そして，どの 2 種類の値を選んでも同様の流れを用いることで，6 種類のルールを満たすような復号表がつれないという結論が導き出される．つまり，作成したい表が存在しないこととなる．

2 つ目の消去の仕方では「同区画中で消去する秘密情報が同じ横列に並ぶ」という条件を満たすように消去するマスを選ぶ．この条件を満たすように，消したいマスを 0 と 1 に設定したとすると図 35 のような表になる．図 35 の右図は w_1, w_2, w_3 を座標とする表にその結果を反映したものである．

	w_3, w_4				
w_1, w_2	(0)	(1)	2		
	3	4	5		
	6	7	8		

	w_3				
w_1, w_2	(0)	(1)	2		
	3	4	5		
	6	7	8		

図 35　0 と 1 を消去した表

次に，w_1, w_2, w_4 を座標とする表において条件を満たすように図 35 の表から 1 の入る候補を探す．1 が入ってはいけないマスに対して×マークを記入すると，w_1, w_2, w_4 を座標とする表において条件を満たすためには図 36 のように×マークが入り，空白のある 2 マスが候補となる．ここで，条件さえ満たせば消去したい秘密情報をどの 2 種類に設定しても良い．

	w_3, w_4				
w_1, w_2	(0)	(1)	×		×
	×		×		×
	×		×		×

図 36　マスの候補

図 36 は横並びの 3 区画における同じ行の中で 1 が一度も存在しないので図 24 のルール (5) を満たしていない表である．つまり，(4,2,4) ランプ型の復号表を作れないことを意味する．従って，この場合もどの 3 個のシェアを選んでも秘密情報のエントロピーが同じ値になるよう

な復号表が存在しない．以上のことから秘密情報サイズ 7, シェアサイズ 3 の $(4, 2, 4)$ ランプ型において，どの 3 個のシェアを選んでも秘密情報のエントロピーが同じ値になるような復号表が存在しないことが分かった．

7 まとめ

確率的な秘密分散法と非確率的な秘密分散法の定義について確認し，複数個のラテン方陣が持つ性質である整合性と直交性を用いて (2,3) しきい値法が構成できることを指摘した．また，ラテン方陣の直交性を用いることによって，Shamir の多項式補間法の性質上構成することができないサイズ 10 の (2,3) しきい値法を構成できることがわかった．一方で文献 [7] における結果から，サイズ 2 と 6 の整合性を満たしたラテン方陣や直交ラテン方陣が存在しないことが分かり，サイズ 2 と 6 の (2,3) しきい値法を構成できないことが分かった．この結果を元に整合性と直交性が等価な概念であると推測して論理的に検証した結果，この 2 つの概念が等価であることを示すことができた．

また，直交配列を用いたランプ型秘密分散法に関する研究 [8] を発展させて秘密情報サイズとシェアサイズが異なるランプ型についても検討および考察を行った．秘密情報サイズとシェアサイズが互いに異なる場合，秘密情報サイズがシェアサイズのべき乗であるかどうかを区別し，それぞれの場合について構成方法の提案や秘密分散法としての安全性や復元性を定義するための指標などを探索した．その結果，べき乗である場合については直交配列や有限体を用いて構成できる例が確認できた．一方，べき乗でない場合についてはべき乗の場合に用いる復号表から行を消去して秘密サイズを調整する構成方法を検討した．また，この構成方法の安全性や性能をエントロピーやシェアの値の分布の観点から考察したが，本研究では確率的なランプ型の定義を満たすようなエントロピーを得ることができなかった．一方，ある特殊なランプ型において，シェアの選び方によらずに秘密情報のエントロピーを等しくできない理由を示すことができた．

今後の課題として，秘密情報サイズがシェアサイズのべき乗でない場合について別の構成方法の提案やそれに対する評価基準の設定，及び秘密分散法としての一般化のための様々な基準の定義が残っている．また，具体的に考えられる新たな構成方法としては秘密情報のサイズを調整する際に，何種類かの秘密情報の値を消去するのではなく別の値に置き換えるという方法が考えられる．

謝辞

この研究を進めるにあたり指導していただいた西新幹彦准教授に感謝する．

参考文献

[1] 山本博資,「秘密分散法とそのバリエーション」, 数理解析研究所講究録, 1361 巻, pp.19–31, 2004 年.

[2] Adi Shamir, “How to Share a Secret,” Communications of ACM, no.22, pp.612–613, Nov. 1979.

[3] 山本博資,「(k, L, n) しきい値秘密分散システム」, 電子情報通信学会論文誌，vol.J68-A, no.9, pp.945–952, 1985 年.

[4] 大草健人,「秘密分散法の非確率的な定義について」，信州大学工学部学士論文, 2015 年 3 月.

[5] M.Hall, 岩堀信子訳, 組合せ理論, 吉岡書店, 1971.

[6] A.S.Hedayat, N.J.A.Sloane, J.Stufken, *Orthogonal Arrays:Theory and Applications*, Springer, 1999.

[7] Solomon W.Golomb, “Latin Squares,” IEEE Information Theory Society Newsletter, June 2016.

[8] 西新幹彦,「強いランプ型秘密分散法は MDS 符号である」, 信学技報, IT2017-43, pp.27–30, 2017 年 9 月.

[9] 西新幹彦, 滝澤克則,「多項式補間法による強いランプ型しきい値秘密分散法」，電子情報通信学会論文誌，vol.J92-A, no.12, pp.1009–1013, 2009 年.

[10] Thomas M.Cover, Joy A.Thomas, *Elements of Information Theory*, Jan. 2006.

付録 A　ソースコード

A.1　サイズ 4 の $(2, 2, 3)$ ランプ型の探索プログラム

```
#include <stdio.h>
#include <stdlib.h>
#include <memory.h>

int
bit_count(unsigned int x0)
{
    unsigned int x1 = (x0 & 0x55555555) + ((x0 & 0xaaaaaaaa) >> 1);
    unsigned int x2 = (x1 & 0x33333333) + ((x1 & 0xcccccccc) >> 2);
    unsigned int x3 = (x2 & 0x0f0f0f0f) + ((x2 & 0xf0f0f0f0) >> 4);
    unsigned int x4 = (x3 & 0x00ff00ff) + ((x3 & 0xff00ff00) >> 8);
    unsigned int x5 = (x4 & 0x0000ffff) + ((x4 & 0xffff0000) >> 16);
    return(x5);
}

void
printarray(char *array)
{
    int i;

    convert(array);
    for (i = 0; i < 80; i++){
        printf("%2d", array[i]);
        if (i % 5 == 4){
            printf("\n");
        }
    }
    printf("\n");

    return;
}

int
convert(char *array)
{
    int i;

    for (i = 0; i < 80; i++){
        if (array[i] == 1){
            array[i] = 0;
        }
        if (array[i] == 2){
            array[i] = 1;
        }
        if (array[i] == 4){
            array[i] = 2;
        }
        if (array[i] == 8){
            array[i] = 3;
        }

    }

    return(0);
}

int current_column;

int
deduce(char (*array)[5], int idx)
{
```

```
    int value;
    int i, j;
    char current_cell = array[idx][current_column];

    for (i = 0; i < current_column ; i++){
        value = array[idx][i];
        for (j = 0; j < 16; j++){
            if (j == idx) continue;
            if (array[j][i] == value && (array[j][current_column] & current_cell)){
                array[j][current_column] &= ~current_cell;
                if (bit_count(array[j][current_column]) == 1){
                    deduce(array, j);
                }
            }

        }
    }

    return(0);
}

int
position(char (*array)[5])
{
    int num;
    int i;
    int nc = 5, idx;

    for (i = 0; i < 16; i++){
        int bc = bit_count(array[i][current_column]);
        if (bc == 0) return(0);
        if (bc == 1) continue;
        if (bc < nc){
            nc = bc;
            idx = i;
        }
    }
    if (nc == 5){
        if (current_column < 4){
            current_column++;
            num = position(array);
            current_column--;
            return(num);
        } else{
            printarray(array);
            exit(0);
            return(1);
        }
    }
    num = 0;
    //printf("nc=%d,idx=%d", nc, idx);
    //exit(0);
    //printf("nc=%d,idx=%d\n", nc, idx);
    //printarray(array);
    for (i = 0; i < 4; i++){
        char array_cp[16][5];
        if ((array[idx][current_column] & (1 << i)) == 0) continue;
        memcpy(array_cp, array, 80);
        array_cp[idx][current_column] = 1 << i;
        deduce(array_cp, idx);
        num += position(array_cp);
    }
    return(num);
}

int
main()
{
    int num;
```

```
    char array[16][5] = {0x1,0x1,0xf,0xf,0xf,
        0x1, 0x2, 0xf, 0xf, 0xf,
        0x1, 0x4, 0xf, 0xf, 0xf,
        0x1, 0x8, 0xf, 0xf, 0xf,
        0x2, 0x1, 0xf, 0xf, 0xf,
        0x2, 0x2, 0xf, 0xf, 0xf,
        0x2, 0x4, 0xf, 0xf, 0xf,
        0x2, 0x8, 0xf, 0xf, 0xf,
        0x4, 0x1, 0xf, 0xf, 0xf,
        0x4, 0x2, 0xf, 0xf, 0xf,
        0x4, 0x4, 0xf, 0xf, 0xf,
        0x4, 0x8, 0xf, 0xf, 0xf,
        0x8, 0x1, 0xf, 0xf, 0xf,
        0x8, 0x2, 0xf, 0xf, 0xf,
        0x8, 0x4, 0xf, 0xf, 0xf,
        0x8, 0x8, 0xf, 0xf, 0xf,
    };

    current_column = 2;
    num = position(array);
    printf("num: %d\n", num);
    return(0);
}
```