Vol.2022-DPS-190 No.37
Vol.2022-CSEC-96 No.37
2022/3/11

軽量Nパーティ秘匿関数計算のsemi-honest モデルにおける安全性の証明

滝 雄太郎[1]　藤田 茂[2]

概要：
本稿では軽量 N パーティ秘匿関数計算において計算主体間の結託は無いと仮定した上で semi-honest モデルにおける安全性の証明を示す．軽量 N パーティ秘匿関数計算とは軽量 3 パーティ秘匿関数計算について各計算プロトコルの一般化を行った方式である．軽量 3 パーティ秘匿関数計算は malicious モデルにおいてエラー検出を可能であること，semi-honest モデルにおいて入力値の安全性を保証することを証明している．しかし，軽量 N パーティ秘匿関数計算では malicious モデルにおいてエラー検出可能であることは示してあるが，semi-honest モデルにおいて安全性を保証することは証明していない．この安全性を保証することを証明する上で重要になるのが完全秘匿性の有無である．そこで，まず提案方式では完全秘匿性を持つことを証明する上で重要な，出力を除く受信データを示す VIEW について一般化を行う．この VIEW についての一般化により従来の証明よりも簡潔な記述方法で示すことが可能になった．本稿では軽量 N パーティ秘匿関数計算が semi-honest モデルにおいて完全秘匿性を持つことを証明した．

キーワード：秘密分散，秘密計算，セキュリティ，安全性

Proof of Security in Semi-honest Model of Lightweight N-party Secure Function Evaluation

Abstract: In this paper, we show the proof of safety in the semi-honest model on the assumption that there is no collusion between subjects in the calculation of lightweight N-party secure function evaluation. Lightweight N-party secure function evaluation is a method that generalizes each calculation protocol for lightweight three-party secure function evaluation. The lightweight three-party secure function evaluation proves that error detection is possible in the malicious model and that the safety of the input value is guaranteed in the semi-honest model. However, although the lightweight N-party secure function evaluation shows that error detection is possible in the malicious model, it does not prove that the safety is guaranteed in the semi-honest model. The presence or absence of perfect secrecy is important in proving that this safety is guaranteed. Therefore, in the proposed method, we first generalize the "VIEW" that shows the received data excluding the output, which is important for proving that it has perfect secrecy. This generalization of "VIEW" has made it possible to show in a simpler description method than the conventional proof. In this paper, we prove that the lightweight N-party secure function evaluation has perfect secrecy in the semi-honest model.

Keywords: Secret Sharing, Secure Computation, Security, Safety

1. はじめに

セキュリティやプライバシー保護等，情報の秘密を守ることが情報システムを構築する上で重要な課題である [1]．情報の秘密を守る手法には日常におけるセキュリティ対策やコンピュータウィルス対策，脆弱性対策等様々な物があ

[1] 千葉工業大学大学院
Graduate School of Information and Computer Science, Chiba Institute of Technology
[2] 千葉工業大学
Faculty of Information and Computer Science, Chiba Institute of Technology

る [2]．これらの対策を行うことがシステムの安全を保つことに繋がり，システム利用者の情報が適切に守られると言える．

ここで，システムを運用するサーバ・部門・団体といったものが単一であるとその一箇所にリスクが集中する．物理的なサーバの場所が一箇所の場合に災害によりシステムが止まる場合，ある団体にセキュリティインシデントが発生した場合に情報が流出してしまう場合など，このシステムの主体が一箇所の場合に発生しうる問題は多い．

この問題を解決するための手法の一つとして，秘密にしたい情報を複数箇所に分散し秘密を守る秘密分散法がある [3], [4], [5]．そして，秘密情報を秘密分散法により分散した上で計算を実行する秘密計算法・マルチパーティ計算・秘匿関数計算が提案されている [6], [7], [8]．これらの秘密分散・秘密計算法は安全性を様々な方法で確保，その正当性を証明している．

証明をする中で安全性を脅かす攻撃者のモデルとして semi-honest と malicious なモデルがある．semi-honest モデルは各主体は決められた計算プロトコルのみを行う．その上で，自分自身が取得した情報から最大限秘密情報の推定を行おうとする．malicious モデルは決められた計算プロトコルから逸脱した操作を行う．具体的には，嘘の情報を送信する，計算を途中で止める等どのような操作を行っても良い．malicious モデルは semi-honest モデルよりも強いモデルであるとも言える．

秘密計算法の一つに，軽量 3 パーティ秘匿関数計算がある [9]．この手法は semi-honest, malicious の両方のモデルで個々に安全性の証明が行われている．semi-honest モデルにおいては利用者からシステムに対して入力された秘密情報の安全性を保証することを証明している．malicious モデルにおいてはシステムから利用者への出力に対してエラー検出が可能であることを証明している．

また，軽量 N パーティ秘匿関数計算は軽量 3 パーティ秘匿関数計算について各計算プロトコルの一般化を行った手法である [10]．この手法は malicious モデルにおいて軽量 3 パーティ秘匿関数計算と同等のエラー検出が可能であることを証明している．しかし semi-honest モデルにおいての秘密情報の安全性を保証することを証明していない．

秘密情報の安全性を保証することで重要なのが完全秘匿性である．秘密計算法における完全秘匿性の有無の保証について証明を行う上で重要な，各主体で各計算プロトコルの際にやり取りされるデータの種類を示す．このデータの種類は，入力・出力・出力を除く受信データ，の 3 種類である．これらのデータのうち，完全秘匿性の有無の保証について重要になってくるのが出力を除く受信データである．この出力を除く受信データを VIEW と示す．

本稿では VIEW についての一般化を行う．これにより，従来の軽量 3 パーティ秘匿関数計算より簡潔な記述方法で semi-honest モデルにおける安全性の証明を可能とする．まず，2 章で軽量 N パーティ秘匿関数計算の各計算プロトコルの説明及び軽量 3 パーティ秘匿関数計算における VIEW の定義について述べる．次に，3 章で軽量 N パーティ秘匿関数計算における VIEW の性質について述べる．そして，4 章で軽量 N パーティ秘匿関数計算の安全性の証明を行う．最後に，5 章でまとめ及び今後の展望について述べる．

2. 関連研究

この章では，まず証明対象となる軽量 N パーティ秘匿関数計算の各計算プロトコルの説明について述べる．次に，一般化のもととなる軽量 3 パーティ秘匿関数計算における VIEW の定義について述べる．そして，以下の説明は素数 m を法とした有限体 $\mathbf{Z}/m\mathbf{Z}$ 上で行われるものとする．これにより，使用される値は 0 から $m-1$ までの値となる．ここで，n と k を特別な値として取り扱う．n は秘匿関数計算システムに参加する主体の総数であり，k は分散された秘密情報を復元するために必要な主体の数である．また，各主体に対してデータを纏めて送信・保管する場合にシェアという集合を利用する．シェアの表記は i 番目の主体 P_i に対してデータ a をもととした場合，$[a]_i := (a_i, \ldots, a_{n-k+i})$ となる．

2.1 軽量 N パーティ秘匿関数計算の各計算プロトコル

この節では，軽量 N パーティ秘匿関数計算の各計算プロトコルについて説明を行う．軽量 N パーティ秘匿関数計算には秘密分散・復元・加減算・定数倍・乗算・論理回路演算の演算がある．今回 VIEW を用いた semi-honest モデルにおける安全性を証明する演算は秘密分散，乗算の 2 つである．

秘密分散，乗算の演算以外の証明が必要ない理由としては，まず，復元については各主体にはデータが送信されることが無いからである．また，加減算・定数倍は各主体の中で演算が終わるからである．そして，論理演算については各演算を組み合わせた処理になるため，他の演算の安全性が証明されれば加えて安全性が証明されるからである．これは対象とする semi-honest モデルの攻撃者は他者と通信を行わない場合新たな情報を得られないためである．下記に，秘密分散，乗算，論理回路演算の各演算の処理を示す．

秘密分散は入力された任意の値 a を n 個に分割した後にを各主体 P_i に分散する演算である．なお，記述の簡略化のため分割された a の添字については主体の数 n の剰余を取ったものとする．

Share： k-out-of-n 秘密分散

入力：$a \in \mathbf{Z}/m\mathbf{Z}$
出力：$P_i([a]_i)$
(1) $a_0, \ldots, a_{n-1} \in \mathbf{Z}/m\mathbf{Z}$ をランダムに選択する．
(2) $a_{n-1} := a - \sum_{i=0}^{n-2} a_i$ を計算する．
(3) $i = 0, \ldots n-1$ について，$[a]_i := (a_i, \ldots, a_{n-k+i})$ として P_i に送信する．

乗算処理については処理部分についてのみ示す．各処理中の a_{row} b_{col} といった表記の意味合いや，“必要とする主体に送信する” の必要とする主体の決め方等はここに記載されている．

Mul： a, b のシェアから ab のシェアを作成

入力：$P_i([a]_i, [b]_i)$
出力：$P_i([ab]_i)$
(1) P_0 の操作
  (a) $r_1, \ldots, r_{n-1}, c_0, \ldots, c_{n-k-1} \in \mathbf{Z}/m\mathbf{Z}$ をランダムに選択する．
  (b) $c_{n-k} := a_{row}b_{col} + a_{col}b_{row} + \cdots + \sum_{i=0}^{n-k} a_i b_i - \sum_{i=1}^{n-1} r_i - \sum_{i=0}^{n-k-1} c_i$ を計算する．
  (c) $(r_i, c_i, \ldots c_{n-k})$ を必要とする主体に送信する．
  (d) $[ab]_0 := (c_0, \ldots, c_{n-k})$ とする．
(2) P_i の操作
  (a) P_i は $[a]_i, [b]_i, r_i$ を用いて $S_i := a_{row}b_{col} + a_{col}b_{row} + \cdots + r_i$ を計算する．
  (b) P_{2i-1}, P_{2i} 同士で S_i を送信し合う．
  (c) P_{2i-1}, P_{2i} は $c_{n-k+i} := S_{2i-1} + S_{2i} + a_{n-k+i}b_{n-k+i}$ を計算する．
  (d) c_{n-k+i} を必要とする主体に送信する．
  (e) $[ab]_i := (c_i, \ldots, c_{n-k+i})$ とする．

Not, **And**, **Or**, **Xor** の各論理演算については，加減算・定数倍・乗算の演算を組み合わせた処理になっている．

Not： $a \in \{0, 1\}$ のシェアから $\bar{a}$ のシェアを作成

入力：$P_i([a]_i)$
出力：$P_i([\bar{a}]_i)$
- 各パーティは $\bar{a_0} := 1 - a_0, \bar{a_i} := a_i$ を計算し $[\bar{a}]_i$ とする．

And:a, b のシェアから $a \wedge b$ のシェアを作成

入力：$P_i([a]_i, [b]_i)$
出力：$P_i([a \wedge b]_i)$
- P_i は $\mathbf{Sub}(\mathbf{Mul}([a]_i, [b]_i))$ を実行する．

Or:a, b のシェアから $a \vee b$ のシェアを作成

入力：$P_i([a]_i, [b]_i)$
出力：$P_i([a \vee b]_i)$
- P_i は $\mathbf{Sub}(\mathbf{Add}([a]_i, [b]_i), \mathbf{Mul}([a]_i, [b]_i))$ を実行する．

Xor:a, b のシェアから $a \oplus b$ のシェアを作成

入力：$P_i([a]_i, [b]_i)$
出力：$P_i([a \oplus b]_i)$
- P_i は $\mathbf{Sub}(\mathbf{Add}([a]_i, [b]_i), \mathbf{CoMul}(\mathbf{Mul}([a]_i, [b]_i), 2))$ を実行する．

本稿ではこれらの演算について安全性の証明を行う．

2.2 軽量 3 パーティ秘匿関数計算における VIEW の定義

この節では，軽量 3 パーティ秘匿関数計算における VIEW の定義について述べる．軽量 3 パーティ秘匿関数計算における VIEW の定義及び安全性の証明としては，完全秘匿性についての定義，秘密計算における完全秘匿性についての定義，各主体が結託しない場合の完全秘匿性の定義の順番で成り立っている．下記にこの定義について示す．

定義 2.1 中の $Pr()$ は確率関数を示す．また，S が秘密のデータであり $F(S)$ が開示されるデータである．

定義 2.1. 集合 $\mathcal{S}$ 上の確率変数 S と独立であるような，$\mathcal{S}$ から集合 $\mathcal{D}$ への確率的関数 F が完全秘匿性を持つとは，任意の実際の秘密のデータ $s \in \mathcal{S}$ と出力 $d \in \mathcal{D}$ に対して，

$$Pr(S = s | F(S) = d) = Pr(S = s)$$

となることをいう．

次に秘密計算における完全秘匿性の定義について示す．また，x は任意の入力であり I は主体の集合である．

定義 2.2. j 番目の主体に関するデータが以下であるような n パーティプロトコルを考える．
- 入力：x_j
- 出力：$f_j(x_1, \ldots, x_n)$
- 出力を除く受信データ：$\mathrm{VIEW}_j(x_1, \ldots, x_n)$

このプロトコルが *semi-honest* モデルで完全秘匿性を持つとは，*semi-honest* モデルにおける攻撃者に対して以下のような確率的関数 S が存在することである．

任意の入力 $\vec{x} = \{x_1, \ldots, x_n\}, I = \{i_1, \ldots, i_{|I|}\} \subseteq \{1, \ldots, n\}$ に対して $S(I, \vec{x}_I, f_I(\vec{x}))$ と $\mathrm{VIEW}_I(\vec{x})$ の確率分布は等しい．

ただし，$\vec{x} = \{x_{i_1}, \ldots, x_{i_{|I|}}\}, f_I = (f_{i_1}, \ldots, f_{i_{|I|}}), \mathrm{VIEW}_I = (\mathrm{VIEW}_{i_1}, \ldots, \mathrm{VIEW}_{i_{|I|}})$ である．

ここで主体の集合である I が 2 以上の場合は複数の主体が結託した場合を示す．このため，この定義では主体間

の結託が発生した場合に安全であるかどうかの定義となっている．しかし，軽量 3 パーティ秘匿関数計算では各主体が結託しない場合を仮定している．そこで，最後に軽量 3 パーティ秘匿関数計算で仮定している主体間が結託しない場合の完全秘匿性の定義を示す．

定義 2.3. 定義 2.2 と同等のプロトコルを考え，$I = \{i_1, \ldots, i_t\} \subseteq \{1, \ldots, N\}$ とする．このプロトコルが *semi-honest* モデルで主体集合 I に対して完全秘匿性を持つとは．*semi-honest* モデルにおける攻撃者に対して以下のような確率的関数 S が存在することである．

任意の入力 $\vec{x} = \{x_1, \ldots, x_N\}$ に対して $S(I, \vec{x_I}, f_I(\vec{x}))$ と $\mathrm{VIEW}_I(\vec{x})$ の確率分布は等しい．

定義 2.3 をふまえて今回の証明で用いる十分条件を下記に示す．

命題 2.4. 定義 2.2 と同等のプロトコルを考え，$I = \{i_1, \ldots, i_t\} \subseteq \{1, \ldots, N\}$ とする．このとき，$\mathrm{VIEW}_I(\vec{x})$ が一様乱数ならば，プロトコルは *semi-honest* モデルで全体集合 I に対して完全秘匿性を持つ．

Proof. 定義中，S をつねに一様乱数を出力する関数とすればよい． □

命題 2.4 より VIEW がすべて一様乱数となっていれば，計算主体間の結託は無いと仮定した上で semi-honest モデルにおいて安全性が証明される．

3. 軽量 N パーティ秘匿関数計算における VIEW の性質

命題 2.4 より VIEW がすべて一様乱数となっていれば，計算主体間の結託は無いと仮定した上で semi-honest モデルにおいて安全性が証明されることが示された．ここで，軽量 N パーティ秘匿関数計算における VIEW の性質について述べる．

軽量 3 パーティ秘匿関数計算及び軽量 N パーティ秘匿関数計算は素数 m を法とした有限体 $\mathbf{Z}/m\mathbf{Z}$ 上で行われている．この有限体上での演算についての各項における確率変数について以下のような定義される．

定義 3.1. 素数 m を法とした有限体 $\mathbf{Z}/m\mathbf{Z}$ 上において互いに独立な一様乱数を加減乗除した確率変数の結果は常に一様乱数である．

定義 3.1 より VIEW について以下の条件が言える．

命題 3.2. 軽量 N パーティ秘匿関数計算の各計算プロトコルの処理において，各項の確率変数を示した上でその中に一様乱数が含まれていれば結果を一様乱数としてみなしても良い．

Proof. 定義 3.1 より，軽量 N パーティ秘匿関数計算は素数 m を法とした有限体 $\mathbf{Z}/m\mathbf{Z}$ 上で行われている．したがって確率変数の結果は一様乱数となる． □

命題 3.2 より軽量 N パーティ秘匿関数計算の安全性の証明を行う際には各計算プロトコルの各項の VIEW を示した上でその中に一様乱数が含まれているかどうか確認を行い，処理手順の結果がすべて一様乱数になっているか確認を行えば良い．

4. 軽量 N パーティ秘匿関数計算の安全性の証明

この章では，軽量 N パーティ秘匿関数計算の安全性の証明を行う．ここで，各項の確率変数をもとの項を x_i とした時に V^{x_i} とする．このとき，各手順における V^{x_i} についてどのような理由で一様乱数であるかについて述べる．この繰り返しにより，前後の証明との対応をより明確にする．結果として証明としての記述量は増加するが，各処理の項とその確率変数の対応が明確になり結果として簡潔に証明を記述することが可能となる．

4.1 秘密分散の安全性の証明

下記に秘密分散の処理の手順を各項を確率変数に書き換えた上で示す．

Share : k-out-of-n 秘密分散

入力：$a \in \mathbf{Z}/m\mathbf{Z}$
出力：$P_i([a]_i)$
(1) $V^{a_0}, \ldots, V^{a_{n-1}} \in \mathbf{Z}/m\mathbf{Z}$ をランダムに選択する．
(2) $V^{a_{n-1}} := a - \sum_{i=0}^{n-2} V^{a_i}$ を計算する．
(3) $i = 0, \ldots n-1$ について，$[a]_i := (V^{a_i}, \ldots, V^{a_{n-k+i}})$ として P_i に送信する．

補題 4.1. **Share** において VIEW はすべて一様乱数である．

Proof. 処理の各手順について VIEW が一様乱数であるか確認を行う．
(1) $V^{a_0}, \ldots, V^{a_{n-1}}$ は一様乱数．
(2) $\sum_{i=0}^{n-2} V^{a_i}$ は (1) より一様乱数．$V^{a_{n-1}}$ は命題 3.2 より一様乱数．
(3) $i = 0, \ldots n-1$ について V^{a_i} は (1) より一様乱数．
(1-3) より **Share** において VIEW はすべて一様乱数である． □

4.2 乗算の安全性の証明

下記に秘密分散の処理の手順を各項を確率変数に書き換えた上で示す．

Mul： a, b のシェアから ab のシェアを作成

入力：$P_i([a]_i, [b]_i)$
出力：$P_i([ab]_i)$
(1) P_0 の操作
(a) $V^{r_1}, \ldots, V^{r_{n-1}}, V^{c_0}, \ldots, V^{c_{n-k-1}} \in \mathbf{Z}/m\mathbf{Z}$ をランダムに選択する．
(b) $V^{c_{n-k}} := V^{a_{row}}V^{b_{col}} + V^{a_{col}}V^{b_{row}} + \cdots + \sum_{i=0}^{n-k} V^{a_i}V^{b_i} - \sum_{i=1}^{n-1} V^{r_i} - \sum_{i=0}^{n-k-1} V^{c_i}$ を計算する．
(c) $(V^{r_i}, V^{c_i}, \ldots V^{c_{n-k}})$ を必要とする主体に送信する．
(d) $[ab]_0 := (V^{c_0}, \ldots, V^{c_{n-k}})$ とする．
(2) P_i の操作
(a) P_i は $[a]_i, [b]_i, V^{r_i}$ を用いて $V^{S_i} := V^{a_{row}}V^{b_{col}} + V^{a_{col}}V^{b_{row}} + \cdots + V^{r_i}$ を計算する．
(b) P_{2i-1}, P_{2i} 同士で V^{S_i} を送信し合う．
(c) P_{2i-1}, P_{2i} は $V^{c_{n-k+i}} := V^{S_{2i-1}} + V^{S_{2i}} + V^{a_{n-k+i}}V^{b_{n-k+i}}$ を計算する．
(d) $V^{c_{n-k+i}}$ を必要とする主体に送信する．
(e) $[ab]_i := (V^{c_i}, \ldots, V^{c_{n-k+i}})$ とする．

補題 4.2. **Mul** において VIEW はすべて一様乱数である．

Proof. 処理の各手順について VIEW が一様乱数であるか確認を行う．
(1) P_0 の操作
(a) $V^{r_1}, \ldots, V^{r_{n-1}}, V^{c_0}, \ldots, V^{c_{n-k-1}}$ は一様乱数．
(b) $V^{a_{row}}V^{b_{col}} + V^{a_{col}}V^{b_{row}} + \cdots + \sum_{i=0}^{n-k} V^{a_i}V^{b_i}$ は **Share** より一様乱数．$\sum_{i=1}^{n-1} V^{r_i} - \sum_{i=0}^{n-k-1} V^{c_i}$ は (1a) より一様乱数．$V^{c_{n-k}}$ は命題 3.2 より一様乱数．
(c) (1a,1b) より $(V^{r_i}, V^{c_i}, \ldots V^{c_{n-k}})$ は一様乱数．
(d) (1a,1b) より $(V^{c_0}, \ldots, V^{c_{n-k}})$ は一様乱数．
(2) P_i の操作
(a) $V^{a_{row}}V^{b_{col}} + V^{a_{col}}V^{b_{row}}$ は **Share** より一様乱数．V^{r_i} は (1a) より一様乱数．V^{S_i} は命題 3.2 より一様乱数．
(b) V^{S_i} は (2a) より一様乱数．
(c) $V^{a_{n-k+i}}V^{b_{n-k+i}}$ は **Share** より一様乱数．$V^{S_{2i-1}} + V^{S_{2i}}$ は (2a) より一様乱数．$V^{c_{n-k+i}}$ は命題 3.2 より一様乱数．
(d) $V^{c_{n-k+i}}$ は (2c) より一様乱数．
(e) $V^{c_i}, \ldots, V^{c_{n-k+i}})$ は (2d) より一様乱数．
(1a-2e) より **Mul** において VIEW はすべて一様乱数である．□

4.3 論理演算の安全性の証明

論理演算については **And**，**Or**，**Xor** が処理の中で **Mul** を必要としている．を利用している．これについて安全性の証明を行う．

補題 4.3. **And**，**Or**，**Xor** において VIEW はすべて一様乱数である．

Proof. 各プロトコルにおいて **Add**，**Sub**，**CoMul** については VIEW の変動は無い．**Mul** について命題 3.2 より一様乱数．したがって，**And**，**Or**，**Xor** において VIEW はすべて一様乱数である．□

5. おわりに

本稿では軽量 N パーティ秘匿関数計算において計算主体間の結託は無いと仮定した上で semi-honest モデルにおける安全性の証明を示した．軽量 N パーティ秘匿関数計算の元となった軽量 3 パーティ秘匿関数計算では計算主体間の結託は無いと仮定した semi-honest モデルにおいて安全性を証明していた．また，軽量 N パーティ秘匿関数計算では malicious モデルにおいてエラー検出が可能であった．しかし，軽量 N パーテイ秘匿関数計算は semi-honest モデルにおいて元と同等の安全性があることを証明していなかった．そこで，本稿では各計算プロトコルにおいて出力を除く受信データを示す VIEW について一般化を行い，より簡潔な記述方法で証明を示すことが可能となった．

今後の展望としては本稿の安全性の証明を他の秘密計算法に適応し，証明が可能であるか検証を行う．その上で，より汎用的な秘密計算法の semi-honest モデルにおける証明方法として本稿の手順を改善する．

参考文献

[1] 総務省：情報セキュリティ対策の必要性，https://www.soumu.go.jp/main_sosiki/joho_tsusin/security/business/executive/01.html. (Accessed on 10/28/2017).
[2] IPA 独立行政法人情報処理推進機構：情報セキュリティ，https://www.ipa.go.jp/security/. (Accessed on 01/29/2020).
[3] Shamir, A.: How to share a secret, *Communications of the ACM*, Vol. 22, No. 11, pp. 612–613 (1979).
[4] 岩村惠市，辻下健太郎，山根将司：サーバ資源を削減可能なパスワード付秘密分散法，電子情報通信学会論文誌 D， Vol. 102, No. 11, pp. 740–749 (2019).
[5] 菊池亮，千田浩司，濱田浩気，五十嵐大：Fast Large-Scale Honest-Majority MPC for Malicious Adversaries, 日本セキュリティ・マネジメント学会誌， Vol. 33, No. 1, pp. 49–50 (2019).
[6] 落合将吾，岩村惠市：n ¡ 2k - 1 において malicious な攻撃者に対しても安全な秘密分散を用いた秘匿計算とその拡張，情報処理学会論文誌，Vol. 62, No. 9, pp. 1457–1475 (2021).
[7] Lu, W., Kawasaki, S. and Sakuma, J.: Using Fully Homomorphic Encryption for Statistical Analysis of Categorical, Ordinal and Numerical Data., *IACR Cryptology*

ePrint Archive, Vol. 2016, p. 1163 (2016).
[8] 天田拓磨，奈良成泰，西出隆志，　吉浦裕ほか：通信量を削減した浮動小数点演算のためのマルチパーティ計算，情報処理学会論文誌，Vol. 60, No. 9, pp. 1433–1447 (2019).
[9] 千田浩司，五十嵐大，濱田浩気，高橋克巳：エラー検出可能な軽量 3 パーティ秘匿関数計算の提案と実装評価，情報処理学会論文誌，Vol. 52, No. 9, pp. 2674–2685 (2011).
[10] 滝雄太郎，藤田　茂，宮西洋太郎，白鳥則郎：軽量 N パーティ秘匿関数計算の一般化，情報処理学会論文誌，Vol. 59, No. 10, pp. 1895–1902 (2018).