準同型暗号を用いたプライバシ保護型条件付き集計プロトコル

Hayashi, Hiroki / 林, 宏樹

(出版者 / Publisher)
法政大学大学院情報科学研究科
(雑誌名 / Journal or Publication Title)
法政大学大学院紀要. 情報科学研究科編 / 法政大学大学院紀要. 情報科学研究科編
(巻 / Volume)
10
(開始ページ / Start Page)
1
(終了ページ / End Page)
6
(発行年 / Year)
2015-03-24
(URL)
https://doi.org/10.15002/00011707

準同型暗号を用いたプライバシ保護型条件付き集計プロトコル
Protocols for Evaluating Conditional Sum on Encrypted Data

林　宏樹
Hiroki Hayashi
法政大学大学院情報科学研究科
Email: hiroki.hayashi.7s@stu.hosei.ac.jp

Abstract—In this paper, we propose a protocol to evaluate conditional sum on encrypted data. Here, conditional sum is a sort of computation which takes pair of data $(a_i, d_i) \in A \times D$ $(i = 1, 2, ...)$ and a condition $\tilde{A} \subseteq A$ as input and output summation of d_is such that corresponding a_i satisfies $a_i \in \tilde{A}$ (i.e., $\sum_{a_i \in \tilde{A}} d_i$.) Such a computation is used in statistical analysis of large scale data. For example, it is used to analyze a total medical cost of employee with their 50s, 40s and 30s.

We construct protocols based on additive homomorphic encryption and 2-DNF cryptosystems, respectively, where 2-DNF cryptosystem is a homomorphic cryptosystem which allows arbitrary number of addition and single multiplication on encrypted data. The size of ciphertext for (a_i, d_i) of the proposed schemes protocols on additive homomorphic encryption and 2-DNF encryption are $O(n)$ and $O(\sqrt{n})$, respectively, where n denotes the cardinality of the set A.

We also proposed protocols for more complex conditions based on 2-DNF cryptsystems and Somewhat Homomorphic Encryption. These protocols are able to calculate conditional sum which takes the data (a_i, b_i, d_i) and conditon $\tilde{A}, \tilde{B}$ as input, and output $\sum_{a_i \in \tilde{A} \wedge b_i \in \tilde{B}} d_i$ or $\sum_{a_i \in \tilde{A} \vee b_i \in \tilde{B}} d_i$.

1　研究背景

クラウドコンピューティングは，従来ユーザーが手元のコンピュータで管理，利用していたソフトウェアやデータなどをネットワークを介してコンピュータ処理を行うことができるサービスである．ユーザーはローカルな場所にデータを保持する必要がなく，ネットワーク環境さえあればどこからでもデータにアクセスすることが可能となる．クラウドを活用することで，従来のデータベース管理システムなどでは扱うことのできなかった巨大なデータ (ビッグデータ) の集約，解析が可能となり，現在様々なビジネス，サービスに活用されている．

しかし，クラウドには利便性の反面，ネットワーク上でデータを扱うことに起因する情報漏洩のリスクがあり，ビッグデータをクラウド上で扱うにあたってプライバシーの保護という問題が常に伴う．データを暗号化して保存することでプライバシーを保つことが可能になるが，暗号化された状態ではデータの解析を行うことができない．解析を行う際には暗号化されたデータを復号する必要があり，復号されたデータはサーバの管理者や，その通信を盗聴している悪意のある第三者に漏洩する危険性が考えられる．この問題に対処するため，プライバシ保護型データマイニング (Privacy Preserving Data Mining:PPDM) と呼ばれる新しい技術が開発されている．PPDM は，データを秘匿にした状態で格納し，それらのデータを集積してデータマイニング手法を適用したのと同等の結果を得るための手法である．PPDM における主要な技術として，準同型暗号がある．準同型暗号は，データを暗号化した状態で復号を行うことなく様々な情報処理を行う手法である．準同型暗号を利用した PPDM としては，秘匿共通積集合計算プロトコルやプライバシー保護型協調フィルタリング等がある [7][1].

本研究では，準同型暗号を用いた条件付き集計を行うプロトコルについての提案を行う．条件付き集計とは，データ対集合 $(a_i, d_i) \in A \times D$ $(i = 1, 2, ...)$ 及び条件 $\tilde{A} \subseteq A$ を入力とし，$a_i \in \tilde{A}$ となる a_i に対応する d_i の合計を集計する集計法である．条件付き集計により，(年齢，医療費) のデータ対から特定の年代の医療費の合計を取り出すといった大規模データの統計的分析を行うことができる．本論文では，暗号化した状態で条件付き集計を行う 3 つのプロトコルと複数の変数を条件に含むより複雑な集計プロトコルの提案を行う．

2　準備

2.1　条件付き集計

条件付き集計とは，データ対集合 $(a_i, d_i) \in A \times D (i = 1, 2, ...)$ 及び条件 $\tilde{A} \subseteq A$ を入力とし，$a_i \in \tilde{A}$ となる a_i に対応する d_i の合計を集計する集計法である．条件付き集計により，(年齢，医療費) のデータ対から特定の年代の医療費の合計を取り出すといった大規模データの統計的分析を行うことができる．本研究の目的は，条件付き集計を暗号化した状態で行うことである．すなわち，(a_i, d_i) $(i = 1, 2, ...)$ を入力として，$\mathsf{Enc}_{pk}(\sum_{a_i \in \tilde{A}} d_i)$ を出力することである．

2.2　公開鍵暗号

公開鍵暗号は 1976 年に Diffie と Hellman により提案された概念であり，受信者が暗号化するための鍵 (公開鍵) と復合するための鍵 (秘密鍵) を生成，公開し，これらの鍵を用いて暗号文の生成・復号を行う暗号方式である [5]．公開鍵暗号は以下のアルゴリズム $(\mathsf{Gen}, \mathsf{Enc}_{pk}, \mathsf{Dec}_{sk})$ からなっている．

Gen:1^k(k:セキュリティパラメータ) を入力とし，公開鍵と秘密鍵のペアを出力するアルゴリズム．このとき平文空間 M_{pk} が定まる．

Enc_{pk}:公開鍵 pk と平文 $m \in M_{pk}$ を入力とし，暗号文 c を出力するアルゴリズム．

Dec_{sk}:公開鍵 pk と秘密鍵 sk と暗号文 c を入力とし，平文 m を出力するアルゴリズム．

セキュリティパラメータは，公開鍵や秘密鍵のサイズを決定するものであり，$k = 1024$ ならば鍵生成アルゴリズム Gen は 1024bit の鍵を生成する．このとき Gen は確率的なアルゴリズムであり，同じセキュリティパラメータが入力されても毎回そ

のアルゴリズム内の独立的な乱数に基づく値を出力する．また，復号は多くの場合確定的なアルゴリズムである．代表的な公開鍵暗号としては，RSA 暗号や ElGamal 暗号などがある．

2.3 準同型暗号

準同型暗号とは，公開鍵暗号方式が以下の性質を満たすことである．

1) 平文空間 M_{pk} と暗号文空間 C が代数的構造をなす．平文空間，暗号文空間上の演算子をそれぞれ $*$,$\star$ とする．
2) 任意の $m \in M_{pk}$ に対し，$\mathsf{Enc}_{pk}(pk, m) \in C$ である．
3) 任意の $m_1, m_2 \in M_{pk}$ と $c_1, c_2 \in C$，ただし $c_1 = \mathsf{Enc}_{pk}(pk.m_1)$ かつ $c_2 = \mathsf{Enc}_{pk}(pk, m_2)$，に対し $c_1 * c_2 = \mathsf{Enc}_{pk}(pk, m_1 \star m_2)$ が成立する．

準同型暗号を用いることで，暗号化したデータを復号を経ずに加算や乗算といった情報処理を行うことが可能となる．理論上はどのような処理でも可能となる完全準同型暗号 (Fully Homomorphic Encryption:FHE) が 2009 年に Gentry によって提唱されている [8]．しかし，FHE は通常のデータ処理と比べ計算効率が悪く，またその計算法についても十分な研究が為されていない．

そのため，機能を限定して実装することで現実的な実行速度を実現した部分準同型暗号が提案されている．提案方式では，加法準同型性及び 2-DNF 準同型性を用いてプロトコルを構成する．

2.4 2-DNF 暗号

2-DNF(BGN) 暗号は，2005 年に Bonne, Goh, Nissim によって提唱されたペアリング写像を利用した準同型暗号であり，一回の乗算と任意回の加算を行うことができる [2]．ペアリング写像は以下の性質を持つ．演算 $*$ が定義された巡回群 $\mathbb{G}_1$ (単位元を e_1 とする)，演算 $\cdot$ が定義された巡回群 $\mathbb{G}_2$(単位元を e_2 とする)，演算 $\odot$ が定義された巡回群 $\mathbb{G}_T$(単位元は e_T とする) に対し写像 $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ が双線型性，および非退化なる以下の 2 つの性質を満たす時，この写像 e をペアリング写像と呼ぶ．

- 双線形性
  任意の $u, u_1, u_2 \in \mathbb{G}_1, v, v_1, v_2 \in \mathbb{G}_2$ に対し以下の 2 つが成り立つ．
  $e(u_1 * u_2, v) = e(u_1, v) \odot e(u_2, v)$
  $e(u, v_1 \cdot v_2) = e(u, v_1) \odot e(u, v_2)$
- 非退化
  任意の $v \in \mathbb{G}_2$ に対し $e(u, v) = e_T (u \in \mathbb{G}_1)$ であるならば， $u = e_1$ である．
  任意の $u \in \mathbb{G}_1$ に対し $e(u, v) = e_T (v \in \mathbb{G}_2)$ であるならば， $v = e_2$ である．

ペアリング写像の定義から次の関係が導かれる．任意の整数 $a, b \in Z$ に対し次式が成立する．

$$e(u^a, v^b) = e(u^b, v^a) = e(u, v)^{ab}$$

また，巡回群 $\mathbb{G}_2$ から $\mathbb{G}_1$ への写像 $\phi : \mathbb{G}_2 \to \mathbb{G}_1$ が同型写像であるとき，任意の $u, v \in \mathbb{G}_2$ に対し次式が成立する．

$$e(\varphi(u), v) = e(\varphi(v), u)$$

以上の定義を用いて，2-DNF 暗号のアルゴリズムは以下のように定義される．

1) Gen: セキュリティパラメータ $\tau \in \mathbb{Z}^+$ を入力として以下のアルゴリズム $\mathcal{G}(\tau)$ を用いて $(q_1, q_2, \mathbb{G}, \mathbb{G}_1, e)$ を出力する．
   a) τ-bit の 2 つのランダムな素数 q_1, q_2 を生成し，$n = q_1 q_2 \in \mathbb{Z}$ を設定する．
   b) 位数 n の双線型群 $\mathbb{G}$ を生成する．g は $\mathbb{G}$ の生成元とし，$e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_1$ とする．
   c) $(q_1, q_2, \mathbb{G}, \mathbb{G}_1, e)$ を出力する．
   d) $n = q_1 q_2$ とする． 2 つのランダムな生成元 $g, u \xleftarrow{R} \mathbb{G}$ を生成し，$h = u^{q_2}$ を設定する．
   e) 公開鍵を $\mathcal{PK} = (n, \mathbb{G}, \mathbb{G}_1, e, g, h)$，秘密鍵を $\mathcal{SK} = q_1$ とする．
2) Enc:公開鍵 $\mathcal{PK}$，平文 m を用いて 乱数 $r \xleftarrow{R} \{0, 1, \ldots, n-1\}$ を選び，以下の計算を行う．
   $$C = g^m h^r \in \mathbb{G}$$
   暗号文として C を出力する．
3) Dec: 秘密鍵 $\mathcal{SK} = q_1$ を用いて暗号文 C を以下の計算により復号する．
   $$C^{q_1} = (g^m h^r)^{q_1} = (g^{q_1})^m$$
   $\hat{g} = g^{q_1}$ とした時，$\log_{\hat{g}}(C^{q_1})$ により m を求める

準同型性：公開鍵を $(n, \mathbb{G}, \mathbb{G}_1, e, g, h)$ とする．平文 $m_1, m_2 \in \{0, 1, \ldots, T\}$ の暗号文をそれぞれ $C_1, C_2 \in \mathbb{G}_1$ とした時，加算 $m_1 + m_2 \bmod n$ は $C = C_1 C_2 h^r$ によって計算できる．また，双線型性を用いることで以下のようにして一回のみ乗算を行うことができる．$g_1 = e(g, g)$，$h_1 = e(g, h)$ を設定する．このとき g_1 の位数は n，h_1 の位数は q_1 である．また $h = g^{\alpha q_2} (\alpha \in \mathbb{Z})$ とする．2 つの暗号文 C_1，C_2 が与えられた時，1) 乱数 $r \in \mathbb{Z}_n$ を選び，2)$C = e(C_1, C_2) h_1^r \in \mathbb{G}_1$ を設定する．以下により乗算を行う．

$$\begin{aligned} C =& e(C_1, C_2) h_1^r = e(g^{m_1} h^{r_1}, g^{m_2} h^{r_2}) h_1^r \\ =& g_1^{m_1 m_2} h_1^{m_1 r_2 + r_1 m_2 + \alpha q_2 r_1 r_2 + r} = g_1^{m_1 m_2} h_1^{\tilde{r}} \in \mathbb{G}_1 \end{aligned}$$

この時，$\mathbb{G}_1$ 上の暗号文 $g_1^m h_1^r$ は加法準同型性を保っているため，加算については前述の方法により行うことができる．

3 条件付き集計プロトコル

3.1 モデル

本節では，条件付き集計を行う PPDM のモデルを定義する．このモデルには Data Provider，Service Provider，Analyzer の三種類のエンティティが存在する．各エンティティの役割は以下の通りである．

- Data Provider (*DP*)： DP_i $(i = 1, 2, \ldots)$ はデータ (a_i, d_i) を *SP* の公開鍵 pk を用いて暗号化し，暗号化したデータ (a_i, d_i) を *SP* のデータベースに登録する．
- Service Provider (*SP*)：*SP* は $DP_i (i = 1, 2, \ldots)$ によって登録された暗号化データ (a_i, d_i) を保持し，Analyzer の要請を受けて，条件 $\tilde{A}$ に対する条件付き集計 $\mathsf{Enc}_{pk}(\sum_{a_i \in \tilde{A}} d_i)$ を計算する．
- Analyzer：Analyzer は *SP* に対し，条件 $\tilde{A}$ に対する条件付き集計を要請し，暗号化された計算結果 $\mathsf{Enc}_{pk}(\sum_{a_i \in \tilde{A}} d_i)$ を *SP* から受け取る．

また，2 つのアルゴリズム Register，Eval を設定する．

- Register : Register は DP_i がデータ (a_i, d_i) を登録する際に呼び出される．*SP* の公開鍵 pk 及びデータ (a_i, d_i) を入力とし，暗号化されたデータ (a_i, d_i) を出力する．
- Eval : Eval は *SP* が条件付き集計を行う際に呼び出される．*SP* の公開鍵 pk，DP_i $(i = 1, 2, \ldots,)$ によって暗号化されたデータ $\mathsf{Register}(pk, (a_i, d_i))$ 及び Analyzer から得た条件 $\tilde{A}$ を入力とし，条件付き集計 $\mathsf{Enc}_{pk}(\sum_{a_i \in \tilde{A}} d_i)$ を出力して Analyzer へ送る．

上記 2 つのアルゴリズムは以下の正当性条件を満足しなければならない．

$$\begin{aligned} &\mathsf{Eval}(pk, \mathsf{Register}(pk, (a_1, d_1)), \ldots, \mathsf{Register}(pk, (a_\ell, d_\ell)), \tilde{A}) \\ &= \mathsf{Enc}_{pk}(\sum_{a_i \in \tilde{A}} d_i) \end{aligned}$$

3.2 要件

条件付き集計は以下の 3 つの要件を満たす必要がある．

- 任意の $\tilde{A} \subseteq A$ となる条件 $\tilde{A}$ に対して条件付き集計を行うことができる．
- 条件 $\tilde{A}$ はデータ (a_i, d_i) を登録した後から指定することができる．すなわち，$\mathsf{Register}(pk, (a_i, d_i))$ は条件に依存することなく登録することができる．
- $\mathsf{Register}(pk, (a_i, d_i))$ $(i = 1, 2, \ldots, \ell)$，及び $\mathsf{Eval}(pk, \{\mathsf{Register}_1, \ldots, \mathsf{Register}_\ell\}, \tilde{A})$ からはデータ (a_i, d_i) や別の条件 $\tilde{A}'$ に対する条件付き集計の結果を得ることはできない．

提案方式では，上記の要件に加え，

1) $\mathsf{Register}(pk, (a_i, d_i))$ のデータサイズを可能な限り削減する．
2) $\mathsf{Eval}(pk, \{\mathsf{Register}_1, \ldots, \mathsf{Register}_\ell\}, \tilde{A})$ をより効率的に計算することを目的とする．

条件付き集計を暗号化した状態で行うにあたって，どのように条件 $\tilde{A}$ を満たすデータ (a_i, d_i) の検索を行うかが問題となる．提案方式では，条件の検索を行わずに集計を行うことでこの問題を回避している．以下に，加法準同型暗号を用いた方式，2-DNF 暗号を用いた方式，及び somewhat homomorphic encryption(SHE) を用いた最適化の方式について説明する．

3.3 加法準同型暗号を用いた単純な方式

この項では，加法準同型暗号を用いて条件の検索を行わずに集計を行う方式について述べる．

TABLE 1
提案方式における Register

	1	2	3	$\cdots$	$\lvert A\rvert - 1$	$\lvert A\rvert$
$\mathsf{Register}_1$	0	d_1	0	$\cdots$	0	0
$\mathsf{Register}_2$	0	0	d_2	$\cdots$	0	0
:	:	:	:	$\cdots$	:	:
$\mathsf{Register}_k$	0	0	0	$\cdots$	d_k	0

この方式の基本的なアイデアは，表 1 のように (a_i, d_i) を秘匿する点にある．各 (a_i, d_i) の暗号文としてそれぞれ $n = |A|$ 個のデータを持ち，a_i 番目にのみ $\mathsf{Enc}_{pk}(d_i)$ を，それ以外は $\mathsf{Enc}_{pk}(0)$ となるようにデータを保持する．条件 $j \in \tilde{A}$ に関する集計は，各 Register の j 番目のデータのみを加算することによって行う．この時，$a_i = j$ となっていれば $\mathsf{Enc}_{pk}(d_i)$，そうでない場合 $\mathsf{Enc}_{pk}(0)$ が加算されるため，$j \in \tilde{A}$ となるすべての j の集計値の加算を行うことで条件 $a_i \in \tilde{A}$ に対する条件付き集計を行うことができる．

この集計では，データの加算しか行わないため，加法準同型性のみを用いて条件付き集計を行うことが可能となる．しかし，この集計における Register による暗号文の出力データサイズは $O(n)$ であり，非常に大きなものとなっている．Register，Eval の詳細は以下のようになる．

Register: 公開鍵 pk 及び (a_i, d_i) を入力とし，

$$c_{i,j} = \begin{cases} \mathsf{Enc}_{pk}(d_i) & (a_i = j) \\ \mathsf{Enc}_{pk}(0) & (otherwise) \end{cases}$$

となる $(c_{i,0}, c_{i,1}, \ldots, c_{i,n})$ を出力する．
Eval: 公開鍵 pk，$\{\mathsf{Register}_i\}$，及び条件 $\tilde{A}$ を入力として，Eval アルゴリズムは以下によって計算を行う．

$$\mathsf{Eval}(pk, \{\mathsf{Register}_i\}, \tilde{A}) = \prod_i \prod_{j \in \tilde{A}} c_{i,j} = \mathsf{Enc}_{pk}\Big(\sum_{a_i \in \tilde{A}} d_i\Big)$$

3.4 条件を特徴付ける多項式を用いた方式

この方式では，先述の方式と比べ Register の出力データサイズは変わらないが，全く異なる手法によって (a_i, d_i) の暗号化を行う．以下の性質を満たす多項式 $F_{\tilde{A}}(x)$ を導入する．

$$F_{\tilde{A}}(x) = \begin{cases} 1 & (x \in \tilde{A}) \\ 0 & (otherwise) \end{cases}$$

このような多項式 $F_{\tilde{A}}(x)$ を導入することで，条件 $\tilde{A}$ による条件付き集計は以下の式で表すことができる．

$$\sum_{a_i \in \tilde{A}} d_i = \sum_{i=0}^{n} F_{\tilde{A}}(a_i) d_i$$

また，多項式 $F_{\tilde{A}}(x)$ は A と $\tilde{A}$ のみから求めることができる．このような多項式 $F_{\tilde{A}}(x)$ はラグランジュ補間によって求める．

$$F_{\tilde{A}}(x) = \sum_{i=1}^{n} F_i(x) = \sum_{a_i \in \tilde{A}} \prod_{a_j \in A, a_j \neq a_i} \frac{(x - a_j)}{(a_i - a_j)}$$

この時，$F_{\tilde{A}}(x)$ の次数は $(n-1)$ である．
このプロトコルの Register，Eval の詳細は以下のようになる．

Register:公開鍵 pk 及び (a_i, d_i) を入力として

$$e_{i,j} = \mathsf{Enc}_{pk}(a_i^j d_i) \qquad (0 \leq j < n)$$

となる $(e_{i,0}, e_{i,1}, \ldots, e_{i,n-1})$ を出力する．
Eval: 公開鍵 pk，$\{\mathsf{Register}_i\}$，及び条件 $\tilde{A}$ を入力として，Eval アルゴリズムは以下の計算を行う．

1) 以下の式で，$F_{\tilde{A}}(x)$ の x^ℓ に対応する係数 c_ℓ を計算する．

$$F_{\tilde{A}}(x) = c_0 + c_1 x + c_2 x^2 + \cdots + c_{n-1} x^{n-1}$$

2) 以下の式から条件付き集計を計算する．

$$\begin{aligned} &\mathsf{Eval}(pk, \{\mathsf{Register}_i\}, \tilde{A}) \\ =&\prod_{i=0}^{m} \prod_{j=0}^{n-1} e_{i,j}^{c_j} = \prod_{i=0}^{m} \prod_{j=0}^{n-1} \left(\mathsf{Enc}_{pk}(d_i a_i^j)^{c_j}\right) \\ =&\mathsf{Enc}_{pk}\left(\sum_{i=0}^{m} d_i \times (c_0 + c_1 a_i + \ldots + c_{n-1} a_i^{n-1})\right) \\ =&\mathsf{Enc}_{pk}\left(\sum_{i=0}^{m} d_i F_{\tilde{A}}(a_i)\right) = \mathsf{Enc}_{pk}\left(\sum_{a_i \in \tilde{A}} d_i\right) \end{aligned}$$

この方式では， Register の出力データサイズは依然として $O(n)$ である．

3.5 2-DNF 暗号を利用したデータ削減法

第 3 の方式では，2-DNF 暗号を用いることで Register の出力データサイズを $O(\sqrt{n})$ に削減している．データを削減するために，$\sqrt{n}$ 進展開を用いてデータを Register に分割して保持する点が特徴である．

$\sqrt{n}$ 進展開: データサイズ削減のために多項式の次数に対して $\sqrt{n}$ 進展開を適用する．$(0 \leq i \leq n)$ を満たす任意の i は以下の式によって求められる．

$$i = a\lceil\sqrt{n}\rceil + b \qquad (a = i/\sqrt{n}, b = i \bmod \sqrt{n})$$

$\sqrt{n}$ 進展開を x の次数に適用すると

$$x^i = x^{a\sqrt{n}+b} = x^{a\sqrt{n}} \times x^b$$

が得られる．よって，任意の x^i $(0 \leq i \leq n)$ は，$x^0, x^1, \ldots, x^{\sqrt{n}-1}$，及び $x^{\sqrt{n}}, x^{2\sqrt{n}}, \ldots, x^{(\sqrt{n}-1)\sqrt{n}}$ からなる $2\lceil\sqrt{n}\rceil$ 個の

データがあれば計算できる．すなわち，一回の乗算のみで x^i を計算することができる．

$C_1 \otimes C_2 = e(C_1, C_2)$ とすると，Register，Eval の詳細は以下のようになる．

Register: 公開鍵 pk 及び (a_i, d_i) を入力として，

$$e_{i,\ell,j} = \begin{cases} \mathsf{Enc}_{pk}(d_i a_i^j) & (\text{if } \ell = 0) \\ \mathsf{Enc}_{pk}(a_i^{j\cdot(\lceil\sqrt{n}\rceil)^\ell}) & (\text{otherwise}) \end{cases}$$

となる $(e_{i,0,0}, e_{i,0,1}, \ldots, e_{i,0,\sqrt{n}-1}, e_{i,1,0}, e_{i,1,1}, \ldots, e_{i,1,\sqrt{n}-1})$ を出力
Eval: 公開鍵 pk，$\{\mathsf{Register}_i\}$ 及び条件 $\tilde{A}$ を入力として，Eval アルゴリズムは以下の計算を行う．

1) 以下の式で，$F_{\tilde{A}}(x)$ の x^ℓ に対応する係数 c_ℓ を計算する．

$$F_{\tilde{A}}(x) = c_0 + c_1 x + c_2 x^2 + \cdots + c_{n-1} x^{n-1}$$

2) $j_0(j) = j \bmod \lceil\sqrt{n}\rceil$，$j_1(j) = j \operatorname{div} \lceil\sqrt{n}\rceil$ として以下の式から条件付き集計を計算する．

$$\begin{aligned} &\mathsf{Eval}(pk, \{\mathsf{Register}_i\}, \tilde{A}) \\ =&\prod_{i=0}^{m}\prod_{j=0}^{n-1}(e_{i,0,j_0(j)} \otimes e_{i,1,j_1(j)})^{c_j} \end{aligned} \tag{1}$$

上記の方式の正当性は以下の定理によって保証される．

定理 *1:* 条件付き集計は $\tilde{A} \subseteq A$ となる任意の条件 $\tilde{A}$ に対して実行可能であり，条件 $\tilde{A}$ はデータ (a_i, d_i) の登録後に指定することができる．

Proof: $e_{i,0,j_0}$ 及び $e_{i,1,j_1}$ の定義より，(2) 式は以下のように計算される．

$$\begin{aligned} &\prod_{i=0}^{m}\prod_{j=0}^{n-1}(e_{i,0,j_0(j)} \otimes e_{i,1,j_1(j)})^{c_j} \\ =&\prod_{i=0}^{m}\prod_{j=0}^{n-1}\left(\mathsf{Enc}_{pk}(d_i a_i^{j_0(j)}) \otimes \mathsf{Enc}_{pk}(a_i^{j_1(j)\lceil\sqrt{n}\rceil})\right)^{c_j} \\ =&\prod_{i=0}^{m}\prod_{j=0}^{n-1}\left(\mathsf{Enc}_{pk}(d_i a_i^{j})^{c_j}\right) \\ =&\mathsf{Enc}_{pk}\left(\sum_{i=0}^{m} d_i \times (c_0 + c_1 a_i + \ldots + c_{n-1} a_i^{n-1})\right) \\ =&\mathsf{Enc}_{pk}\left(\sum_{i=0}^{m} d_i F_{\tilde{A}}(a_i)\right) = \mathsf{Enc}_{pk}\left(\sum_{a_i \in \tilde{A}} d_i\right) \end{aligned}$$

ここで $\mathsf{Enc}_{pk}\left(\sum_{a_i\in\tilde{A}} d_i\right)$ は，2.1 で示されている通り条件 $\tilde{A}$ に対する d_i の条件付き集計となっている．この集計では，$e_{i,0,j_0}$ 及び $e_{i,1,j_1}$ は条件 $\tilde{A}$ には依存しない値となるため条件の独立性は保たれている． □

3.6 SHE_c による一般化

前述の方式では，Register の出力データサイズを $O(\sqrt{n})$ に削減した．ここで，任意回の加算と $c-1$ 回の乗算が可能な準同型性をもつ暗号 SHE_c を定義する．
SHE_c を用いて，以下の式の様に $\sqrt[c]{n}$ 進展開を行うことで更なるデータサイズの削減が可能となる．

$$a_i^j = a_i^{j_0 + j_1(\sqrt[c]{n}) + \cdots + j_{c-2}(\sqrt[c]{n})^{c-2} + j_{c-1}(\sqrt[c]{n})^{c-1}}$$

SHE_3 を用いると，Register の出力データサイズは $O(\sqrt[3]{n})$ まで削減される．SHE_3 による暗号文同士の乗算を $\otimes$ とした時 Register，Eval アルゴリズムの詳細は以下のようになる．

Register: 公開鍵 pk 及び (a_i, d_i) を入力として，

$$e_{i,\ell,j} = \begin{cases} \mathsf{Enc}_{pk}(d_i a_i^j) & (\text{if } \ell = 0) \\ \mathsf{Enc}_{pk}(a_i^{j\cdot(\lceil\sqrt{n}\rceil)^\ell}) & (\text{otherwise}) \end{cases}$$

となる
$(e_{i,0,0}, \ldots, e_{i,0,\sqrt[3]{n}-1}, e_{i,1,0}, \ldots, e_{i,1,\sqrt[3]{n}-1}, e_{i,2,0}, \ldots, e_{i,2,\sqrt[3]{n}-1})$
を出力する．
Eval: 公開鍵 pk，$\{\mathsf{Register}_i\}$ 及び条件 $\tilde{A}$ を入力として，Eval アルゴリズムは以下の計算を行う．

1) 以下の式で，$F_{\tilde{A}}(x)$ の x^ℓ に対応する係数 c_ℓ を計算する．

$$F_{\tilde{A}}(x) = c_0 + c_1 x + c_2 x^2 + \cdots + c_{n-1} x^{n-1}$$

2) $j_0(j) = j \bmod \lceil\sqrt[3]{n}\rceil$，$j_1(j) = (j \operatorname{div} \lceil\sqrt[3]{n}\rceil) \bmod \lceil\sqrt[3]{n}\rceil$，$j_2(j) = j \operatorname{div} (\lceil\sqrt[3]{n}\rceil)^2$ として条件付き集計を以下の式により行う．

$$\begin{aligned} &\mathsf{Eval}(pk, \{\mathsf{Register}_i\}, \tilde{A}) \\ =&\prod_{i=0}^{m}\prod_{j=0}^{n-1}(e_{i,0,j_0(j)} \otimes e_{i,1,j_1(j)} \otimes e_{i,2,j_2(j)})^{c_j} \end{aligned}$$

よって，乗算の回数を増やすことにより，出力データサイズの大幅な削減が可能となる．また，上述の方式は SHE_c に一般化することが可能であり，その場合の出力データサイズは $O(c\cdot\sqrt[c]{n})$ となる．

上述の方式では，データ (a_i, d_i) の a_i を条件，d_i を集計データとして扱っている．次の方式は，Data Provider DP_i に複合データ $d_{i,1}, d_{i,2}, \ldots, d_{i,R}$ $(d_{i,r} \in D_r)(r = 1, 2, \ldots, R)$ を登録することで，条件，集計どちらにでも利用可能とした方式となっている．

SHE_3 を用いた際のプロトコルの詳細は以下のようになる．また，SHE_c を用いた際にも同様に一般化される．

Register: 公開鍵 pk 及び $d_{i,k} \in D_k$ を入力として，

$$e_{i,k,\ell,j} = \mathsf{Enc}_{pk}(d_{i,k}^{j\cdot(\lceil\sqrt{n}\rceil)^\ell})$$

となる
$(e_{i,k,0,0}, \ldots, e_{i,k,0,\sqrt{n}-1}, e_{i,k,1,0}, \ldots, e_{i,k,1,\sqrt{n}-1})$
を出力する
Eval: Analyzer が条件 $\tilde{D}_{k'} \subseteq D_{k'}$ による D_k の集計を求めたとする．公開鍵 pk，$\{\mathsf{Register}_i\}$，及び条件 $\tilde{D}_{k'}$ を入力として，Eval アルゴリズムは以下の計算を行う．

1) 以下の式により $F_{\tilde{D}_{k'}}(x)$ の係数 c_ℓ を計算する．

$$F_{\tilde{D}_{k'}}(x) = c_0 + c_1 x + c_2 x^2 + \cdots + c_{n-1} x^{n-1}$$

2) 以下の計算により条件付き集計を行う．

$$\begin{aligned} &\mathsf{Eval}(pk, \{\mathsf{Register}_i\}, \tilde{D}_{k'}) \\ =&\prod_{i=0}^{m} e_{i,k,0,1} \otimes \prod_{j=0}^{n-1}(e_{i,k',0,j_0} \otimes e_{i,k',1,j_1})^{c_j} \\ =&\prod_{i=0}^{m}\mathsf{Enc}_{pk}(d_{i,k}) \otimes \\ &\qquad \prod_{j=0}^{n-1}\left(\mathsf{Enc}_{pk}(d_{i,k'}^{j_0})\mathsf{Enc}_{pk}(d_{i,k'}^{j_1\lceil\sqrt{n}\rceil})\right)^{c_j} \\ =&\prod_{i=0}^{m}\mathsf{Enc}_{pk}(d_{i,k}) \otimes \prod_{j=0}^{n-1}\left(\mathsf{Enc}_{pk}(d_{i,k'}^{j})^{c_j}\right) \\ =&\mathsf{Enc}_{pk}\left(\sum_{i=1}^{m} d_{i,k} \times \sum_{j=0}^{n-1} c_j d_{i,k}^j\right) \end{aligned}$$

$$=\mathsf{Enc}_{pk}\left(\sum_{i=0}^{m} d_{i,k}F_{\tilde{D}_{k'}}(d_{i,k'})\right)=\mathsf{Enc}_{pk}\left(\sum_{d_{i,k'}\in\tilde{D}_{k'}} d_{i,k}\right)$$

この方式では，Register の出力データサイズは n_i を集合 D_i の要素数とした時 $O(\sum_i n_i)$ となる．

ここで，SHE_3 の代わりに SHE_c を用いた場合，出力データサイズは $O((c-1)\cdot\sum_i \sqrt[c-1]{n_i})$ に削減される．また， c を $\log n$ とした時出力データサイズは $O(\sum_i \log n_i)$ となる．

4 複数の変数を条件に含む集計方式

上述した方式では，単一の条件に対する条件付き集計を行った．この章では，複数の変数を条件に含む条件付き集計を行うより複雑なプロトコルについて説明する．

4.1 2 変数を条件とする条件付き集計

このプロトコルは，データ組集合 $(a_i,b_i,d_i)\in A\times B\times D(i=1,2,...)$ が与えられた時，条件 $\tilde{A}\subseteq A$，$\tilde{B}\subseteq B$ を入力として，$a_i\in\tilde{A}$，$b_i\in\tilde{B}$ であるとき $a_i\in\tilde{A}\wedge b_i\in\tilde{B}$ または $a_i\in\tilde{A}\vee b_i\in\tilde{B}$ に対応する d_i の値を集計するプロトコルである．すなわち，$(a_i,b_i,d_i)(i=1,2,\dots)$ を入力として，$\mathsf{Enc}_{pk}(\sum_{a_i\in\tilde{A}\wedge b_i\in\tilde{B}} d_i)$ または $\mathsf{Enc}_{pk}(\sum_{a_i\in\tilde{A}\vee b_i\in\tilde{B}} d_i)$ を出力するプロトコルとなる．

提案方式のアイデアは，3.4 の方式と同様に，以下の性質を持つ多項式 $F_{\tilde{A}}(x),F_{\tilde{B}}(y)$ を導入する．

$$F_{\tilde{A}}(x)=\begin{cases}1 & (x\in\tilde{A})\\ 0 & (otherwise)\end{cases}\quad F_{\tilde{B}}(y)=\begin{cases}1 & (y\in\tilde{B})\\ 0 & (otherwise)\end{cases}$$

この 2 つの多項式の積 $F_{\tilde{A}}(x)F_{\tilde{B}}(y)$ は以下の性質を持つ．

$$F_{\tilde{A}}(x)F_{\tilde{B}}(y)=\begin{cases}1 & (x\in\tilde{A}\wedge y\in\tilde{B})\\ 0 & (otherwise)\end{cases}$$

多項式 $F_{\tilde{A}}(x)F_{\tilde{B}}(y)$ は $A,\tilde{A},B,\tilde{B}$ のみから求めることが可能である．この多項式を用いることで，$a_i\in\tilde{A}\wedge b_i\in\tilde{B}$ を条件とする集計は次の式のように表せる．

$$\mathsf{Enc}_{pk}(\sum_{a_i\in\tilde{A}\wedge b_i\in\tilde{B}} d_i)=\mathsf{Enc}_{pk}(\sum_{i=0}^{m} d_iF_{\tilde{A}}(x)F_{\tilde{B}}(y))$$

$\sum_{a_i\in\tilde{A}\wedge b_i\in\tilde{B}} d_i$ を集計するプロトコルの詳細は，n_a,n_b をそれぞれ集合 A,B の要素数としたとき以下のようになる．

Register: 公開鍵 pk 及び (a_i,b_i,d_i) を入力として，

$$e_{i,j}=\mathsf{Enc}_{pk}(d_ia_i^j)(0\le j<n_a)$$
$$h_{i,\ell}=\mathsf{Enc}_{pk}(b_i^\ell)(0\le \ell<n_b)$$

となる $(e_{i,0},\dots,e_{i,n_a-1})$, $(h_{i,0},\dots,h_{i,n_b-1})$
を出力する

Eval: 公開鍵 pk，$\{\mathsf{Register}_i\}$ 及び条件 $\tilde{A},\tilde{B}$ を入力として，Eval アルゴリズムは以下の計算を行う．

1) 以下の式で，$F_{\tilde{A}}(x)F_{\tilde{B}}(y)$ の x^j,y^ℓ に対応する係数 c_j,t_ℓ を計算する．

$$F_{\tilde{A}}(x)F_{\tilde{B}}(y)=\left(\sum_{j=0}^{n_a-1} c_jx^j\right)\left(\sum_{j=0}^{n_b-1} t_\ell y^\ell\right)$$

2) 以下の式によって条件付き集計を行う．

$$\mathsf{Eval}(pk,\{\mathsf{Register}_i\},\tilde{A},\tilde{B})=\prod_{i=0}^{m}\prod_{j=0}^{n_a-1}\prod_{\ell=0}^{n_b-1}(e_{i,j}\otimes h_{i,\ell})^{c_jt_\ell}$$
$$=\prod_{i=0}^{m}\prod_{j=0}^{n_a-1}\prod_{\ell=0}^{n_b-1}\mathsf{Enc}_{pk}(d_ia_i^j)\otimes\mathsf{Enc}_{pk}(b_i^\ell)^{c_jt_\ell}$$
$$=\prod_{i=0}^{m}\prod_{j=0}^{n_a-1}\prod_{\ell=0}^{n_b-1}\mathsf{Enc}_{pk}(d_ia_i^jb_i^\ell)^{c_jt_\ell}$$
$$=\mathsf{Enc}_{pk}\left(\sum_{i=0}^{m} d_i\times\left(\sum_{j=0}^{n_a-1} c_ja_i^j\right)\left(\sum_{\ell=0}^{n_b-1} t_\ell b_i^\ell\right)\right)$$
$$=\mathsf{Enc}_{pk}\left(\sum_{i=0}^{m} d_iF_{\tilde{A}}(a_i)F_{\tilde{B}}(b_i)\right)$$
$$=\mathsf{Enc}_{pk}\left(\sum_{\{a_i\in\tilde{A}\wedge b_i\in\tilde{B}\}} d_i\right)$$

また，$\sum_{a_i\in\tilde{A}\vee b_i\in\tilde{B}} d_i$ は $\hat{A}=\bar{\tilde{A}},\hat{B}=\bar{\tilde{B}}$ としたとき以下の式によって求められる．

$$\mathsf{Enc}_{pk}\left(\sum_{a_i\in\tilde{A}\vee b_i\in\tilde{B}} d_i\right)=\mathsf{Enc}_{pk}\left(\sum_{all} d_i-\sum_{a_i\in\hat{A}\wedge b_i\in\hat{B}} d_i\right)$$

これらの集計における Register の出力データサイズは $O(n_a+n_b)$ となる．

4.2 SHE_c を用いた多変数を条件に含む集計方式

前述の SHE を適用することで，3 条件以上での集計，もしくは暗号文の出力データサイズの削減をすることが可能となる．

3 条件以上を条件とした集計方式

SHE_3 を用いると，3 条件での集計が可能となる．このとき，n_a,n_b,n_v を集合 A,B,V の要素数として，$\sum_{a_i\in\tilde{A}\wedge b_i\in\tilde{B}\wedge v_i\in\tilde{V}} d_i$ を集計するプロトコルの詳細は以下のようになる．

Register: 公開鍵 pk 及び (a_i,b_i,v_i,d_i) を入力として，

$$e_{i,j}=\mathsf{Enc}_{pk}(d_ia_i^j)(0\le j<n_a)$$
$$h_{i,\ell}=\mathsf{Enc}_{pk}(b_i^\ell)(0\le \ell<n_b)$$
$$g_{i,s}=\mathsf{Enc}_{pk}(v_i^s)(0\le s<n_v)$$

となる
$(e_{i,0},\dots,e_{i,n_a-1})$, $(h_{i,0},\dots,h_{i,n_b-1})$, $(g_{i,0},\dots,g_{i,n_v-1})$ を出力する

Eval: 公開鍵 pk，$\{\mathsf{Register}_i\}$ 及び条件 $\tilde{A},\tilde{B},\tilde{V}$ を入力として，Eval アルゴリズムは以下の計算を行う．

1) 以下の式で，$F_{\tilde{A}}(x)F_{\tilde{B}}(y)F_{\tilde{V}}(z)$ の x^j,y^ℓ,z^s に対応する係数 c_j,t_ℓ,u_s を計算する．

$$F_{\tilde{A}}(x)F_{\tilde{B}}(y)F_{\tilde{V}}(z)$$
$$=\left(\sum_{j=0}^{n_a-1} c_jx^j\right)\left(\sum_{\ell=0}^{n_b-1} t_\ell y^\ell\right)\left(\sum_{s=0}^{n_v-1} u_sz^s\right)$$

2) 以下の式によって条件付き集計を行う．

$$\mathsf{Eval}(pk,\{\mathsf{Register}_i\},\tilde{A},\tilde{B},\tilde{V})$$
$$=\prod_{i=0}^{m}\prod_{j=0}^{n_a-1}\prod_{\ell=0}^{n_b-1}\prod_{s=0}^{n_v-1}(e_{i,j}\otimes h_{i,\ell}\otimes g_{i,s})^{c_jt_\ell u_s}$$

出力データサイズを削減する方式

この項では，暗号文の出力データサイズをより削減する方式について説明する．SHE_4 を用いると以下によって出力データサイズの削減が可能となる．プロトコルの詳細は以下のようになる．

Register: 公開鍵 pk 及び (a_i, d_i) を入力として，

$$e_{i,k,j} = \begin{cases} \mathsf{Enc}_{pk}(d_i a_i^j) & (\text{if } k = 0) \\ \mathsf{Enc}_{pk}(a_i^{j\cdot(\lceil\sqrt{n}\rceil)^k}) & (\text{otherwise}) \end{cases}$$

$$h_{i,s,\ell} = \begin{cases} \mathsf{Enc}_{pk}(b_i^{\ell}) & (\text{if } s = 0) \\ \mathsf{Enc}_{pk}(b_i^{\ell\cdot(\lceil\sqrt{n}\rceil)^s}) & (\text{otherwise}) \end{cases}$$

となる

$(e_{i,0,0}, e_{i,0,1}, \ldots, e_{i,0,\sqrt{n}-1}, e_{i,1,0}, e_{i,1,1}, \ldots, e_{i,1,\sqrt{n}-1})$
$(h_{i,0,0}, h_{i,0,1}, \ldots, h_{i,0,\sqrt{n}-1}, h_{i,1,0}, h_{i,1,1}, \ldots, h_{i,1,\sqrt{n}-1})$ を出力する．

Eval: 公開鍵 pk，$\{\mathsf{Register}_i\}$ 及び条件 $\tilde{A}, \tilde{B}$ を入力として，Eval アルゴリズムは以下の計算を行う．

1) 以下の式で，$F_{\tilde{A}}(x)F_{\tilde{B}}(y)$ の x^j, y^ℓ に対応する係数 c_j, t_ℓ を計算する．

$$F_{\tilde{A}}(x)F_{\tilde{B}}(y) = \left(\sum_{j=0}^{n_a-1} c_j x^j\right)\left(\sum_{\ell=0}^{n_b-1} t_\ell y^\ell\right)$$

2) 以下の式によって条件付き集計を行う．

$$\mathsf{Eval}(pk, \{\mathsf{Register}_i\}, \tilde{A}, \tilde{B})$$
$$= \prod_{i=0}^{m}\prod_{j=0}^{n_a-1}\prod_{\ell=0}^{n_b-1} (e_{i,0,j_0(j)} \otimes e_{i,1,j_1(j)})^{c_j} \otimes (h_{i,0,\ell_0(\ell)} \otimes h_{i,1,\ell_1(\ell)})^{t_\ell}$$

より効率的な集計方式

上述の方式を実際のデータに適用する際には，データの種類によって，暗号文の出力データサイズの削減を行うもの，行わないものを分けることでより効率的な集計を行うことができる．例えば，(a_i, b_i, d_i) から $\mathsf{Enc}_{pk}(\sum_{a_i\in\tilde{A}\wedge b_i\in\tilde{B}} d_i)$ の集計を行う際に，A が年齢や身長などある程度の値の上限があるようなデータであり，$n_a \ll n_b$ となるデータである時，a_i についてはデータの分割を行わず $\mathsf{Enc}_{pk}(d_i a_i)$ として保持し，b_i に対してデータの分割を行うことで，集計における暗号文の総出力データサイズを抑えることができる．

以下に SHE_4 を用いた際の一例を示す．

Register: 公開鍵 pk 及び (a_i, b_i, d_i) を入力として，

$$e_{i,j} = \mathsf{Enc}_{pk}(a_i^j d_i) \qquad (0 \le j < n_a)$$

$$h_{i,s,\ell} = \begin{cases} \mathsf{Enc}_{pk}(b_i^{\ell}) & (\text{if } s = 0) \\ \mathsf{Enc}_{pk}(b_i^{\ell\cdot(\lceil\sqrt{n}\rceil)^s}) & (\text{otherwise}) \end{cases}$$

となる

$(e_{i,0}, e_{i,1}, \ldots, e_{i,n_a-1})$ $(h_{i,0,0}, \ldots, h_{i,0,\sqrt[3]{n_b}-1}, h_{i,1,0}, \ldots, h_{i,1,\sqrt[3]{n_b}-1}, h_{i,2,0}, \ldots, h_{i,2,\sqrt[3]{n_b}-1})$ を出力する．

Eval: 公開鍵 pk，$\{\mathsf{Register}_i\}$ 及び条件 $\tilde{A}, \tilde{B}$ を入力として，Eval アルゴリズムは以下の計算を行う．

1) 以下の式により，$F_{\tilde{A}}(x)F_{\tilde{B}}(y)$ の x^j, y^ℓ に対応する係数 c_j, t_ℓ を計算する．

$$F_{\tilde{A}}(x)F_{\tilde{B}}(y) = \left(\sum_{j=0}^{n_a-1} c_j x^j\right)\left(\sum_{\ell=0}^{n_b-1} t_\ell y^\ell\right)$$

2) $\ell_0(\ell) = j \bmod \lceil\sqrt[3]{n_b}\rceil$, $\ell_1(\ell) = (\ell \operatorname{div} \lceil\sqrt[3]{n_b}\rceil) \bmod \lceil\sqrt[3]{n_b}\rceil$, $\ell_2(\ell) = \ell \operatorname{div} (\lceil\sqrt[3]{n_b}\rceil)^2$ として以下の式によって条件付き集計を行う．

$$\mathsf{Eval}(pk, \{\mathsf{Register}_i\}, \tilde{A}, \tilde{B})$$
$$= \prod_{i=0}^{m}\prod_{j=0}^{n_a-1}\prod_{\ell=0}^{n_b-1} (e_{i,j})^{c_j} \otimes (h_{i,0,\ell_0(\ell)} \otimes h_{i,1,\ell_1(\ell)} \otimes h_{i,2,\ell_2(\ell)})^{t_\ell}$$

この集計における暗号文の出力データサイズは $O(n_a + 3\sqrt[3]{n_b})$ であり $n_a \ll n_b$ であるときには，少ないデータでの集計が可能となる．

同様の議論により，乗算の回数を増やすことにより，(1) 多変数の任意の条件での集計 (2) 出力データサイズの削減が可能 (3)$\tilde{A} \cap \tilde{B}, \tilde{A} \cup \tilde{B}$ を組み合わせることで，任意の条件での集計が可能となる．また，上記の方式は SHE_c に容易に適用可能である．

5 むすび

本研究では，準同型暗号を用いた条件付き集計プロトコルについての提案を行った．提案方式では，加法準同型暗号のみを用いた方式と比較して，Register の出力データサイズが $O(n)$ から $O(\sqrt{n})$ へと削減されている．また，多変数の条件に対する集計方式についての提案を行った．

提案方式の優位性として，(i) 任意の条件を指定可能である (ii) 指定した条件での集計値以外の情報は漏洩しない (iii)SHE への拡張が容易に行える点があげられる．SHE を用いた場合更なるデータサイズの削減，及び任意のデータでの集計が可能となる．

今後の課題としては，提案方式の安全性の証明及び方式の更なる一般化が挙げられる．

REFERENCES

[1] A. Basu, J. Vaidya, and H. Kikuchi, “Efficient Privacy-Preserving Collaborative Filtering Based on the Weighted Slope One Predictor,” in Journal of Internet Services and Information Security, 2001, vol. 1, no. 4, pp. 26-46.

[2] D. Bone et al, “Evaluating 2-DNF Formulas on Ciphertexts,” in Proc. Theory of Cryptography Conf. 2005, LNCS 3378, pp.325-342.

[3] I. Damgard and M. Jurik. “A generalisation, a simplification and some applications of Paillier’s probabilistic public-key system,” Proc. Public Key Cryptography 2001, LNCS 1992, pp.119-136. Springer.

[4] I. Damgard and M. Jurik, “A Length-Flexible Threshold Cryptosystem with Applications,” in Australasian Conf. Information Security and Privacy 2003,Heidelberg, 2003, LNCS 2727, pp.350-364.

[5] W. Diffie and M. E. Hellman, “New Directions in Cryptography”, IEEE Transactions on Information Theory, vol.IT-22, No.6, pp.644-654, Nov, 1976.

[6] T. ElGamal, “A Public-Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms,” IEEE Transactions on Information Theory, 1985, v. IT-31, n. 4, pp. 469-472 or CRYPTO 84, pp. 10-18.

[7] M. J. Freedman et al, “Efficient private matching and set intersection,” in Proc. of Eurocrypt 2004, 2004, LNCS 3027, pp.1-19.

[8] C. Gentry, “Fully homomorphic encryption using ideal lattices,” in Symp. the Theory of Computing , 2009, pp. 169-178.

[9] E. Kushilevitz and R. Ostrovsky. “Replication is not needed: Single database, computationallyprivate information retrieval (extended abstract)”, in 38th Annual Symp. Foundations of Computer Science, 1997, pp. 364-373.

[10] D. Nakamura and K. Kobayashi, “Modified ElGamal Cryptosystem,” in ISW’97 Proc. First International Workshop on Informetion Security,1998, pp. 96-108.

[11] P. Paillier, “Public-Key Cryptosystems Based on Composite Degree Residuosity Classes,” EUROCRYPT 1999, 1999, pp. 223-238.