プライバシーを考慮した準同型暗号ベースの秘匿類似検索技術

平野　貴人†　　川合　豊†　　小関　義博†

†三菱電機株式会社　情報技術総合研究所
247-8501　神奈川県鎌倉市大船 5-1-1
{Hirano.Takato@ay, Kawai.Yutaka@da, Koseki.Yoshihiro@ak}.MitsubishiElectric.co.jp

あらまし 暗号化したまま類似検索できる準同型暗号ベースの秘匿検索技術は，プライバシーを保護したままゲノム分析などができる手法として期待されている．しかし，直接的に本技術を適用した場合，データが暗号化されているにも関わらず，検索者（ゲノム分析者）はその平文を推測できる可能性があるため，プライバシーの問題が生じる．そこで本稿では，本課題を解決する準同型性を利用した手法を提案する．具体的には，準同型性を用いて，暗号文の中の平文をランダム化する手法を利用し，検索者には，検索結果のスコアがある閾値以下であれば平文のスコア情報を，閾値以上であればその事実のみ知らせるような手法を提案する．また，スコアがある閾値よりも大きいか否かの情報のみ検索者は得られるような手法も提案する．

Privacy-Enhancing Techniques for Similarity-Searchable Encryption Based on Homomorphic Encryption

Takato Hirano†　　Yutaka Kawai†　　Yoshihiro Koseki†

†Mitsubishi Electric Corporation, Information Technology R&D Center.
5-1-1 Ofuna, Kamakura, Kanagawa 247-8501, JAPAN
{Hirano.Takato@ay, Kawai.Yutaka@da, Koseki.Yoshihiro@ak}.MitsubishiElectric.co.jp

Abstract Similarity-searchable encryption technology based on homomorphic encryption has been proposed and expected as a potential tool for constructing privacy-preserving genome analysis methods. However, there is a privacy issue that a data searcher (i.e. genome analyst) would easily infer plain genome data of patients when the method is directly used for constructing the private analysis scheme, even if the genome data is encrypted. In this paper, we propose generic techniques to address the privacy issue. Concretely, by using a homomorphic operation which randomizes the hidden plaintext of encrypted data, we show a technique that the data searcher obtains the plain score given as search result if it is less than a given threshold value, or receives only information on "the plain score is over the threshold value". We also propose further techniques that the data searcher receives only information on the fact whether the plain score is over the threshold value or not.

1　はじめに

秘匿検索技術とは，データを暗号化したまま検索可能な暗号技術であり，クラウドコンピューティング向けの技術として知られている．現在までに数多くの具体的な方式が提案されているが，その多くが完全一致検索のみに注目してい

た方式であった．近年，部分一致検索可能な方式 [3]，あいまい検索可能な方式 [7]，類似検索可能な方式 [9, 10] が提案され始めている．

中でも類似検索可能な方式は，プライバシーを保護したままデータ分析を実行できるため，プライバシーを保護したままゲノム分析などができる手法として期待されている．秘匿類似検索方式の多くは，準同型暗号と呼ばれる，暗号化したまま隠れた平文同士の和や積などの演算ができる暗号技術を用いて構成されている．

ここで，準同型暗号は，大まかには下記のような 3 タイプに分類される．

群準同型暗号 暗号文同士の加算や乗算のどちらか一方の演算のみ実行可能な暗号は群準同型暗号と呼ばれる．例えば，RSA 暗号方式や ElGamal 暗号方式は乗算のみ実行可能な群準同型暗号であり，乗法的準同型暗号とも呼ばれる．また，ElGamal 暗号方式の平文のエンコーディング方法を変形させた Additive ElGamal 暗号方式や Paillier 暗号方式 [6] は，加算のみ実行可能な群準同型暗号であり，加法的準同型暗号とも呼ばれる．

Somewhat 準同型暗号 暗号文同士の加算と乗算の両方の演算を実行可能であるが，演算回数に制限がある暗号は Somewhat 準同型暗号と呼ばれる．例えば，楕円曲線上で構成された Boneh-Goh-Nissim 暗号方式 [1] や，格子をベースに構成された Brakerski-Vaikuntanathan 暗号方式 [2] などが Somewhat 準同型暗号方式として知られている．また，内積などの演算の高速化に注目した Lauter-Naehrig-Vaikuntanathan 方式 [5] や安田らの方式 [8]，小暮らの方式 [10] も知られている．

完全準同型暗号 暗号文同士の加算と乗算の両方の演算を制限回数なく実行可能な暗号は完全準同型暗号と呼ばれる．格子をベースに構成された Gentry 暗号方式 [4] などが知られている．

加法的準同型暗号をベースに構成された秘匿類似検索方式 [9] では，化合物データベースの検索に注目しており，類似度の指標として Tversky Index を用いている．本方式では，データベース内のデータを検索者に隠しつつ，また検索者のクエリ内容をデータベース側に知られることなくく秘匿類似検索を達成している．また，安全性を向上させるため，ダミー情報を追加することや，類似度のアファイン変換，また複数個の検索結果をランダムに置換して検索者に返信することで，検索結果として類似度がある閾値以内か否かの情報のみが現れるような（つまり，漏れる情報量を極力減らす）手法も [9] の中で述べられている．なお，本方式では，化合物データベースのデータはすべて平文で保管されていることが前提であり，保管データが暗号化されている場合や，クラウドサーバなどへのデータ保管や計算委託のような場合は述べられていない．

内積計算の効率化に注目した Somewhat 準同型暗号を直接的に用いて構成された秘匿類似検索 [10] では，[9] とは異なり，クラウドサーバへのデータ保管や計算委託を意識しており，保管データと検索データがともに暗号化されている状況での秘匿類似検索に注目している．本方式では，類似度としてハミング距離を用いており，保管データに対して，検索者は自身の設定した検索パターンのすべての位置に関する（1 文字ずつずらした毎の）類似度を得ることができる．

具体的には，平文が「00011」の保管データに対して，平文が「000」の検索パターンで検索した場合には，保管データの先頭 3 ビット「000」と検索パターン「000」のハミング距離「0」，保管データの 2 ビット目から 4 ビット目の「001」と検索パターン「000」のハミング距離「1」，保管データの 3 ビット目から 5 ビット目の「011」と検索パターン「000」のハミング距離「2」を得ることができる．つまり，検索結果のスコアとして「012」を検索者は得ることができる．

[10] では提案方式のゲノム分析への応用についても述べられている．しかしながら，本方式のような保管データと検索パターンとのすべての場所の類似度を検索者（ゲノム分析者）が得られる方式を直接的に用いて構成しても，検索者は自身の検索した検索パターンと検索結果のスコアから暗号化ゲノムデータの平文を推測でき

る可能性があり，プライバシーの問題が生じる．

前述の例を再度用いると，検索者は自身の検索パターン「000」と検索の結果得られたスコア「012」を使うことで，保管データ「00011」を復元できる．具体的には，検索パターン「000」に対して最初の 3 ビットのスコアが「0」であることから，検索者は保管データの先頭 3 ビットを「000」と復元できる．また，2 ビット目から 4 ビット目のスコアが「1」であることや，2 ビット目と 3 ビット目がすでに「0」と確定しているので，4 ビット目は「1」であることが復元できる．同様にして，5 ビット目を「1」と推測でき，保管データを「00011」と復元できる．つまり，検索者は，自身が選択した検索パターンとのスコアが「0」と出たところから辿れば，保管データの平文を特定することが可能である．

そこで本稿では，準同型性を用いて，本課題を解決する一般的な手法を提案する．具体的には，ある閾値以上の値を持つ平文のみをランダム化できるように準同型性を用いて暗号文を操作することで，データ検索者（ゲノム分析者）には，検索結果のスコアが閾値以上であればその情報のみを，閾値以下であれば実際のスコアが現れるような方式を提案する（方式 1）．また，方式 1 をさらに応用して，データ分析者はスコアがある閾値以上か否かの情報のみがわかるような方式も提案する（方式 2）．さらに，乗法的な準同型性を用いて，方式 2 の暗号文サイズを減らす手法も提案する（方式 3）．

2 準備

2.1 準同型暗号

以下のアルゴリズムの 5 つ組 (**KeyGen**, **Enc**, **Dec**, **AddHomo**, **MultHomo**) を，(公開鍵型) 準同型暗号方式と呼ぶ．

1. **KeyGen:** セキュリティパラメータ 1^λ を入力として，公開鍵 pk と秘密鍵 sk のペアを出力する．
2. **Enc:** 平文空間 $\mathcal{M}_{pk}$ の元 m と公開鍵 pk を入力として，暗号文 c を出力する．ここで，$c = E_{pk}(m)$ と表すことにする．
3. **Dec:** 暗号文 c と秘密鍵 sk を入力として，平文 $m' \in \mathcal{M}_{pk}$ を出力する．
4. **AddHomo:** 2 つの暗号文 $c_1 = E_{pk}(m_1)$, $c_2 = E_{pk}(m_2)$ と公開鍵 pk を入力として，暗号文 $c = E_{pk}(m_1 + m_2)$ を出力する．なお，c_1 と c_2 に対する本演算を $\oplus$ と表わし，$c = c_1 \oplus c_2$ と表すことにする．
5. **MultHomo:** 2 つの暗号文 $c_1 = E_{pk}(m_1)$, $c_2 = E_{pk}(m_2)$ と公開鍵 pk を入力として，暗号文 $c = E_{pk}(m_1 \times m_2)$ を出力する．なお，c_1 と c_2 に対する本演算を $\otimes$ と表わし，$c = c_1 \otimes c_2$ と表記する．

ここで，[5, 8, 10] のような複数の平文を一括で暗号化できる準同型暗号方式では，平文のベクトル $\vec{m} = (m_1, \ldots, m_n)$ に対して，暗号文 c を $E_{pk}(\vec{m})$ または $E_{pk}(m_1, \ldots, m_n)$ と表記することにする．また，平文を一括で暗号化できない準同型暗号方式に対して，暗号文 $(E_{pk}(m_1), \ldots, E_{pk}(m_n))$ も，$E_{pk}(\vec{m})$ または $E_{pk}(m_1, \ldots, m_n)$ と表記することにする．なお，後に提案する 3 方式はどちらのタイプの準同型暗号方式にも適用できるため，本稿では区別せず同じ表記方法を用いることとする．

2.2 類似度

よく知られている類似度として，ハミング距離や L_1, L_2 ノルム（より一般に L_p ノルム）などがある．また，化合物同士の類似度を測る指標として Tversky Index が知られている [9]．ここで，$\vec{x}, \vec{y} \in \{0,1\}^n$ とし，n は 1 以上の整数とする．

- $\vec{x}$ と $\vec{y}$ の間のハミング距離 $w_H(\vec{x}, \vec{y})$ の定義は，

$$w_H(\vec{x}, \vec{y}) = \sum_{i=1}^{n} (x_i \oplus y_i)$$

である．ただし，$\vec{x} = (x_1, \ldots, x_n)$ および $\vec{y} = (y_1, \ldots, y_n)$ とする．$w_H(\vec{x}, \vec{y})$ は，(ある閾値よりも) 小さいほど 2 つのベクトルは似ていると判断する．なお，準同型暗号を

用いることで，次のようにして暗号文 $c_1 = E_{pk}(\vec{x})$ と $c_2 = E_{pk}(\vec{y})$ の暗号化ハミング距離 $E_{pk}(w_H(\vec{x},\vec{y}))$ を計算できる．

$$\bigoplus_{i=1}^{n}((E_{pk}(x_i) \oplus E_{pk}(-y_i)) \otimes (E_{pk}(x_i) \oplus E_{pk}(-y_i)))$$

ここで，$E_{pk}(-y_i)$ は，加法的な準同型演算を用いて $E_{pk}(y_i)$ から効率的に計算できることに注意する．

なお，[5, 8, 10] では，暗号化したままベクトル同士の内積演算をより高速に計算できる手法が提案されており，その手法を用いれば上記よりも高速に暗号化ハミング距離を計算できる．

- $\vec{x},\vec{y}$ の間の Tversky Index $w_{TI}(\vec{x},\vec{y})$ の定義は，

$$w_{TI}(\vec{x},\vec{y}) = \frac{|\vec{x} \cap \vec{y}|}{|\vec{x} \cap \vec{y}| + \alpha|\vec{x} \backslash \vec{y}| + \beta|\vec{y} \backslash \vec{x}|}$$

である．ただし，$\alpha, \beta \in [0,1] \subseteq \mathbb{R}$ である．$w_{TI}(\vec{x},\vec{y})$ は，（ある閾値よりも）大きいほど 2 つのベクトル（化合物）は似ていると判断する．なお，暗号化したまま Tversky Index を計算できる手法が [9] で提案されている．

ここで，暗号化類似度の表記方法について述べる．2 つのベクトル $\vec{x} = (x_1, \ldots, x_n)$，$\vec{y} = (y_1, \ldots, y_m)$ $(m \leq n)$ のそれぞれの暗号文 $E_{pk}(\vec{x})$ と $E_{pk}(\vec{y})$ に対して，それらの暗号文の暗号化類似度を $E_{pk}(\vec{d}) = E_{pk}(d_1, \ldots, d_\ell)$ と書くことにする．例えば，$\vec{x} = (0,0,1)$, $\vec{y} = (0,0,0)$ かつ類似度としてハミング距離を用いるとき，暗号化類似度は $E_{pk}(\vec{d}) = E_{pk}(1)$（$\ell = 1$）と表わされる．また，[10] において，$\vec{x} = (0,0,0,1,1)$, $\vec{y} = (0,0,0)$ の場合の暗号化類似度は $E_{pk}(\vec{d}) = E_{pk}(0,1,2)$（$\ell = 3$）と表わされる．

3 エンティティ

本稿では，下記のエンティティが現れるシステムモデルを想定して議論を行う．

鍵生成者: 本システムに関わる公開鍵と秘密鍵のペアを作成し，公開鍵のみを登録者とサーバに送付し，また公開鍵と秘密鍵のペアを検索者に送付するエンティティ．悪意はないと仮定する．

登録者: 鍵生成者から受け取った公開鍵を使って，自身のゲノム情報などのデータを暗号化し，暗号化データをサーバに保管するエンティティ．悪意はないと仮定する．

検索者: 鍵生成者から受け取った公開鍵と秘密鍵を使って，検索パターンを暗号化し，サーバに保管されている登録者の暗号化データに対して，その暗号化検索パターンを用いて類似検索を行うエンティティ．登録者の暗号化データの平文に対して興味があり，Semi-Honest な振る舞いをすると仮定する．また，サーバと結託しないと仮定する．

サーバ: 登録者から暗号化データを受けとり，暗号化データを保管するエンティティ．また，検索者から受け取った暗号化検索パターンと，鍵生成者から受け取った公開鍵を使って，登録者の暗号化データに対して処理を行い，検索者にその処理結果を返すことも行う．登録者の暗号化データの平文や，検索者の暗号化検索パターンの平文に興味があり，Semi-Honest な振る舞いをすると仮定する．また，検索者と結託しないと仮定する．

4 提案方式

本章では，プライバシー性を高めた秘匿類似検索方式を 3 つ提案する．各方式で用いている共通の方針（手法）としては，下記の通りである．

- 閾値以内の平文を持つ暗号文から，0 の平文を持つ暗号文を作る．
- 閾値よりも大きい平文を持つ暗号文は，ランダムな値を平文として持つ暗号文に変換する．

上記は準同型性を用いて達成できる．具体的に，平文が d の暗号文 $E_{pk}(d)$ に対して，以下のような手順に従って計算を行うことで達成できる．

1. $E_{pk}(d)$ に対して，加法的な準同型演算を用いて，$E_{pk}(d-0), \ldots, E_{pk}(d-\theta)$ を計算する．このとき，$0 \leq d \leq \theta$ であれば，$E_{pk}(d-0), \ldots, E_{pk}(d-\theta)$ の中に必ず $E_{pk}(0)$ が現れる．そうでなければ，$E_{pk}(0)$ は現れない．
2. 各 $E_{pk}(d-0), \ldots, E_{pk}(d-\theta)$ に対して，0 でない乱数 $R_0, \ldots, R_\theta$ を用いて，乗法的な準同型演算から $E_{pk}(R_0(d-0)), \ldots, E_{pk}(R_\theta(d-\theta))$ を計算する．このとき，平文が 0 の暗号文は平文は 0 のまま変化せず，一方で 0 以外の平文はランダム化される．つまり，$0 \leq d \leq \theta$ であれば，ランダム化しても同様に $E_{pk}(R_0(d-0)), \ldots, E_{pk}(R_\theta(d-\theta))$ の中で $E_{pk}(0)$ が必ず現れる．従って，各暗号文を復号してどこかで 0 が現れれば d は閾値以内，そうでない場合には d は閾値より大きいと判定できる．

なお，閾値の設定方法であるが，検索者が検索時に閾値を設定できるようにすると，大きな閾値を設定することや，同じクエリ内容を用いて徐々に閾値を上げていくことで，暗号化された登録データの平文の情報が復元されてしまう可能性がある．よって，提案方式では，閾値はシステムのセットアップ時に決まっていることや，登録者が（小さな）閾値を設定していることを仮定する．

4.1 方式 1

本節で提案する方式 1 では，検索者は，類似度が閾値以内であれば具体的な類似度の値を，そうでない場合は閾値よりも大きいといった事実のみ知ることができる．本方式を以下に提案する．

- **KeyGen:**
  1. 鍵生成者は，準同型暗号方式の公開鍵 pk と秘密鍵 sk を生成する．
  2. 鍵生成者は，登録者とサーバに pk を，検索者に (pk, sk) を渡す．
- **Store:**
  1. 登録者は，保管したいデータ $\vec{m} = (m_1, \ldots, m_n)$ に対して，$C = E_{pk}(\vec{m})$ を計算する．
  2. 登録者は，C をサーバに保管する．
- **Search:**
  1. 検索者は，検索したいパターン $\vec{p} = (p_1, \ldots, p_m)$ $(m \leq n)$ に対して，$E_{pk}(\vec{p})$ を計算し，サーバに送る．
  2. サーバは，受け取った $E_{pk}(\vec{p})$ と各保管データ C に対して，$E_{pk}(\vec{d})$ を計算する．
  3. サーバは，閾値 θ を用いて，各暗号文 C に対して以下を計算する．

$$\begin{aligned} C^0 &= E_{pk}(\vec{d}) \oplus E_{pk}(-\vec{0}) \\ &= E_{pk}(d_1, \ldots, d_\ell) \\ &\vdots \\ C^\ell &= E_{pk}(\vec{d}) \oplus E_{pk}(-\vec{\theta}) \\ &= E_{pk}(d_1 - \theta, \ldots, d_\ell - \theta) \end{aligned}$$

  4. サーバは，乱数 $\vec{R_0} = (R_{01}, \ldots, R_{0\ell})$, $\ldots, \vec{R_\theta} = (R_{\theta 1}, \ldots, R_{\theta\ell})$（ただし，任意の i, j について $R_{ij} \neq 0$）を生成し，上記で得られた $(C^0, \ldots, C^\theta)$ に対して以下を計算する．

$$\begin{aligned} RC^0 &= C^0 \otimes E_{pk}(\vec{R_0}) \\ &= E_{pk}(R_{01}d_1, \ldots, R_{0\ell}d_\ell) \\ &\vdots \\ RC^\theta &= C^\theta \otimes E_{pk}(\vec{R_\theta}) \\ &= E_{pk}(R_{\theta 1}(d_1 - \theta), \ldots, R_{\theta\ell}(d_\ell - \theta)) \end{aligned}$$

  5. サーバは，$\{(RC^0, \ldots, RC^\theta)\}_C$ を検索者に送る．
  6. 検索者は，各暗号文 C に対する $(RC^0, \ldots, RC^\theta)$ を復号して，$(\vec{M_0}, \ldots, \vec{M_\theta})$ を得る．ここで，$\vec{M_i} = (M_{i1}, \ldots, M_{i\ell})$ とする．

7. 検索者は，各暗号文Cに対するリスト $\texttt{result} = [(1, > \theta), \ldots, (\ell, > \theta)]$ を生成する．
8. 検索者は，$M_{ij} = 0$ となるすべての (i, j) に対して，$\texttt{result}[j] = (j, i)$ と更新する．
9. 検索者は，検索結果として $\{\texttt{result}\}_C$ を得る．

なお，$RC^i (0 \leq i \leq \theta)$ は，乗法的な準同型演算を用いて求められているが，$E_{pk}(\vec{d}) = (E_{pk}(d_1), \ldots, E_{pk}(d_\ell))$ の場合（つまり，$E_{pk}(\vec{d})$ が各類似度の暗号文のベクトルで表わされている場合）は，加法的な準同型演算のみで実行可能である．つまり，乱数 R に対して，以下のようにして $E_{pk}(R \times d)$ を求めることができる．

$$E_{pk}(R \times d) = \underbrace{E_{pk}(d) \oplus \cdots \oplus E_{pk}(d)}_{R}$$

4.2 方式 2

本節で提案する方式 2 では，方式 1 とは異なり，類似度そのものが検索結果として現れず，類似度が閾値以内であるか否かといった事実のみ現れる．方式 2 を以下に提案する．なお，**KeyGen** と **Store** の処理は方式 1 と同様のため省略する．

- **Search:**

1. 検索者は，検索したいパターン $\vec{p} = (p_1, \ldots, p_m)\ (m \leq n)$ に対して，$E_{pk}(\vec{p})$ を計算し，サーバに送る．
2. サーバは，受け取った $E_{pk}(\vec{p})$ と各保管データ C に対して，$E_{pk}(\vec{d})$ を計算する．
3. サーバは，閾値 θ を用いて，各暗号文 C に対して以下を計算する．

$$\begin{aligned} C^0 &= E_{pk}(\vec{d}) \oplus E_{pk}(-\vec{0}) \\ &= E_{pk}(d_1, \ldots, d_\ell) \\ &\vdots \\ C^\ell &= E_{pk}(\vec{d}) \oplus E_{pk}(-\vec{\theta}) \\ &= E_{pk}(d_1 - \theta, \ldots, d_\ell - \theta) \end{aligned}$$

4. サーバは，乱数 $\vec{R_0} = (R_{01}, \ldots, R_{0\ell})$, $\ldots, \vec{R_\theta} = (R_{\theta 1}, \ldots, R_{\theta\ell})$（ただし，任意の i, j について $R_{ij} \neq 0$）を生成し，上記で得られた $(C^0, \ldots, C^\theta)$ に対して以下を計算する．

$$\begin{aligned} RC^0 &= C^0 \otimes E_{pk}(\vec{R_0}) \\ &= E_{pk}(R_{01}d_1, \ldots, R_{0\ell}d_\ell) \\ &\vdots \\ RC^\theta &= C^\theta \otimes E_{pk}(\vec{R_\theta}) \\ &= E_{pk}(R_{\theta 1}(d_1 - \theta), \ldots, R_{\theta\ell}(d_\ell - \theta)) \end{aligned}$$

5. サーバは，置換 $\pi : \{0, \ldots, \theta\} \to \{0, \ldots, \theta\}$ を一様ランダムに選び，$(RC^{\pi^{-1}(0)}, \ldots, RC^{\pi^{-1}(\theta)})$ とおく．
6. サーバは，各暗号文 C に対する $(RC^{\pi^{-1}(0)}, \ldots, RC^{\pi^{-1}(\theta)})$ を検索者に送る．
7. 検索者は，各暗号文 C に対する $(RC^{\pi^{-1}(0)}, \ldots, RC^{\pi^{-1}(\theta)})$ を復号して，$(\vec{M}_0, \ldots, \vec{M}_\theta)$ を得る．ここで，$\vec{M}_i = (M_{i1}, \ldots, M_{i\ell})$ とする．
8. 検索者は，各暗号文 C に対するリスト $\texttt{result} = [(1, > \theta), \ldots, (\ell, > \theta)]$ を生成する．
9. 検索者は，$M_{ij} = 0$ となるすべての (i, j) に対して，$\texttt{result}[j] = (j, \leq \theta)$ と更新する．
10. 検索者は，検索結果として $\{\texttt{result}\}_C$ を得る．

4.3 方式 3

本節で提案する方式 3 では，方式 2 と同様に，類似度そのものが検索結果として現れず，類似度が閾値以内であるか否かといった事実のみ現れる．方式 2 と異なる点として，乗法的な準同型演算を用いることで，暗号文サイズを減らすことができる．方式 3 を以下に提案する．なお，**KeyGen** と **Store** の処理は方式 1 と同様のため省略する．

- **Search:**

1. 検索者は，検索したいパターン $\vec{p} = (p_1, \ldots, p_m)\,(m \leq n)$ に対して，$E_{pk}(\vec{p})$ を計算し，サーバに送る．
2. サーバは，受け取った $E_{pk}(\vec{p})$ と各保管データ C に対して，$E_{pk}(\vec{d})$ を計算する．
3. サーバは，閾値 θ を用いて，各暗号文 C に対して以下を計算する．

$$\begin{aligned} C^0 &= E_{pk}(\vec{d}) \oplus E_{pk}(-\vec{0}) \\ &= E_{pk}(d_1, \ldots, d_\ell) \\ &\vdots \\ C^\ell &= E_{pk}(\vec{d}) \oplus E_{pk}(-\vec{\theta}) \\ &= E_{pk}(d_1 - \theta, \ldots, d_\ell - \theta) \end{aligned}$$

4. サーバは，上記で得られた $(C^0, \ldots, C^\theta)$ を用いて，以下のように $\hat{C}$ を計算する．

$$\begin{aligned} \hat{C} &= C^0 \otimes \cdots \otimes C^\ell \\ &= E_{pk}(\prod_{i=0}^{\theta}(d_1 - i), \ldots, \prod_{i=0}^{\theta}(d_\ell - i)) \end{aligned}$$

5. 乱数 $\vec{R} = (R_1, \ldots, R_\ell)$ を生成し，上記で得られた $\hat{C}$ を用いて，以下を計算する．

$$\begin{aligned} \hat{RC} &= \hat{C} \otimes E_{pk}(\vec{R}) \\ &= E_{pk}(R_1 \prod_{i=0}^{\theta}(d_1 - i), \ldots, \\ &\qquad R_\ell \prod_{i=0}^{\theta}(d_\ell - i)) \end{aligned}$$

6. サーバは，各暗号文 C に対する $\hat{RC}$ を検索者に送る．
7. 検索者は，各暗号文 C に対する $\hat{RC}$ を復号して，$(M_1, \ldots, M_\ell)$ を得る．
8. 検索者は，各暗号文 C に対するリスト $\texttt{result} = [(1, > \theta), \ldots, (\ell, > \theta)]$ を生成する．
9. 検索者は，$M_i = 0$ となるすべての i に対して，$\texttt{result}[i] = (i, \leq \theta)$ と更新する．
10. 検索者は，検索結果として $\{\texttt{result}\}_C$ を得る．

5 安全性

本章では，サーバと検索者の安全性について考察する．

5.1 サーバに対する安全性

サーバは，Semi-Honest と仮定しているため，基本的には暗号化登録データと暗号化検索パターンとの間の準同型演算を行うのみであり，方式 1〜3 では復号されたデータがサーバに現れることは一切ない．つまり，サーバは全く復号オラクルが使えない状況のため，もしベースとなる準同型暗号方式が IND-CPA 安全性を満たしていれば，方式 1〜3 のサーバに対していずれも IND-CPA 安全性が保証できる．

5.2 検索者に対する安全性

検索者は，Semi-Honest と仮定しているため，逸脱した検索パターンを使った攻撃（例えば，検索パターンはバイナリベクトルを想定しているところに対して，整数値ベクトルを検索パターンとして用いるなど）は実施されることはない．

方式 1 では，類似度が閾値以内のポジションについては類似度そのものが現れるため，検索者はその部分に対応した平文が復元できる．一方で，閾値よりも大きな類似度をもつポジションに対応する平文については，類似度がランダム化されているため，検索者は非常に限られた平文の情報しか得ることができない．ただし，類似度が閾値以内のポジションの情報を考慮すると，隠れた平文の値の候補が幾分絞りこめる可能性があることに注意する．

方式 2 と 3 では，方式 1 よりも検索者に対して平文の情報をより制限できている．つまり，方式 1 とは異なり，方式 2 と 3 では，検索者が得られる情報が閾値以内か否かと少なくなっているため，閾値以内のポジションに関連した平文も方式 1 よりも推測しにくくなっており，また閾値よりも大きいポジションに関連した平文もより推測しにくくなっている．

なお，方式 1〜3 に共通して，検索者が非常に短い検索パターンで検索した場合（例えば，0

といった1ビットの検索パターン）は，たとえ閾値以内か否かの情報だけであっても登録データの平文の多くが推測できてしまうため，現実的には，検索パターンはある程度長いものサイズのものしか受け付けないようにするか，もしくは閾値を非常に小さくしなければならない．

6 まとめ

本稿では，秘匿類似検索方式を直接的に用いてゲノムなどのデータ分析に応用するとプライバシーの課題に直面することを示し，準同型性を用いて，そのプライバシーの課題を解決する手法を3つ提案した．今後の課題として，暗号学的に安全性をモデル化して厳密な安全性証明を付けること，またSemi-Honestな検索者とサーバを仮定していたため，Maliciousな攻撃者にも対応できるような方式を検討することである．

参考文献

[1] D. Boneh, E.J. Goh, and K. Nissim. Evaluating 2-dnf formulas on ciphertexts. In *TCC 2005*, volume 3378 of *LNCS*, pages 325–341, 2005.

[2] Z. Brakerski and V. Vaikuntanathan. Fully homomorphic encryption from ring-lwe and security for key dependent messages. In *CRYPTO 2011*, volume 6841 of *LNCS*, pages 505–524, 2011.

[3] M. Chase and E. Shen. Substring-searchable symmetric encryption. In *PETS 2015*, pages 263–281, 2015.

[4] C. Gentry. Fully homomorphic encryption using ideal lattices. In *ACM STOC 2009*, pages 79–88, 2009.

[5] K. Lauter, M. Naehrigi, and V. Vaikuntanathan. Can homomorphic encryption be practical? In *ACM CCSW 2011*, pages 113–124, 2011.

[6] P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *EUROCRYPT 1999*, volume 1592 of *LNCS*, pages 223–238, 1999.

[7] C. Wang, K. Ren, S. Yu, and K. M. R. Urs. Achieving usable and privacy-assured similarity search over outsourced cloud data. In *IEEE INFOCOM 2012*, pages 451–459, 2012.

[8] M. Yasuda, T. Shimoyama, J. Kogure, K. Yokoyama, and T. Koshiba. Secure pattern matching using somewhat homomorphic encryption. In *ACM CCSW 2013*, pages 65–76, 2013.

[9] 縫田 光司, 清水 佳奈, 荒井 ひろみ, 浜田 道昭, 津田 宏治, 広川 貴次, 花岡悟一郎, 佐久間 淳, and 浅井 潔. 加法準同型暗号を用いた化合物データベースの秘匿検索プロトコル. In *CSS 2012*, pages 382–389, 2012.

[10] 小暮 淳, 安田 雅哉, 下山 武司, 小柴 健史, and 横山 和弘. 準同型暗号を用いた秘匿検索. In *SCIS 2014*, pages 1D3–5, 2014.