Vol.2016-MPS-108 No.21
Vol.2016-BIO-46 No.21
2016/7/5

疾患リスクの評価へ向けた加法準同型性暗号によるプライバシー保護HMMの実装と評価 (2016年6月2日版)

三品気吹[†1]　浜田道昭[†1]

概要：ゲノムシーケンシングコストの低下により，疾患に関する遺伝子の研究や，個人ゲノムを用いた診断などが行われるようになってきた．しかしゲノムは個人に関する非常に多くの重要な情報を含んでいるため扱いが難しく，プライバシー上の問題が研究や遺伝子診断の普及の妨げになっている．そこで，本研究では安全にゲノム解析を行う方法として，HMM に対して加法準同型性暗号の一つである Paillier 暗号と，暗号プロトコルの一つである 1-out-of-n 紛失通信を適用した Privacy-preserving HMM のための Secure Forward Algorithm を提案する．Secure Forward Algorithm では，ゲノムを所持する人と HMM を所持する人の二者が，互いの持っている情報を一切共有せずに計算結果のみを得ることができる．暗号化を適用しない通常の Forward Algorithm と比較して，誤差率 0.0465%という高い精度で安全な計算を行うことができた．これによって，プライバシーを保護した状態でゲノムから疾患リスクの評価などを行うことができると考えられる．

Implementation and evaluation of privacy-preserving HMM using homomolphic encryption toward disease risk estimation (version 2016/6/2)

Mishina Ibuki[†1]　Hamada Michiaki[†1]

1. はじめに

次世代シーケンサなどの新技術の開発によりゲノムシーケンシングコストは近年で大幅に減少し，一人の全ゲノムを十万円程度で解読できるようになった．それによって個人ゲノムに関する研究が盛んに行われるようになり，病気の診断や治療，新薬の開発などに役立てられている反面，ゲノムに対するプライバシー保護技術の研究があまり進んでいないという現状がある．

個人ゲノムは「個人の特徴を示す情報」と「個人を識別するための情報」のどちらも含まれている究極の個人情報であり，それ一つから個人に関する非常に多くの情報が手に入る [1]．そのためゲノム解析によって有益な情報を手に入れることができる一方で，その情報を悪用することもできるようになってしまう．例えばゲノム解析によって，ある病気への罹患リスクを評価することができれば病気の早期発見に繋がるが，その情報によって遺伝的差別などの社会的不利益を被る可能性も考えられる．現在ゲノムシーケンサは高価であり一部の研究者などしか使えないが，将来的には低廉化によって多くの人が容易に購入し，使えるようになると考えられるため，ゲノムを安全に取り扱う技術の早急な開発が必要である．そこで，本研究では隠れマルコフモデル (Hidden Markov Model：HMM) と既存の暗号技術を用いてプライバシー保護ゲノム解析を行うことを提案する．

本研究では，入力である個人ゲノムの部分配列を所持した Alice(クライアント)，疾患に関する HMM を所持する Bob(サーバ) の二者がいることを前提とする．入力のゲノ

[†1] 現在，早稲田大学 先進理工学研究科　電気・情報生命専攻
Presently with Waseda Uniersity

ム部分配列と HMM の各パラメータをすべて秘匿した状態で前向き計算 (Forward Algorithm) を実行し，そこで得られた計算結果から疾患リスクの評価を行うことが目標である．Forward Algorithm を実行する際，加法準同型性暗号の一つである Paillier 暗号と，1-out-of-n 紛失通信という暗号プロトコルを用いることで，各データを暗号化したまま計算を行うことを可能にする．

暗号化を用いた疾患リスク評価の手法としては，遺伝的要因と環境的要因に注目してロジスティック回帰を用いて評価を行うものや [2]，SNPs と環境・臨床データに基づいて患者，医療機関，クラウド，ゲノム解析機関の四者で評価を行うシステム [3] などが既に提案されている．これらの手法は個人ゲノム以外にも環境的要因など考慮しているため予測精度は高いが，計算やシステムが複雑になっている．本研究で用いる Forward Algorithm はこれらに比べて計算が簡単であるため，計算コストを抑えることができると考えられる．また HMM という確率モデルを用いることで，アラインメントや文字列マッチングなどの手法を用いるよりも細かな違いに柔軟に対応することができるため，予測精度の向上が期待できる．

また HMM はバイオインフォマティクスで頻繁に用いられ，疾患遺伝子同定 [4] や病気の相互作用解析 [5]，健康データに基づいた生活習慣病リスクの評価 [6] などに応用されている．そのため，暗号化を適用した HMM は今まで主に音声認識の領域で研究されてきたが [7]，バイオインフォマティクスの領域でも本研究のみならず様々な研究に応用できると考えられる．

2. 理論と方法

2.1 隠れマルコフモデル (HMM)

HMM は $\lambda = (\mathbf{A}, \mathbf{B}, \mathbf{\Pi})$ という三個のパラメータで表される確率モデルで，観測列は隠れ状態から出力される (図 1)．この際，隠れ状態はマルコフモデルにしたがっている．観測列 $\mathbf{x}$ と隠れ状態列 $\mathbf{q}$ をそれぞれ式 (1)， 式 (2) のように定式化する．$v_1, v_2, \cdots, v_K$ は観測されうるシンボルの集合であり，$S_1, S_2, \cdots, S_N$ は隠れ状態の集合である．

$$\mathbf{x} = x_1, x_2, \cdots, x_T \in \{v_1, v_2, \cdots, v_K\}^T \quad (1)$$

$$\mathbf{q} = q_1, q_2, \cdots, q_T \in \{S_1, S_2, \cdots, S_N\}^T \quad (2)$$

- $\mathbf{A}$ は状態遷移行列で，$\mathbf{A} = (a_{ij}) = Pr\{q_{t+1} = S_j | q_t = S_i\}$ $(1 \leq i, j \leq N)$．このとき q_t は時間 t にどの隠れ状態にいるかを表している $(1 \leq t \leq T)$.
- $\mathbf{B}$ は出力確率行列で $\mathbf{B} = (b_i(v_k))(1 \leq i \leq N, 1 \leq k \leq K)$.
- $\mathbf{\Pi}$ は初期状態確率ベクトルで $\mathbf{\Pi} = (\pi_i) = Pr\{q_1 = S_i\}(1 \leq i \leq N)$.

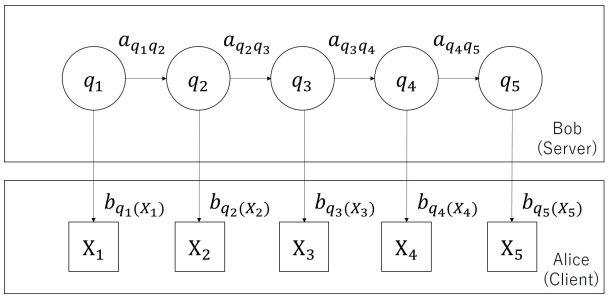

図 1 隠れマルコフモデルの概念図

2.2 Forward Algorithm

Forward Algorithm は隠れ状態に対して周辺化計算を行うことで観測列の生成確率を計算するためのアルゴリズムである．ここで前向き変数を以下の式 (3) で定義する．

$$\alpha_t(j) = Pr\{x_1, x_2, \cdots, x_t, q_t = S_j | \lambda\}(1 \leq j \leq N) \quad (3)$$

Forward Algorithm の計算手順を示す． まず初期状態確率 $\mathbf{\Pi}$ と出力確率 $\mathbf{B}$ を用いて，式 (4) によって前向き変数 $\alpha_t(j)$ の初期化を行う. 出力確率 $b_j(x_1)$ は時間 $t = 1$ において初期状態 S_j からシンボル x_1 が出力される確率を表しているため

$$\alpha_1(j) = \pi_j b_j(x_1), 1 \leq j \leq N \quad (4)$$

となる．次に式 (5) に基づいて $1 \leq j \leq N$ かつ $1 \leq t \leq T$ の範囲で再帰的に $\alpha_t(j)$ の計算を行う．

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(x_{t+1}) \quad (5)$$

最後に式 (6) のように，$t = T$ における前向き変数 $\alpha_T(j)$ を $1 \leq j \leq N$ の範囲で全て足し合わせることによって周辺化確率 $Pr\{x_1, x_2, \cdots, x_T | \lambda\}$ を得る．

$$Pr\{x_1, x_2, \cdots, x_T | \lambda\} = \sum_{j=1}^{N} \alpha_T(j) \quad (6)$$

本研究では加法準同型性暗号で計算できるようにするため，対数変換版 Forward Algorithm を用いる．対数変換版 Forward Algorithm では再帰式 (5) が以下の式 (7) のようになる．

$$\log \alpha_{t+1}(j) = \log \left[\sum_{i=1}^{N} \exp(\log \alpha_t(j) + \log a_{ij})\right] + \log b_j(x_{t+1}) \quad (7)$$

2.3 公開鍵暗号 [8]

暗号化と復号に別々の鍵を用いることで，暗号化と復号に同じ鍵を用いる共通鍵暗号方式での鍵の輸送問題を解決した暗号方式である．

あるデータを暗号化した状態で送受信するとき，受信者は暗号化のための鍵 (公開鍵) と復号のための鍵 (秘密鍵)

を生成し，暗号化のための公開鍵を送信者に送る．この際，公開鍵から秘密鍵に関する知識は一切手に入らないような公開鍵を作らなければならない．送信者は受信者から受け取った公開鍵を用いてデータを暗号化し，受信者に送る．受信者は自分だけが所持している秘密鍵を用いて受け取った暗号文を復号し，平文を手に入れることができる．秘密鍵をやりとりすることがないため，攻撃者は公開鍵と暗号文を傍受することができても復号することができない．

2.4 準同型暗号

準同型暗号は公開鍵暗号の一つで，平文や秘密鍵に関する知識なしに暗号文の加算や乗算を行うことができる性質を持った暗号方式である．

二つの平文 m_1, m_2 に対して公開鍵及び秘密鍵のペアが (pk, sk) であるとき，二つの平文 m_1, m_2 の暗号文は次のように表される (式 (8))．

$$c_1 = E_{pk}(m_1), c_2 = E_{pk}(m_2) \tag{8}$$

ここで暗号系が加法に関して準同型性を持っていた場合，式 (9) を計算することができる．

$$E_{pk}(m_1) \circ E_{pk}(m_2) = E_{pk}(m_1 + m_2) \tag{9}$$

$$m_1 + m_2 = D_{sk}(E_{pk}(m_1 + m_2)) \tag{10}$$

$E_{pk}(m_1 + m_2)$ を復号すると平文の $m_1 + m_2$ を得ることができる式 (10)．また，式 (9) の性質を利用することで式 (11) のような計算もできる．

$$E_{pk}(m)^k = E_{pk}(km) \tag{11}$$

2.5 Paillier 暗号 [9]

Paillier 暗号は加法準同型暗号の一つである．平文 $m \in \mathbf{Z}_n$ に対して Paillier 暗号文 c は式 (12)〜(14) のように定義され，公開鍵と秘密鍵は式 (15) のように表される．p, q はそれぞれ大きな素数， r は乱数で $r \in \mathbf{Z}^*_{n^2}$ である．$\mathbf{Z}_n$ は n 個の元 $\{0, 1, \cdots, n-1\}$ からなる整数の集合であり，$\mathbf{Z}^*_{n^2}$ は n^2 個の元 $\{0, 1, \cdots, n^2-1\}$ からなる集合 $\mathbf{Z}_{n^2}$ から 0 を除いた集合 $\{1, 2, \cdots, n^2-1\}$ である．

$$c = E_{pk}(m) = g^m \cdot r^n \mod n^2 \ (g, r \in \mathbf{Z}^*_{n^2}) \tag{12}$$

$$n = pq \tag{13}$$

$$g = 1 + kn \mod n^2 \ (k \in \mathbf{Z}_n) \tag{14}$$

$$pk = (n, g), sk = (p, q) \tag{15}$$

暗号文を復号する際は式 (16) の計算を行う

$$D_{sk}(E_{pk}(m)) = \frac{L(c^\lambda \mod n^2)}{L(g^\lambda \mod n^2)} \ (\lambda = lcm(p-1, q-1)) \tag{16}$$

次に，二つの平文 m_1, m_2 の暗号化を考えると，それぞれ式 (17) のようになる．

$$E_{pk}(m_1) = g^{m_1} \cdot r_1^n, \ E_{pk}(m_2) = g^{m_2} \cdot r_2^n \mod n^2 \tag{17}$$

式 (12) の形から，暗号文同士の積をとると平文の和が求められることが分かる．$E_{pk}(m_1)$ と $E_{pk}(m_2)$ の積を求めると式 (18) のようになり，$m_1 + m_2$ が得られる．

$$g^{m_\Gamma} \cdot r_1^n \times g^{m_2} \cdot r_2^n \ = g^{m_1+m_2} \cdot (r_1 r_2)^n \mod n^2 \tag{18}$$

2.6 1-out-of-n 紛失通信 [10]

1-out-of-n 紛失通信は暗号プロトコルの一つである．データの送信者が n 個のデータ，データの受信者がそれに紐づく n 個のインデックスを所持しているとき，受信者は n 個のうちの何番目のデータを抜き取ったかを送信者に知られることなく希望のデータのみを得ることができる．

3. 本論

本研究では最終的に Paillier 暗号と 1-out-of-n 紛失通信を適用した Secure Forward Alorithm を実装するため，四個のプロトコルに分けて実装を行ったのち，それらを組み合わせて Secure Forward Algorithm の実装を行う．

本章で説明を行う全てのプロトコルに関して，あらかじめ Alice が公開鍵と秘密鍵の生成を行って Bob に公開鍵を渡しているものとし，また Paillier 暗号文は，以降 $\xi(\cdot)$ と表すものとする．

3.1 Secure Logarithm Protocol

Bob は暗号文 $\xi(\theta)$ を受け取り，θ に関する情報を何も得ないまま $\xi(\log\theta)$ を計算する．計算手順を Algorithm1 に示す．

Algorithm 1 Secure Logarithm Protocol

Require: $\xi(\theta)$
Ensure: $\xi(\log\theta)$
1: Bob は正の整数の乱数 β を選び，$\xi(\theta)^\beta = \xi(\beta\theta)$ を計算して Alice に送る
2: Alice は $\xi(\beta\theta)$ を復号して得た $\beta\theta$ から $\log\beta\theta$ を計算
3: Alice は $\log\beta\theta$ を暗号化し，$\xi(\log\beta\theta)$ を Bob に送る
4: Bob は手順 1 で選んだ β から $\xi(-\log\beta)$ を計算
5: Bob は $\xi(\log\beta\theta) \cdot \xi(-\log\beta) = \xi(\log\theta)$ を計算

実装を行う際には，$\log\beta$ や $\log\theta$ は非常に小さい値となるため 10^6 程度の大きな値をかけることで整数として扱う．これは暗号化するために整数である必要があるためで，第 3 章で説明する全てのプロコトルに関して共通である．

3.2 Secure Exponent Protocol

Bob は暗号文 $\xi(\log\theta)$ を受け取り，θ に関する情報を何

も得ないまま $\xi(\theta)$ を計算する．$\frac{1}{\beta}$ は β の逆数で，$\beta \in Z^*_{n^2}$．計算手順を Algorithm2 に示す．

Algorithm 2 Secure Exponent Protocol

Require: $\xi(\log\theta)$
Ensure: $\xi(\theta)$
1: Bob は正の整数の乱数 β を選んで $\xi(\log\beta)$ を計算し，$\xi(\log\beta\theta)$ を Alice に送る
2: Alice は $\xi(\log\beta\theta)$ を復号して得た $\log\beta\theta$ から $\beta\theta$ を計算
3: Alice は $\beta\theta$ を暗号化し，$\xi(\beta\theta)$ を Bob に送る
4: Bob は $\xi(\beta\theta)^{\frac{1}{\beta}} = \xi(\theta)$ を計算

3.3 Secure Logsum Protocol

Bob は暗号文 $\xi(\log\theta_1), \xi(\log\theta_2), \cdots, \xi(\log\theta_n)$ と定数ベクトル $a_1, a_2, \cdots, a_n$ を受け取り，θ に関する情報を何も得ないまま $\xi(\log\sum_{i=1}^{n} a_i\theta_i)$ を計算する．計算手順を Algorithm3 に示す．

Algorithm 3 Secure Logsum Protocol

Require: $\xi(\log\theta_1), \xi(\log\theta_2), \cdots, \xi(\log\theta_n)$, $a_1, a_2, \cdots, a_n$
Ensure: $\xi(\log\sum_{i=1}^{n} a_i\theta_i)$
1: Bob と Alice は $\xi(\log\theta_1), \xi(\log\theta_2), \cdots, \xi(\log\theta_n)$ に対して Secure Exponent protocol を n 回行い，$\xi(\theta_1), \xi(\theta_2), \cdots, \xi(\theta_n)$ を計算
2: Bob は $\xi(\theta_1), \xi(\theta_2), \cdots, \xi(\theta_n)$ と $a_1, a_2, \cdots, a_n$ から，Paillier 暗号の加法準同型性を利用して $\xi(\sum_{i=1}^{n} a_i\theta_i)$ を計算

$$\xi(\sum_{i=1}^{n} a_i\theta_i) = \prod_{i=1}^{n} \xi(a_i\theta_i) = \prod \xi(\theta_i)^{a_i}$$

3: Alice と Bob は Secure Logarithm Protocol を行い $\xi(\sum_{i=1}^{n} a_i\theta_i)$ から $\xi(\log\sum_{i=1}^{n} a_i\theta_1)$ を計算

3.4 1-out-of-n 紛失通信 [11]

Bob(送信者) が n 個のデータ $\alpha_1, \alpha_2, \cdots, \alpha_n$，Alice(受信者) がそれらのデータに対応する n 個のインデックス $\beta_1, \beta_2, \cdots, \beta_n (\beta_i = 1, \beta_j = 0 (j \neq i))$ を自分で生成する．i は Alice が欲しいデータの番号である．計算手順を Algorithm4 に示す．

Algorithm 4 1-out-of-n 紛失通信

1: Alice はインデックス $\beta_1, \beta_2, \cdots, \beta_n (\beta_i = 1, \beta_j = 0 (j \neq i))$ を作る
2: Alice はインデックスを暗号化し，Bob に送る
3: Bob は受け取ったインデックス $\xi(\beta_1), \xi(\beta_2), \cdots, \xi(\beta_n)$ と自分の持っているデータ $\alpha_1, \alpha_2, \cdots, \alpha_n$ から以下の計算を行う

$$\xi(\beta_1)^{\alpha_1}, \xi(\beta_2)^{\alpha_2}, \cdots, \xi(\beta_n)^{\alpha_n} = \xi(\alpha_1\beta_1), \xi(\alpha_2\beta_2), \cdots, \xi(\alpha_n\beta_n)$$

4: Bob は計算した $\xi(\alpha_1\beta_1), \xi(\alpha_2\beta_2), \cdots, \xi(\alpha_n\beta_n)$ を Alice に送る
5: Alice が受け取った $\xi(\alpha_1\beta_1), \xi(\alpha_2\beta_2), \cdots, \xi(\alpha_n\beta_n)$ を復号すると，i 番目が α_i となり，その他は 0 となっているため希望の i 番目のデータのみを入手できる

3.5 Secure Forward Algorithm Protocol

Secure Forward Algorithm Protocol では，HMM λ と配列 $x_1, x_2, \cdots, x_T$ から $\xi(\log Pr\{x_1, x_2, \cdots, x_T|\lambda\})$ を計算する．Paillier 暗号において計算できるのは加算のみであるため，対数変換版の Forward Algorithm を用いる．またフォワードスコアは小さな値となるうえに小数のままでは暗号化できないため，10^6 程度の大きな値をかけることで整数として扱いアンダーフローを防ぐ．計算手順を Algorithm5 に示す．

Algorithm 5 Secure Forward Algorithm Protocol

Require: 配列 $x_1, x_2, \cdots, x_T$ および HMM $\lambda = (A, B, \Pi)$
Ensure: $\xi(\log Pr\{x_1, x_2, \cdots, x_T|\lambda\})$
1: Bob は乱数 γ を選び，$\log b_j(v_k) + \gamma$ を計算 $(for\ 1 \leq k \leq K, 1 \leq j \leq N)$
2: Alice は自分の $x_1, x_2, \cdots, x_T$ に基づいて，対応する $\log b_j(x_t)+\gamma$ を 1-out-of-n 紛失通信を用いて入手する $(for\ 1 \leq t \leq T)$
3: Alice は受け取った $\log b_j(x_t)+\gamma$ を暗号化し，$\xi(\log b_j(x_t)+\gamma)$ を Bob に送る $(for\ 1 \leq t \leq T, 1 \leq j \leq N)$
4: Bob は $\xi(\log b_i(x_T) + \gamma) \cdot \xi(-\gamma) = \xi(\log b_i(x_T))$ を計算 $(for\ 1 \leq t \leq T, 1 \leq j \leq N)$
5: Bob は $\xi(\log\alpha_1(j)) = \xi(\log\pi_j) \cdot \xi(\log b_j(x_1))$ を計算 $(for\ 1 \leq j \leq N)$
6:

$$\xi(\log\alpha_{t+1}(j)) = \xi(\log\sum_{l=1}^{N} \alpha_t(l)a_{ij}) \cdot \xi(_j(x_{t+1}))$$

に基づいて，Alice と Bob は Secure LOGSUM protocol を用いて $\xi(\log\alpha_T(j))$ を計算 $(for\ 1 \leq t \leq T-1, 1 \leq i, j \leq N)$
7: Secure LOGSUM protocol を用いて，$\xi(\log\alpha_T(j))$ から $\xi(\log\sum_{j=1}^{N}\alpha_T(j)) = \xi(\log Pr\{x_1, x_2, \cdots, x_T|\lambda\})$ を計算

3.6 Forward Algorithm 用いた疾患リスクの評価

本研究では Alice が持っている個人ゲノムの部分配列を観測列の $x_1, x_2, \cdots, x_T \in \{A, T, G, C\}^T$ とし，Bob が所持している HMM $\lambda = (\mathbf{A}, \mathbf{B}, \mathbf{\Pi})$ はある遺伝子疾患に関して既にパラメータ推定を行っているものとする．そのため本研究ではパラメータ推定については扱わず，全て既知であると仮定している．

Forward Algorithm を用いることでゲノム部分配列の生成確率が計算できるため，その値に基づいて疾患リスクの評価を行うことを提案する．その際，ゲノム部分配列や HMM のパラメータに関する情報が漏洩することを防ぐため，Seucure Forward Algorithm を用いる．

4. 実験と考察

1-out-of-n 紛失通信に対する処理時間の測定および Secure Forward Algorithm Protocol に対する処理時間と誤差の測定を行った．本研究で用いたプロトコルは全て Alice と Bob がそれぞれ別のマシンであり，本来は Alice-Bob 間で通信を行うが，本研究では通信を行わずに一つのマシン内で計算を行った．したがって第 4 章で示す実験結果に

は，通信にかかる時間が含まれていない．測定環境を表 1 に示す．

表 1 実験環境

bit 数	OS	メモリ	CPU
64bit	Linux(Ubuntu)	2GB	Core i5 2.60GHz

処理時間の測定は clock 関数をプログラム内に挿入し，Paillier 暗号の bit 数 (鍵長) が 256bit， 512bit， 1024bit，2048bit の場合に対して行い，それぞれ 1000 回分の平均値と標準偏差を計算した．測定の際，プログラム全体の処理時間，鍵生成のみの処理時間，計算部分（鍵生成以外）の処理時間に分けて測定を行った．また比較対象として，暗号化を適用していない Forward Algorithm を実装し，同様に 1000 回の処理時間の測定を行った．そのうえで暗号化を適用した場合と適用していない場合の処理時間の比較を行った．

誤差率の算出方法は (｜計算値 - 理論値｜ / 理論値) × 100 [%] で，1000 回分の値から平均値を求めた．計算値は暗号化を用いて計算を行った時の値で，理論値は暗号化をせずに計算を行ったときの値である．

4.1 1-out-of-n 紛失通信

1-out-of-n 紛失通信について，各 bit 数におけるプログラム全体の処理にかかった時間，鍵生成にかかった時間および計算部分にかかった時間の平均と標準偏差を表 2 に纏めた．以下に示す結果は，Bob と Alice の持っているデータ数及びインデックス数 $n = 15$ の場合の測定結果である．

表 2 1-out-of-n 紛失通信 処理時間 測定結果 [msec]

		全体	鍵生成	計算部分
256bit	平均	16.61	2.129	14.48
	偏差	2.225	1.153	1.753
512bit	平均	32.03	8.785	23.25
	偏差	7.692	7.144	2.392
1024bit	平均	135.3	57.44	77.84
	偏差	51.67	50.89	5.156
2048bit	平均	1020	571.7	448.7
	偏差	513.4	512.0	24.05

次にプログラム全体の時間に占める，鍵生成にかかる時間の割合を各 bit 数ごとに計算したものを以下の表 3 に示す．

表 3 1-out-of-n 紛失通信 鍵生成が全体に占める割合 [%]

256bit	512bit	1024bit	2048bit
12.82	27.42	42.46	56.03

表 2 および表 3 の結果から，bit 数を増やすにつれて鍵生成にかかる時間が大幅に増え，鍵生成にかかる時間のばらつきも大きくなっていることが分かる．bit 数を大きくすると，鍵生成のためにそれだけ大きな素数を探す必要があるため，時間が大きく増加したと考えられる．また bit 数が大きくなればなるほど，すぐに素数が見つかった場合とそうでなかった場合の差が広がるため，ばらつきも大きくなったと考えられる．

4.2 Secure Forward Algorithm Protocol

Secure Forward Algorithm Protocol について，各 bit 数におけるプログラム全体の処理にかかった時間，鍵生成にかかった時間および計算部分の処理時間の平均と標準偏差を表 4 に纏めた．以下に示す結果は，Alice が持っている入力の配列長 $T = 5$，Bob の持っている HMM の隠れ状態数 $n = 5$ の場合の測定結果である．

表 4 処理時間 測定結果 [msec]

		全体	鍵生成	計算部分
256bit	平均	226.5	2.915	223.6
	偏差	33.08	1.593	32.76
512bit	平均	375.7	11.36	364.3
	偏差	50.08	8.809	48.45
1024bit	平均	1460	85.30	1374
	偏差	197.9	81.20	170.5
2048bit	平均	8666	820.2	7845
	偏差	931.5	781.2	513.0

プログラム全体の時間に占める鍵生成にかかる時間の割合を各 bit 数ごとに計算したものを以下の表 5 に示す．

表 5 Secure Forward Algorithm Protocol 鍵生成が全体に占める割合 [%]

256bit	512bit	1024bit	2048bit
1.287	3.023	5.844	9.464

表 4 と表 5 から，2048bit の場合でも 8.7 秒程度で計算ができていることが分かる．また bit 数を増やすと鍵生成にかかる時間が全体に占める割合が増加する傾向は 1-out-n-紛失通信と同じで、原因も同じであると考えられる．

次に Secure Forward Algorithm Protocol の誤差を以下の表 6 に示す．

表 6 誤差 測定結果 [%]

256bit	512bit	1024bit	2048bit
0.0465	0.0465	0.0465	0.0465

どの bit 数でも誤差率の平均値は同じ値となり，暗号の bit 数と誤差の相関関係は見られなかった．計算の途中で小数から整数へ直す際に切り捨てが生じていることが誤差の主な原因であると考えられる．今回は 10^6 をかけて整数化したが，この数字の桁数を増やすことによって更に誤差

を減らせる可能性はある．

最後に Secure Forward Algorithm Protocol と，暗号化を適用していない Forward Algorithm の処理時間の比較を行った．暗号化を適用した場合の計算時間が，暗号化を適用していない場合の何倍になるかを計算したものを以下の表 7 に示す．暗号化を適用しない場合の処理時間は，1000 回の平均で 0.0426[msec] であった．

表 7 Secure Forward Algorithm Protocol 暗号化無しとの比較 [倍]

256bit	512bit	1024bit	2048bit
5318	8820	3427×10	2034×10^2

2048bit では暗号化した場合としない場合で 20 万倍程度の差が出たが，暗号化していない場合の処理時間が非常に短いことと，暗号化を適用しても 8.7 秒程度で終わっていることから Secure Forward Algorithm も十分に実用的であると考えられる．

5. 結論・今後の展望

暗号化を適用した Forward Algorithm に対して処理時間と誤差の測定を行った結果，誤差率 0.0465%という高い精度で計算を行うことができた．また計算時間に関しては十分に実用的であり，パソコンの性能次第で更なる短縮も可能であると考えられる．しかし，本研究では入力の配列長 $T = 5$，隠れ状態数 $N = 5$ という小さな値での実験を行ったが，実際に疾患リスクの評価を行うと想定した場合，配列長が数 100 程度必要となってくる．Forward Algorithm では計算量が $O(N^2T)$ であるため，入力の配列長や隠れ状態数によっては処理時間が大幅に増加すると考えられる．配列長や状態数を増やした場合についても追加で実験を行い，その際の処理時間の変化についても関係を調べる必要がある．

今後の課題は本研究で省いた通信部分の実装をし，通信時間も含めた処理時間の測定を行うことである．現在 TCP/IP を用いて通信部分の実装を行っている．また疾患リスクの評価を行うためには，適切なパラメータの推定と状態数を考えることも必要となる．パラメータの推定には，ゲノムを秘匿したまま計算することができる Secure Baum-Welch Algorithm の実装が必要となってくるが，本研究で Secure Forward Algorithm の実装を行ったため，Secure Baum-Welch Algorithm の実装もできる考えられる．既に Secure Baum-Welch Algorithm のプロトコルの提案を行っている論文がある [12]．

また本研究の応用として，profile HMM を用いた場合の Secure Forward Algorithm の研究を現在行っている．profile HMM は pfam の検索システム HMMER でタンパク質のドメイン解析に用いられている，配列をアラインメントするための HMM である [13]．本研究で用いた HMM ではギャップを扱うことができないが，profile HMM では挿入，欠失，一致の状態を持っているためギャップを扱うことができ，HMM より更に柔軟である [14]．

6. おわりに

HMM はバイオインフォマティクスで頻繁に用いられる確率モデルであるため，プライバシー保護 HMM も今後バイオインフォマティクスの領域で様々な応用が期待できる．今後もプライバシー保護 HMM に関する研究を続けていきたい．

参考文献

[1] 佐久間淳. “ ゲノムとプライバシー. ” IEICE ESS Fundamentals Review 7.4 (2014): 348-364.
[2] 呉双, et al. “ 準同型性暗号に基づいたプライバシ保護オンラインロジスティック回帰 (機械学習). ” 電子情報通信学会技術研究報告. IBISML, 情報論的学習理論と機械学習 113.139 (2013): 67-74.
[3] J.-P. Hubaux, J. Fellay, E. Ayday, M. Laren, J. L. Raisaro, P. Jack, et al, “ Privacy-preserving computation of disease risk by using genomic, clinical, and environmental data, ”In Proceedings of USENIX Security Workshop on Health Information Technologies (HealthTech ” 13), number EPFL-CONF-187118, 2013.
[4] Sreekala, S., and K. A. Nazeer. “ A literature search tool for identifying disease-associated genes using Hidden Markov model. ”Computational Systems and Communications (ICCSC), 2014 First International Conference on. IEEE, 2014.
[5] Sherlock, Chris, et al. “ A coupled hidden Markov model for disease interactions. ” Journal of the Royal Statistical Society: Series C (Applied Statistics) 62.4 (2013): 609-627.
[6] Kawamoto, Ryouhei, et al.“ Hidden Markov model for analyzing time-series health checkup data. ”MedInfo. 2013.
[7] Pathak, Manas, et al. “ Privacy preserving probabilistic inference with hidden Markov models. ”Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on. IEEE, 2011.
[8] 光成滋生 (2015)「クラウドを支えるこれからの暗号技術」(秀和システム) pp19-41
[9] Paillier, Pascal.“ Public-key cryptosystems based on composite degree residuosity classes. ”Advances in cryptology—EUROCRYPT ’ 99. Springer Berlin Heidelberg, 1999. p.223-238.
[10] Naor, Moni, and Benny Pinkas. “ Oblivious polynomial evaluation. ” SIAM Journal on Computing 35.5 (2006): 1254-1281.
[11] Shimizu, Kana, Koji Nuida, and Gunnar Rtsch. “ Efficient Privacy-Preserving String Search and an Application in Genomics. ” Bioinformatics. 2016 Mar 2. doi: 10.1093/bioinformatics/btw050
[12] Nguyen, Huan X., and Matthew Roughan. “ Multi-observer privacy-preserving hidden markov models. ”Signal Processing, IEEE Transactions on 61.23 (2013): 6010-6019.
[13] “ Pfam ”(オンライン), 入手先 (http://pfam.xfam.org/)(参照 2016-5-29)
[14] Richard Durbin 他 著　阿久津達也 他 訳 (2001)「バイオインフォマティクス-確率モデルによる遺伝子配列解析-」(医学出版) pp120-159