6U-02

# 分散処理の導入による 完全準同型暗号を用いたゲノム秘匿検索の高速化

山田 優輝 † 小口 正人 †

† お茶の水女子大学

## 1. はじめに

近年ヒトゲノムの解析と応用が可能になり，バイオインフォマティクスをはじめとする様々な分野でゲノムデータ利用の実用化が注目されている．ゲノムデータの活用には大型の計算機とストレージが必要になるため，データをクラウドに預け，利用者の問い合わせを受けてクラウド上で演算を行う委託システムが今後普及していくと考えられる．しかしクラウドは安全性が低く，またゲノム情報は変更できない重要な個人情報であるため，プライバシ保護のための暗号化処理が必須となるが．従来の共通鍵暗号方式でこのシステムの実装を試みた場合，クラウド上でゲノムデータを復号することが可能となるため，データの秘匿には適さない．

特にバイオインフォマティクスの研究において頻繁に行われる検索演算についてこのシステムを実現するために，先行研究 [1]-[2] では暗号化されたデータ同士での演算が可能な完全準同型暗号（以下 FHE: Fully Homomorphic Encryption）を用いる秘匿検索手法を提案しその高速化に取り組んでいるが，依然としてサーバ側で行われる秘匿検索演算の計算量が課題となっている．そこで本研究では，先行研究のサーバ側で行われる FHE 演算にデータベースの分割による分散処理を適用し，実用化に向けた高速化を目指す．

## 2. 先行研究

FHE は暗号文同士の加算と乗算を両方可能としているが，演算のたびに暗号文に含まれるランダムなノイズが増加し，閾値を越えると複号できなくなるという問題点も持っている．これに対しては，演算回数の限定，または bootstrapping と呼ばれるノイズ削減手法の導入，という二つの解決手法が考えられるが，石巻ら（2016）は後者を用い，従来の FHE を用いたゲノム秘匿検索手法に bootstrapping を導入しその演算の最適化を行うことで，計算量の削減を行った [1]．石巻らのシステムはサーバとクライアントが 1:1 で問い合わせを行うもので，サーバはクライアントから検索したい文字列を暗号化処理したものとその文字列を検索したい配列上のポジションを受け取り，秘匿検索を行ってマッチしたか否かの結果を返す．また，山本ら（2017）は従来のゲノム検索手法に分散処理を導入することによる高速化を試みている [2]．山本らのシステムには bootstrapping が導入されておらず問い合わせごとの演算回数が限られるため，クライアントはクエリを一文字ずつ暗号化して問い合わせを行い，サーバから受け取った演算結果同士の比較によって最終的な結果を得る．このサーバ側での演算に対してデータベースの分割によるマスタ・ワーカ型の分散処理を適用することで，高速化が行われている．

## 3. 提案手法

### 3.1 提案システム概要

本研究では先行研究に基づき，以下の FHE を用いたゲノム秘匿検索のマスタ・ワーカ型の分散システムを提案する．図 1 に提案システムの概要を示す．

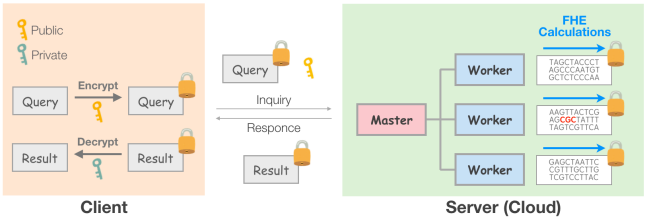


図 1: 提案手法概観

(1) クライアントはクエリ全文を暗号化し，公開鍵などと共にマスタへ送信する．

(2) マスタは受け取ったデータを各ワーカに転送する．

(3) 各ワーカは FHE を用いた秘匿検索演算を行う．また，bootstrapping によるノイズのリセットを適宜行う．

(4) 秘匿検索演算終了後，各ワーカは結果をマスタへ転送する．

(5) マスタは結果を収集し，クライアントへ結果を送信する．

(6) クライアントは復号を行い，結果を得る．

以上のプロトコルを C++ で実装した．

また，完全準同型暗号計算には HElib[3] を，分散時における各マシンの制御のための MPI(Message Passing Interface) を利用するライブラリとして， Open MPI[4] を用いた．

Speeding Up Genome Secure Search Implemented with Fully Homomorphic Encryption by Introduction of Distributed Processing

† Yuki YAMADA, † Masato OGUCHI
Ochanomizu University (†)

### 3.2 分散方法

検索アプリケーションの分散化手法としては，データベースの分割による分散処理，独立性の高い計算部位に適用する分散処理，独立性の高い手順に適用する分散処理，などが挙げられるるが，山本らによる先行研究 [2] では分散化した各ワーカマシンにデータベースを設置する，というデータベースの分割による分散処理が適用され，これにより同時により多くのクエリとデータベース間のマッチング有無を調べることが可能となっている．また石巻らによる先行研究 [1] のプロトコルでは直前の演算結果を用いた演算を繰り返し行うため，計算部位や処理手順に分散処理を適用するのは非効率的である．したがって本研究の提案システムにおいても，データベースの分割による分散処理を適用した．

## 4. 実験

### 4.1 実験環境

ワーカとして同スペックのマシンを複数用い，プログラムを実行した．各マシンの環境は，Intel®Xeon ®Processor E5-2643 v3 3.4GHz，6 コア，12 スレッド，メモリ容量 512GB，ストレージは RAID0 の SSD が 480GB，HDD が 2TB であり，これを 4 台使用する．各マシン上で最大 2 スロットのワーカを稼働させ，そのうち 1 スロットにマスタの機能を持たせた．よって最大で 7 スロット分のワーカを稼働させてワーカ数ごとの実行時間を比較する実験を行った．

実験に使用するゲノムデータは 1 サンプルあたり 10,000 文字のデータを 200 サンプル用意した．また，検索クエリは長さ 5 文字のものを用いた．さらに秘匿検索の秘匿性を高めるためにダミーの検索開始ポジションを加えることにより，1 クエリにつきデータベースの 30 箇所を始点とした文字列検索を行った．

### 4.2 実験結果

クエリとデータベースとの間でマッチングの有無を判定するプログラムを分散化した環境上で稼働させた．実験を 3 回ずつ行った際の，ワーカ数ごとのマスタにおける平均実行時間のグラフを図 2 に示す．

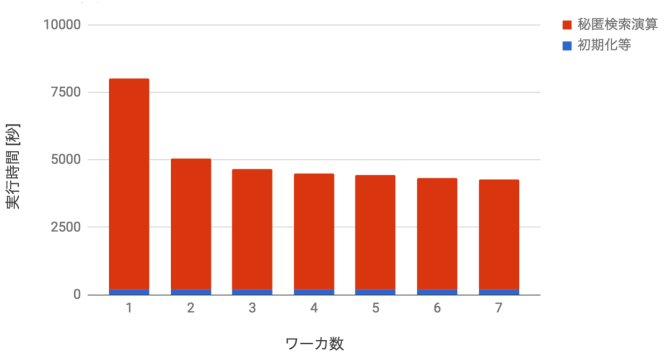


図 2: ワーカ数ごとのマスタにおける平均実行時間 (秒)

マスタでの実行時間において，ワーカ数の増加とともに計算時間が減少することが確認された．クエリ等の送受信やワーカとのデータ共有等を含む初期化に要する時間はワーカ数に応じて増加するが，秘匿検索演算の実行時間と比較するとオーバーヘッドは十分に小さい．

また，ワーカ数が増えるに連れて，分散処理の導入による計算時間の減少効果は徐々に横ばいになっている．これは計算手順の中にデータベースの大きさには依存しない計算量の処理が含まれているためである．

## 5. まとめと今後の課題

実験により，FHE 及び bootstrapping を用いたゲノム秘匿検索のサーバ側での処理にマスタ・ワーカ型の分散処理を適用すると，分散台数に応じて計算時間が減少することが示された．

今後は実装の改善により更なる高速化に取り組むとともに，実用化を目指して実際によく行われる操作に特化した高速化処理なども検討していきたい．

## 謝辞

本研究を進めるにあたり，大変有益なアドバイスを頂いた早稲田大学山名研究室及び工学院大学山口研究室の皆様に感謝いたします．

特に早稲田大学山名研究室所属の石巻さん及びお茶の水女子大学小口研究室所属の山本さんからは，ゲノム秘匿検索システムのプログラムと多くの助言を賜りました．深く感謝いたします．

また本研究は，JST CREST JPMJCR1505 の支援を受けております．

## 参考文献

[1] Y. Ishimaki, H. Imabayashi, K. Shimizu and H. Yamana : ”Privacy-preserving string search for genome sequences with FHE bootstrapping optimization,” 2016 IEEE International Conference on Big Data (Big Data), Washington, DC, 2016, pp. 3989-3991.

[2] Y. Yamamoto and M. Oguchi : ”A Decentralized System of Genome Secret Search Implemented with Fully Homomorphic Encryption,” 2017 IEEE International Conference on Smart Computing (SMARTCOMP), Hong Kong, 2017, pp. 1-6.

[3] Shoup V. and Halevi S. : http://shaih.github.io/HElib/index.html, 2017 年 12 月閲覧

[4] Open MPI : https://www.open-mpi.org/, 2017 年 12 月閲覧