完全準同型暗号を用いた FP-growth によるデータマイニングのアルゴリズム改良手法の提案と評価

理学専攻・情報科学コース　1940652　種村 真由子　(指導教員：小口正人)

1　はじめに

近年，ビッグデータの収集・分析などが，ビジネスを中心とする多くの分野で進んでいる．大規模なデータを扱う処理を行う際，処理能力の高い計算機システムを用意する事が困難な場合，クラウド等の外部の計算資源を利用する方法があるが，使用するデータの機密性が高い場合は特に管理に注意する必要がある．

ここでプライバシ保護のため，外部に送信するデータを完全準同型暗号 (Fully Homomorphic Encryption, FHE) で暗号化することを考える．FHE は，暗号文同士の加算と乗算が成立する公開鍵暗号であり，委託先サーバに復号鍵を渡さず，統計処理を行うことが可能となる．FHE の応用例として，頻出パターンマイニングをクライアント・サーバ型のシステムで実現した，Liu ら (2015) の P3CC(Privacy Preserving Protocol for Counting Candidates)[1] がある．また，P3CC を SV(Smart-Vercauteren) パッキングや暗号文のキャッシュを利用し高速化した例 [2][3] や，サーバ側の分散処理化 [4] を行った例もある．

本研究では，先行研究において使用していた頻出パターンマイニングのアルゴリズムを変更したシステムの実装を行い，その改善を目標とし，パラメータを変化させた際の挙動の変化を調査，新たな実装を提案した．

2　完全準同型暗号

完全準同型暗号 (FHE) は，加法準同型性と乗法準同型性の特徴をあわせ持った，暗号化した状態での暗号文同士の加算，乗算が成立する公開鍵暗号の一方式である．FHE の概念そのものは Rivest ら (1978) によって提案されており，Gentry(2009) が実現手法を提案した [5]．各暗号文には，暗号の解読不可能性を高めるため，ランダムなノイズが付加されている．一般に暗号文と鍵のデータサイズが大きいことにより処理の計算量が膨大になる．また，ノイズの値が暗号文同士の計算を行うたびに増加し，閾値を超えると復号不可となること，比較演算が困難であるという特徴がある．FHE の実装の 1 つとして，Brakerski(2014) らによって提唱された Leveled FHE がある．これは決まった深さ L の論理回路の結果を評価することができる．本研究ではこれを使用する．復号不可になることなくより多くの計算を行うためには，暗号文のパラメータとして高いレベルを設定する必要があるが，その分暗号文のサイズも大きくなるという特徴がある．

3　頻出パターンマイニング

3.1　概要

頻出パターンマイニングは，データマイニングの一種であり，大量のデータの中から，頻出であるパターンを抽出し，相関ルールを見出すことを目的とした手法である．頻出であることの判定には，サポート値（全体のトランザクション数に対する，あるパターンが含まれるトランザクション数の割合）を用い，各パターンのサポート値があらかじめ指定した最小サポート値以上であれば頻出とする．代表的なアルゴリズムとして，先行研究で使用されている Apriori と，本研究で使用している FP-growth がある．

3.2　FP-growth

FP-growth は，データベースを走査し，トランザクションデータを FP-tree という prefix tree の木構造に格納，その部分木を再帰的に走査することで結果を求めるアルゴリズムである．頻出アイテムセットの候補を列挙しない点で Apriori と大きく異なる．頻出アイテムの列挙がボトルネックになる Apriori と比較して，探索を効率化できるという期待がある．

4　システム

クライアント・サーバ型のデータの委託処理システムについて，Fp-growth を使用するプログラムを実装した．

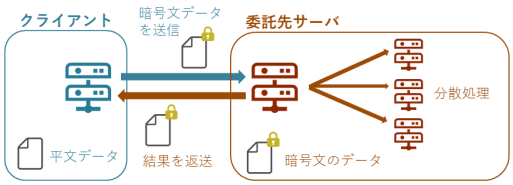

図 1: システムの概要

サーバ側はマスタ・ワーカ型の分散・並列処理を行うことが可能である．FHE を扱うためのライブラリは HElib を使用している．このプログラムにはバージョン 1 と 2 が存在し，バージョン 2 での処理手順は以下の通りである．これらは，頻出を判断する最小サポート値との比較の処理の一部が異なる．

1. サーバ側で通信の受付を行い，クライアントと接続する．
2. クライアントでデータを FHE で暗号化し，最小サポート値に対応する出現回数の閾値と共にサーバに送信する．
3. サーバで暗号文データを受信し，各アイテムのサポート値を計算し，閾値との差分をとった結果をクライアントに返送する．
4. クライアントで結果ファイルを復号し，アイテムごとのサポート値が閾値を超えていたアイテムを抽出し，FP-tree を構築する．
5. FP-tree の走査を行い，結果を出力する．

5　実験

5.1　概要

各項目について，人工的に作成した入力データ（アイテム数 30，トランザクション数 9900）を用いて，実行時間等の挙動を測定した．また，使用したマシンは以下の通りである．分散処理には，以下の表 1 に示す同型のマシンを複数用いる．

表 1: 実験で用いた計算機の性能

OS	CentOS 6.9
CPU	Intel® Xeon® プロセッサ E5-2643 v3 3.6GHz 6 コア　12 スレッド
メモリ	512GB

5.2　結果

また，先行研究との実行時間の比較，FP-growth を使用したプログラム（バージョン 1）の実行時間，リソース使用量等を測定した．全体としてまた，この実行時間の差は暗号文に事前に設定したレベルが大きく関係するという見当から，暗号文のレベルを変化させた際の挙動を測定した．概ねレベルの増加により実行時間は増加するが，例外となるレベルが存在するという結果となった．続いて，FP-growth を使用したプログラム（バージョン 2）で追加された処理についての実行時間の測定を行った．以下の図は，その結果の抜粋である．

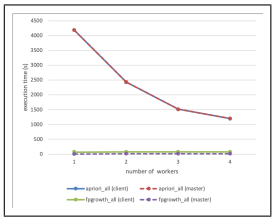

図 2: 分散環境下の Apriori と FP-growth を使用したプログラムの実行時間の比較

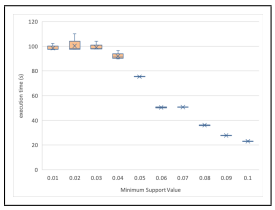

図 3: 最小サポート値を変化させた際の FP-growth を使用したプログラムの実行時間

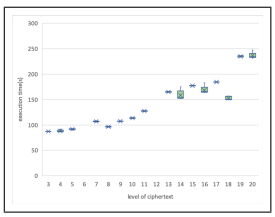

図 4: 暗号文のレベルを変化させた際の実行時間

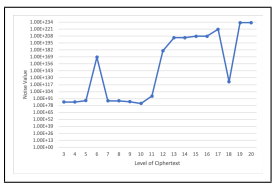

図 5: 暗号文のレベルを変化させた際のノイズ量

6　発展課題 - FP-growth*を使用した実装の提案

6.1　概要

前章までで示した FP-growth のシステムについて，FP-tree を走査する過程において特に多くの計算コストがかかっていることが分かっている．これは，条件付き FP-tree を再帰的に生成することの負荷が大きいためであると考えられる．そこで，FP-growth を元にしたアルゴリズムである，FP-growth*を取り入れたシステムを提案する．ここでは処理の大きな流れは変更せず，現在実装されているシステム内の FP-tree に，以下の 5.2 章に示す Array のデータ構造を紐づけることを検討する．

6.2　FP-growth*

FPgrowth*は，FP-growth において計算コストの高い FP-tree の走査の過程を効率化するために，Array という新しい構造を組み込んだアルゴリズムである [6]．各 FP-tree に対応して Array が一つ作成され，条件付けされた FP-tree を新たに構築する処理を効率化する．疎なデータに対して効果があり，その反面，密なデータに対しては逆に計算コストがかかる場合がある．

7　まとめと今後の課題

FP-growth を使用した頻出パターンマイニングのプログラムについて，先行研究のプログラムとの挙動の比較や，パラメータを変化させた際の実行時間等の測定を行った．また，このシステムにおいて処理の負荷が大きい FP-tree の再帰的な走査部分において，FP-growth*を新たに組み込んだシステムの提案を行った．今後の課題として，セキュリティに関するパラメータの適切な設定，入力データを変化させた測定を行うと同時に，FP-growth*を使用したプログラムの実装を進めていく．

参考文献

[1] J. Liu, J. Li, S. Xu, and B. CM Fung, ”Secure outsourced frequent pattern mining by fully homomorphic encryption”, In International Conference on Big Data Analytics and Knowledge Discovery, pp. 70–81. Springer, 2015.

[2] 高橋卓巳, 石巻優, 山名早人, ”SV パッキングによる完全準同型暗号を用いた安全な委託 Apriori 高速化,” DEIM Forum 2016 F8-6, 2016.

[3] 今林広樹, 石巻優, 馬屋原昂, 佐藤宏樹, 山名早人, ”完全準同型暗号による安全頻出パターンマイニング計算量効率化,” 情報処理学会論文誌データベース (TOD), Vol. 10, No. 1, 2017.

[4] 山本百合, 小口正人, ”完全準同型暗号を用いた秘匿データマイニング計算のデータベース更新時の分散処理による高速化,” DICOMO2018, 2018.

[5] C. Gentry, ”A Fully Homomorphic Encryption Scheme, Doctoral dissertation,” 2009.

[6] GRAHNE, Gösta; ZHU, Jianfei. Efficiently using prefix-trees in mining frequent itemsets. In: FIMI. 2003. p. 65.