3I2-OS-20-4

秘匿計算上の結合アルゴリズム

An Equijoin Algorithm on Secure Function Evaluation

濱田 浩気 Koki HAMADA　菊池 亮 Ryo KIKUCHI　五十嵐 大 Dai IKARASHI　千田 浩司 Koji CHIDA

日本電信電話株式会社
NTT Corporation

1. はじめに

近年，個人に関する様々な情報を容易に取得できる環境の進歩が目覚ましく，活用のニーズが高まっている．その一方で，個人情報保護やプライバシーの観点から，このような情報は極めて慎重な取扱いが必要とされる．この問題の有力な解決手段として，入力となるデータを秘匿したまま各種計算を実現する秘匿計算 [Yao 86] が注目されている．

秘匿計算では，回路を構成することにより，データを秘匿したまま任意の関数を計算できることが知られている．しかし，回路に基づいた一般的な構成方法では実用上重要なアルゴリズムの多くで計算効率が著しく低下してしまうという課題がある．このため，特定のアルゴリズムを秘匿計算で効率よく実現する研究が進んでおり，k 番目の要素の選択やビット分解，比較，剰余計算，ソートなどが提案されている．

データ処理で重要な計算の一つに，等結合がある．等結合は，複数の表として与えられたデータを入力とし，キーと呼ばれる属性の値で名寄せを行った新しい表を得る計算である．関係データベースによるデータ処理のように，複数の表に分割されたデータを統合して分析する際に中心的な役割を果たす．

等結合を秘匿計算で実現する手法は志村らによって提案されている [志村 08]．志村らの手法はレコード数が m, n の 2 つの表の等結合を行う手法である．等結合後の表の大きさをも秘匿したまま計算可能であるが，計算後の表は空を表すレコードを含み，大きさが mn になってしまう．このため，複数の表を対象とした SQL クエリのように等結合を行った結果にさらに計算を行う場合や，3 つ以上の表の等結合を行うために 2 つの表の等結合を繰り返し実行する場合に非効率である．

1.1 本研究の貢献

本稿では，3 つ以上の表に対しても適用可能で，計算結果が等結合後の表に含まれるレコードのみからなるという特徴を持つ，秘匿計算上で等結合を実現するアルゴリズムを提案する．提案する等結合アルゴリズムは入力の表のレコード数の総和を m とすると $O(m \log m)$ の通信量で実現可能である．

1.2 関連研究

等結合と関連が大きい処理に，複数の集合を入力として積集合を計算する処理がある．実用上の重要性から，データを秘匿したまま積集合の計算を行う手法の研究は数多く行われており，大きく二つのアプローチに分類できる．

連絡先: 濱田 浩気，NTT セキュアプラットフォーム研究所，東京都武蔵野市緑町 3-9-11，Tel:(0422)59-3550，Fax:(0422)59-4015，E-mail:hamada.koki@lab.ntt.co.jp

一つは積集合を計算する専用のプロトコルを構成するアプローチである．可換ハッシュ関数を用いて集合の要素のハッシュ値で突合せを行う手法 [Agrawal 03] や，準同型暗号で集合の要素を根に持つ多項式を計算する方法 [Freedman 04]，紛失擬似ランダム関数 [Freedman 05] を用いた手法 [Hazay 08] などがある．[Agrawal 03] のように等結合をも実現可能な手法もあるが，等結合後にさらに計算を行うことは難しい．

もう一つは秘匿計算上でアルゴリズムを構成するアプローチである．Huang らによってレコード数がともに m の 2 つの集合の積集合を計算する手法 [Huang 12] が提案されている．通信量は $\Theta(m \log m)$ だが，等結合への拡張は非自明である．

2. 準備

本稿で提案する等結合アルゴリズムは，秘匿計算上の特定の演算の組み合わせにより実現される．提案アルゴリズムが必要とする演算は，秘匿化，復元，加算，乗算，等号判定，ランダム置換，安定ソートである．これらを提供する秘匿計算を実現するためには，例えば，秘匿化，復元，加算，乗算を提供する [五十嵐 11] の秘匿計算上で [Cramer 01] の等号判定，[濱田 10] のランダム置換，[濱田 11] の安定ソートを用いればよい．

本節では，本稿で用いる記法と秘匿計算が提供する各演算の説明を行う．

2.1 記法

行列 M の i 行目の行ベクトルを $M[i]$，2 つの列ベクトル $\boldsymbol{u}, \boldsymbol{v}$ を縦に結合したベクトルを $\boldsymbol{u}||\boldsymbol{v}$ と表記する．誤解を招く恐れのない場合は，$\boldsymbol{v}[j]$ で列ベクトル $\boldsymbol{v}$ の j 番目の要素を参照する．

2.2 秘匿計算の提供する演算

秘匿化，復元: 対象とする秘匿計算の有限環を R とする．$a \in R$ を秘匿化した値を a の秘匿文と呼び，$[\![a]\!]$ と表記する．ベクトル $\boldsymbol{v}$，行列 M の各要素を秘匿化したベクトル，行列をそれぞれ $[\![\boldsymbol{v}]\!]$，$[\![M]\!]$ と表記する．秘匿文 $[\![a_1]\!], \ldots, [\![a_n]\!]$ から $a_1, \ldots, a_n$ を復元する計算を $a_1, \ldots, a_n \leftarrow \mathsf{Reveal}([\![a_1]\!], \ldots, [\![a_n]\!])$ と表記する．

加算，乗算，等号判定: 2 つの値 $a, b \in R$ の秘匿文を入力とし，それぞれ $a + b$，ab，$a \stackrel{?}{=} b$ の計算結果 c の秘匿文を計算する．但し，$a \stackrel{?}{=} b$ は $a = b$ のとき 1，そうでないとき 0 である．それぞれ $[\![c]\!] \leftarrow \mathsf{Add}([\![a]\!], [\![b]\!])$，$[\![c]\!] \leftarrow \mathsf{Mul}([\![a]\!], [\![b]\!])$，$[\![c]\!] \leftarrow \mathsf{Eq}([\![a]\!], [\![b]\!])$ と表記する．また，$\mathsf{Add}(\mathsf{Add}(\cdots \mathsf{Add}([\![a_1]\!], [\![a_2]\!]) \cdots, [\![a_{n-1}]\!]), [\![a_n]\!])$ を $\sum_{i=1}^{n} [\![a_i]\!]$ と略記する．

Algorithm 1 等結合

入力: $[\![T_1]\!], \ldots, [\![T_\ell]\!]$.
出力: $[\![R_1]\!], \cdots, [\![R_\ell]\!]$.

1: **for each** $i \in \{1, \ldots, \ell\}$ **do in parallel**
2: $\quad [\![T'_i]\!] \leftarrow \mathsf{Shuffle}([\![T_i]\!])$.
3: **end for**
4: $m := \sum_{i=1}^{\ell} m_i$, T'_i $(1 \le i \le \ell)$ の 1 列目を $\boldsymbol{k}'_i$ とする.
5: m 個の互いに異なる 0 でない値を作り，大きさがそれぞれ m_i の ℓ 本の列ベクトル $\boldsymbol{t}_i$ $(1 \le i \le \ell)$ の各要素とする.
6: 各 $\boldsymbol{t}_i$ を秘匿化して $[\![\boldsymbol{t}_i]\!]$ $(1 \le i \le \ell)$ を作る.
7: $[\![\boldsymbol{k}']\!] := [\![\boldsymbol{k}'_1]\!]||\cdots||[\![\boldsymbol{k}'_\ell]\!]$, $[\![\boldsymbol{t}']\!] := [\![\boldsymbol{t}_1]\!]||\cdots||[\![\boldsymbol{t}_\ell]\!]$ とする.
8: $[\![\boldsymbol{k}]\!], [\![\boldsymbol{t}]\!] \leftarrow \mathsf{StableSort}([\![\boldsymbol{k}']\!]; [\![\boldsymbol{k}']\!], [\![\boldsymbol{t}']\!])$.
9: **for each** $j \in \{1, \ldots, m\}$ **do in parallel**
10: $\quad [\![\boldsymbol{e}]\!][j] \leftarrow \begin{cases} \mathsf{Eq}([\![\boldsymbol{k}]\!][j], [\![\boldsymbol{k}]\!][j+\ell-1]) & \text{if } j+\ell-1 \le m, \\ [\![0]\!] & \text{otherwise.} \end{cases}$
11: $\quad [\![\boldsymbol{x}]\!][j] \leftarrow \mathsf{Mul}([\![\boldsymbol{t}]\!][j], [\![\boldsymbol{e}]\!][j])$.
12: $\quad [\![\boldsymbol{g}]\!][j] \leftarrow \sum_{i=\max\{1, j-\ell+1\}}^{j} [\![\boldsymbol{x}]\!][i]$.
13: **end for**
14: $[\![\boldsymbol{t}'']\!], [\![\boldsymbol{g}'']\!] \leftarrow \mathsf{Shuffle}([\![\boldsymbol{t}]\!], [\![\boldsymbol{g}]\!])$.
15: $\boldsymbol{t}'', \boldsymbol{g}'' \leftarrow \mathsf{Reveal}([\![\boldsymbol{t}'']\!], [\![\boldsymbol{g}'']\!])$.
16: $\boldsymbol{g}''$ の要素のうち 0 以外の異なる値を $v_1, \ldots, v_{m_\cap}$ とする.
17: **for each** $i \in \{1, \ldots, \ell\}, j \in \{1, \ldots, m_\cap\}$ **do**
18: $\quad \boldsymbol{g}''[x] = v_j, \boldsymbol{t}''[x] = \boldsymbol{t}_i[y]$ を満たす x, y を求める.
19: $\quad [\![R_i]\!][j] := [\![T'_i]\!][y]$.
20: **end for**

ランダム置換: 行数が n の ℓ 個の行列 $A_1, \ldots, A_\ell$ $(1 \le \ell)$ の秘匿文を入力とし，誰も知らないランダムな全単射 $\pi_r : \{1, \ldots, n\} \to \{1, \ldots, n\}$ について $B_i[j] = A_i[\pi_r(j)]$ を満たす行列 $B_1, \ldots, B_\ell$ の秘匿文を計算する．$[\![B_1]\!], \ldots, [\![B_\ell]\!] \leftarrow \mathsf{Shuffle}([\![A_1]\!], \ldots, [\![A_\ell]\!])$ と表記する.
安定ソート: 行数が n の列ベクトル $\boldsymbol{k}$ と行数が n の ℓ 個の行列 $A_1, \ldots, A_\ell$ $(1 \le \ell)$ の秘匿文を入力とし，$\boldsymbol{k}[\pi_s(i)] < \boldsymbol{k}[\pi_s(j)] \vee (\boldsymbol{k}[\pi_s(i)] = \boldsymbol{k}[\pi_s(j)] \wedge \pi_s(i) < \pi_s(j))$ $(i < j)$ を満たす全単射 $\pi_s : \{1, \ldots, n\} \to \{1, \ldots, n\}$ について $B_i[j] = A_i[\pi_s(j)]$ を満たす行列 $B_1, \ldots, B_\ell$ の秘匿文を計算する．$[\![B_1]\!], \ldots, [\![B_\ell]\!] \leftarrow \mathsf{StableSort}([\![\boldsymbol{k}]\!]; [\![A_1]\!], \ldots, [\![A_\ell]\!])$ と表記する.

3. 等結合アルゴリズム

説明を簡潔にするため，キー属性は各表の 1 列目とする．ℓ 個の行列として与えられた表 $T_1, \ldots, T_\ell$ から共通のキーを持つ行 (レコード) を抜き出し，同じ行になるように並べ替えた新しい表 $R_1, \ldots, R_\ell$ を，表を秘匿化したまま計算する．但し，T_i は $m_i \times n_i$ 行列，R_i は $m_\cap \times n_i$ 行列 $(1 \le i \le \ell)$ とする．ここで $m_\cap$ は等結合後の表のレコード数である．また，各表の中でキーに重複はないものとする．すなわち，$T_i (1 \le i \le \ell)$ の 1 列目の値はすべて異なるものとする．提案する等結合アルゴリズムを Algorithm 1 に示す.

3.1 正当性

$\boldsymbol{k}$ はすべてのテーブルのキー属性の値をまとめて昇順に並べたベクトルである．従って，$\boldsymbol{k}$ に含まれる同一の値はすべて連続している．各 $\boldsymbol{k}'_i$ 内に重複はないので，同一の値の連続は高々 ℓ 個である．従って $\boldsymbol{e}[j]$ は $\boldsymbol{k}$ で $\boldsymbol{k}[j]$ が同一の値が ℓ 個並んでいる箇所の先頭である場合のみ 1 となり，さらに $\boldsymbol{t}' = \boldsymbol{t}_1||\cdots||\boldsymbol{t}_\ell$ でソートが安定であることから $\boldsymbol{t}[j]$ は $\boldsymbol{t}_1$ の要素である．$\boldsymbol{x}[j]$ は $\boldsymbol{e}[j] = 1$ の場合のみ非 0 となり，非 0 の値に重複はない．非 0 の $\boldsymbol{x}[j]$ に対し，$\boldsymbol{x}[j]$ の上下それぞれ $\ell - 1$ 個値は 0 であるので，$\boldsymbol{g}[j] = \boldsymbol{g}[j+1] = \cdots = \boldsymbol{g}[j+\ell-1]$ となる．すなわち，$\boldsymbol{k}[i] = \boldsymbol{k}[j]$ ならば $\boldsymbol{g}[i] = \boldsymbol{g}[j] \neq 0$ が成り立つので，復元後の $\boldsymbol{g}''$ の非 0 で等しい値の要素に対応した $\boldsymbol{t}''$ の要素に対応するレコードは同一のキーを持つ.

3.2 安全性

アルゴリズム中で開示されるのは，15 行目の $\boldsymbol{t}''$ と $\boldsymbol{g}''$ だけである．$\boldsymbol{t}''$ と $\boldsymbol{g}''$ は以下の手順により入力と出力と公開情報から模倣することにより，Algorithm 1 が入力と出力と公開情報以外の情報を漏らさないことを確認する.

公開情報である $\boldsymbol{t}_1, \ldots, \boldsymbol{t}_\ell$ から 1 個ずつの要素を選んだ ℓ 組を重複のないようにランダムに $m_\cap$ 個作る．$\boldsymbol{t}'$ の各要素に対して，その要素が選ばれていれば同じ組の $\boldsymbol{t}_1$ の要素の値を，そうでなければ 0 を対応させたベクトル $\boldsymbol{g}'$ を作る．$\boldsymbol{t}', \boldsymbol{g}'$ をランダム置換したベクトルは $\boldsymbol{t}'', \boldsymbol{g}''$ と区別することができない．以上により模倣ができた.

3.3 計算コスト

2. 節で例としてあげた秘匿計算の場合を例にして計算コストの評価を行う．多くのデータ処理では表の属性数はレコード数 n に比べて非常に小さいため，表の属性数すなわち行列の列数は定数とみなす．環 R の要素 1 個の送受信を単位とすると，秘匿化，復元，加算，乗算，等号判定の通信量は定数，ランダム置換の通信量は $O(n)$，安定ソートの通信量は $O(n \log n)$ である．したがって，入力の表のレコード数の総和を m とすると，Algorithm 1 の通信量は $O(m \log m)$ である.

参考文献

[Agrawal 03] Agrawal, R., Evfimievski, A. V., and Srikant, R.: Information Sharing Across Private Databases, in *SIGMOD Conference*, pp. 86–97, ACM (2003)

[Cramer 01] Cramer, R. and Damgård, I.: Secure Distributed Linear Algebra in a Constant Number of Rounds, in *CRYPTO*, Vol. 2139 of *LNCS*, pp. 119–136, Springer (2001)

[Freedman 04] Freedman, M. J., Nissim, K., and Pinkas, B.: Efficient Private Matching and Set Intersection, in *EUROCRYPT*, Vol. 3027 of *LNCS*, pp. 1–19, Springer (2004)

[Freedman 05] Freedman, M. J., Ishai, Y., Pinkas, B., and Reingold, O.: Keyword Search and Oblivious Pseudorandom Functions, in *TCC*, Vol. 3378 of *LNCS*, pp. 303–324, Springer (2005)

[Hazay 08] Hazay, C. and Lindell, Y.: Efficient Protocols for Set Intersection and Pattern Matching with Security Against Malicious and Covert Adversaries, in *TCC*, Vol. 4948 of *LNCS*, pp. 155–175, Springer (2008)

[Huang 12] Huang, Y., Evans, D., and Katz, J.: Private Set Intersection: Are Garbled Circuits Better than Custom Protocols?, in *NDSS* (2012)

[Yao 86] Yao, A. C.-C.: How to Generate and Exchange Secrets (Extended Abstract), in *FOCS*, pp. 162–167 (1986)

[五十嵐 11] 五十嵐 大, 千田 浩司, 濱田 浩気, 高橋 克巳 : 軽量検証可能 3 パーティ秘匿関数計算の効率化及びこれを用いたセキュアなデータベース処理, in *SCIS* (2011)

[志村 08] 志村 正法, 遠藤 つかさ, 宮崎 邦彦, 吉浦 裕 : 安全で機能制限のないデータベースを実現するマルチパーティプロトコルを用いた関係代数演算, 情報処理学会研究報告. CSEC, Vol. 2008, No. 71, pp. 187–193 (2008)

[濱田 10] 濱田 浩気, 五十嵐 大, 千田 浩司, 高橋 克巳 : 3 パーティ秘匿関数計算上のランダム置換プロトコル, in *CSS* (2010)

[濱田 11] 濱田 浩気, 五十嵐 大, 千田 浩司, 高橋 克巳 : 秘匿関数計算上の線形時間ソート, in *SCIS* (2011)