一般口演4
セキュリティとプライバシー保護

2019年11月22日(金) 09:00 〜 11:00 H会場 (国際会議場 3階中会議室304)

[2-H-1-06] 汎用的な SQLを用いた匿名加工処理技術の開発

○冨樫 由美子[1]、吉野 雅之[1] （1. 株式会社　日立製作所）

キーワード：Anonymization, k-anonymity, Relational Database

改正個人情報保護法や次世代医療基盤法の施行により、匿名加工された医療情報の利活用が今後一層進展していくと期待される。データベースに格納されたデータから匿名加工データを作成する方法としては、格納された生データをファイルにエクスポートして匿名加工ツールに入力する方法と、データベース内のデータを SQLを用いて匿名加工し、直接加工後データの取得を行う方法がある。後者は、生データのエクスポートが不要となるため、匿名加工にかかる工程を削減できるほか、生データファイルの安全管理対策も不要となる利点がある。報告者らは、匿名加工情報の効率的な取得を目的として、リレーショナルデータベースに対して汎用的な SQLを用いて効率的に匿名加工処理ならびに識別リスク確認処理を行い、有用性の高い加工方法をユーザに提示する技術を開発した。匿名加工データは、特定の個人を識別できる情報を削除、置換する加工処理が求められるが、どの情報をどの程度削除、置換すればよいかは一律に決まるものではない。そのため、データ加工においてはデータの個人識別リスクと有用性を考慮しながら加工方法の調整が必要になる。そこで、データ利用者の傾向として、(1)データ利用者はより元データに近いデータを求めている（質の観点）、(2)データ利用者はより多くのデータを求めている（量の観点）、(3)データの質、量のどちらをどの程度重視するかはデータ利用者によって異なる、という３つの前提を置き、 k-匿名性を満たすかどうか、レコード削除許容レコード数の2つのパラメータを用いて有用性の低い加工＆評価作業を省略することで、加工方法の中から有用性の高いものを高速に提示することを可能にした。ダミーデータによる処理時間の評価を行った結果、網羅的に加工＆評価を行う場合と比較して、最大７０％の削減効果が得られた。

汎用的な SQL を用いた匿名加工処理技術の開発

冨樫　由美子[*1]、吉野　雅之[*1]、
*1 株式会社　日立製作所

Anonymization Processing by Standard SQL

Yumiko Togashi[*1] , Masayuki Yoshino[*1]
*1 Hitachi, Ltd.

The enforcement of the Act on the Protection of Personal Information and the low about anonymized medical data for Research and Development in medical field is expected to further promote the utilization of anonymously processed medical information. For the purpose of efficient acquisition of anonymized information, we have developed a technique to acquire anonymized processing results directly from a relational database using standard SQL, and to efficiently present an anonymized processing proposal that is expected to be useful to users. The proposed method reduces the time to present an anonymous processing proposal to the user by up to 72% compared to the case where the anonymized processing method is comprehensively executed.

Keywords: Anonymization, k-anonymity, SQL

1. はじめに

改正個人情報保護法や次世代医療基盤法の施行、ガイドラインの整備により、医療情報の利活用が今後一層進展していくと期待される。法律、ガイドラインでは、個人情報の二次利用においては、個人を識別できないように加工する匿名加工処理が不可欠である。また、目的内利用であっても、保有する個人情報を適切に管理することが求められている。

収集したデータの格納先としては、データベースが一般的である。リレーショナルデータベース（以下、RDB）に格納された生データを元に匿名加工データを作成する方法としては、格納された生データをRDBからファイルにエクスポートして匿名加工処理が可能なツール[1]に入力する方法と、RDB 内のデータを SQL を用いて匿名加工し、RDB から直接加工後のデータを取得する方法がある。後者は、生データのエクスポートが不要となるため、匿名加工データの取得にかかる工程を削減できる。また、生データを RDB の外に出さないため、生データファイルの安全管理対策が不要となる利点がある。

匿名加工した結果がどの程度安全かを判断する指標の一つに k-匿名性がある。k-匿名性を満たす（個人が特定される確率をk分の1以下に低減する）ように加工することをk-匿名化という。効率的に k-匿名化結果を取得する手法[2]が提案されているが、RDB を対象としたものは見当たらない。

そこで報告者らは、匿名加工情報の効率的な取得を目的として、RDB から SQL を用いて効率的に k-匿名化した匿名加工情報を取得する技術を開発した。

RDB のデータを扱う際には、RDB に標準機能として用意されている SQL 関数（以下、標準 SQL）を使う方法のほかに、RDB 固有の関数を使用する方法、ユーザ定義関数やストアドプロシージャなどを登録して利用する方法がある。本論では、汎用性を重視し、標準 SQL を用いた匿名加工処理を対象とした。

2. 標準 SQL を用いた匿名加工データ取得

匿名加工処理は、データにノイズを入れる匿名加工方法（データスワッピング、ノイズ付加、疑似データ作成など）と、ノイズを入れない匿名加工手法（トップ/ボトムコーディング、一般化、セル削除、レコード削除など）に大別できる。医療情報は利用目的によってノイズを入れる加工が許容できないことも多い。そこで、本論では一般化とレコード削除による匿名加工を対象とする。

また、データの利用目的によって、どの匿名加工が最も有用かは異なる。そこで、データ利用者に複数の匿名加工方法とその匿名加工によって得られるデータ件数を提示し、利用者にとっての最適な加工方法を利用者自身が選択する方法とした。

標準 SQL を用いた匿名加工データ取得処理の流れは以下の通りである。

＜匿名加工データ取得手順＞

1. 取得対象データから k-匿名化対象属性を選定する
   ＜例＞生年月日、性別、住所
2. k-匿名化対象属性ごとに一般化ルールを定義する
   ＜例＞
   生年月日：b0:生データ、b1:月まで、b2:年まで、b3:*(単一の値に変換)（4 通り）
   性別：s0:生データ、s1:*（2 通り）
   住所 : a0:生データ、a1:都道府県まで、a2:*（3 通り）
3. 満たすべき k-匿名性の k 値を決定する
4. 一般化ルールの組み合わせ方法ごとに k-匿名性を満たさないレコード件数を集計する集計用 SQL を作成する
5. 集計用 SQL を実行し、集計結果を表示する
6. 集計結果を確認し、ニーズに近い加工方法の匿名加工結果を取得するデータ取得用 SQL を実行する

2.1 課題

匿名加工対象の属性数や匿名加工対象属性ごとの一般化ルールの増加により、加工方法の組み合わせは爆発的に増えていき、集計処理にかかる時間も増加する。データ利用者側の観点としても、有用性の低い一般化パターンをあらかじめ選択肢から除外することはユーザビリティの観点でメリットがある。そこで、集計用 SQL の実行回数削減による、集計処理の高速化を検討した。

3. 提案手法

2.で示した匿名加工データ取得手順の手順 5.集計 SQL の実行処理対象削減による高速化について述べる。高速化にあたり、複数の加工方法が提示された場合のデータ利用者の選択基準として以下の前提をおく。

＜選択基準＞
a. データ利用者はできるだけ生データに近いデータを利用したい（データの質の観点）
b. データ利用者はできるだけ多くの件数のデータを利用したい（データの量の観点）
c. データの質と量のどちらをどの程度重視するかはデータ利用者によって異なる
d. 複数の属性を取得する場合、どの属性がより重要か、どこまで一般化が許容できるかはデータ利用者によって異なる

3.1 一般化度合いが高い加工方法の集計省略

図 1 は、生年月日を様々な条件で一般化した際に、k-匿名性を満たすレコードと満たさないレコードの割合の変化イメージを表す。一般化を進めるほど、k-匿名性を満たすレコードの割合は増加していく。図 1 の例では、生年月日のデータを 10 年単位に丸めた場合、すべてのレコードが k-匿名性を満たしている。このとき、選択基準 a の「データの利用者はできる限り生データに近い情報を取得したい」の前提においては、利用者が 20 年単位に丸めたデータや、情報を完全に秘匿したデータを選択することはない。つまり、20 年ごと、完全秘匿に一般化する加工の集計処理は省略可能となる。

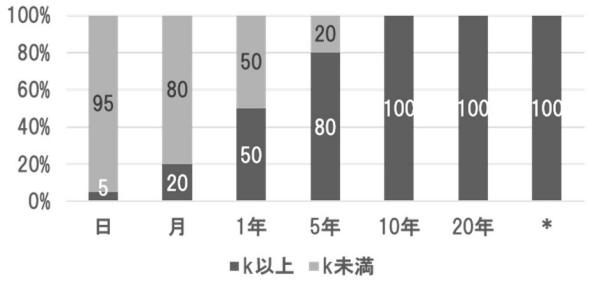

図 1 k-匿名性を満たすレコード割合の推移

3.2 一般化度合いが低い加工方法の集計省略

本節では 3.1 とは逆に、生データに近い一般化を行う加工方法の集計処理の省略の考え方について述べる。図 1 において、生年月日そのままのデータで k-匿名性を満たすレコードは全体の 5%である。これは、データ利用者が k-匿名性を満たす生年月日の匿名加工データを取得しようとすると、k-匿名性を満たした全体の 5%のデータのみを取得するということを意味する。k-匿名性を満たしたごく一部のデータを抽出して分析等を行っても、正しい結果が得られない可能性が高く、データ件数が大幅に削減される結果を利用者が選択する可能性は極めて低いといえる。

そこで、新たに k-匿名性を満たさないレコードの上限数を設定する。たとえば、図 1 の例でレコード削除上限数を 30%を指定した場合、1 年ごと、月ごと、生データの集計処理は省略可能となる。

3.3 双方向での集計省略

本節では、3.1、3.2 の両方の集計省略処理を同時かつ効率的に行うアルゴリズムを示す。図 2 は 2 章で例示した匿名化対象属性と一般化ルールに対してとりうる一般化パターンをラティス構造で表したものである。図 2 ではノードが加工方法を示しており、各ノードから 1 段階一般化が進む（戻る）ノードをつないでいる。上に行くほど生データに近く、最上層は生データ、最下層は完全に秘匿されたデータとなる。3.1 で述べた方法は図 2 の下側の集計処理を省略する方式であり、3.2 で述べた方法は上側を省略する方式である。

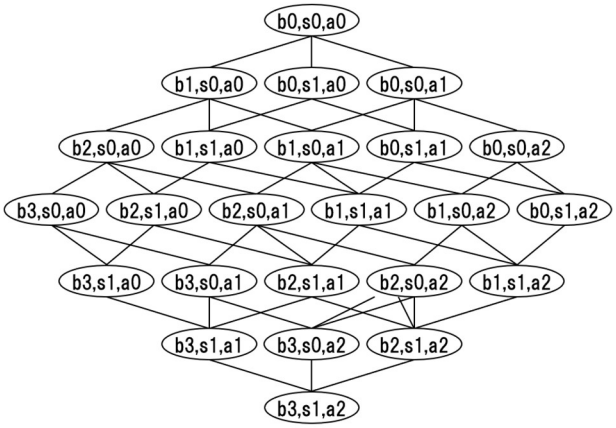

図 2 一般化パターン

3.3.1 事前準備

2 章で示した匿名加工データ取得手順の手順 2 において各属性の一般化ルールに対して生データを 0、完全に秘匿する一般化を 1 とする一般化レベルを合わせて設定する。また、手順 3 において、k-値に加えてレコード削除上限数を設定する。

3.3.2 高速集計処理

双方向で集計処理の省略を行う高速集計処理手順を以下に示す。

＜高速集計処理手順＞
1. すべての加工方法を匿名化対象属性の一般化レベルの和（以下、トータル一般化レベル）が小さい順にソートする
2. ソートした結果、中間に位置する加工方法の集計用 SQL を実行する
3. k-匿名性を満たさないレコード件数が下記条件を満たす場合、省略可能な加工方法を実行対象から削除する
   a. k-匿名性を満たさないレコードが 0 件の場合、2. で実施した加工方法から図 2 で下方向につながる加工方法をすべて実行対象から削除する
   b. k-匿名性を満たさないレコードがレコード削除上限数を超えた場合、実施した加工方法から図 2 で上方向につながる加工方法をすべて実行対象から削除する
4. 実行対象の加工方法がなくなるまで 2～3 を繰り返す
5. 集計結果を表示する

本アルゴリズムでは、トータル一般化レベルが小さい順にソートし、トータル一般化レベルが中間となる一般化パターンに対して集計 SQL を実行することにより、省略可能な可能性の高い一般化レベルが小さいまたは大きい加工方法の集計 SQL の実行を抑制する。

4. 評価実験

RDB に 100 万件のデータを格納したデータテーブルを用意して、集計処理の省略の有無を変えた下記 4 パターンのプログラムを実装し、集計処理の実行時間を比較した。k 値については 2 を設定した。

1. 省略なし
   すべての加工方法に対して集計処理を実施
2. 一般化レベルの高い加工方法の実行省略
   トータル一般化レベルの小さい順に実行し、k-匿名性を満たさないレコードが 0 件となった場合に、そ

の加工方法からさらに一般化した加工方法の集計を省略（3.1 節記載の方法）

3. 一般化レベルの低い加工方法の実行省略
   トータル一般化レベルの大きい順に実行し、k-匿名性を満たさないレコードがレコード上限数を超えた場合に、その加工方法からさらに生データに近くなる加工方法での集計を省略（3.2 節記載の方法）
4. 双方向での実行省略
   トータル一般化レベルが中間に位置する加工方法から順次実行し、条件を満たす場合に一般化レベルの高い加工方法、または低い方法の実行を省略する（3.3 節記載の方法）

4.1.1 加工方法の増加による影響評価

k-匿名化対象属性と属性ごとの一般化ルール数を変えて、集計処理の実行時間を計測し、加工方法の増加による影響を評価した。レコード削除の上限数は 1 万件（レコード全体の 1%）とした。評価に使用した属性、一般化ルールと対応する一般化レベルは以下の通りである。

a. 3 属性、3 階層（全 27 パターン）
   i．住所:3 階層（0:生データ、0.5:都道府県、1:＊）
   ii．生年月日:3 階層（0:生データ、0.5:生年、1:＊）
   iii．医療機関コード:3 階層(0:生データ、0.5:上 2 ケタ、1:＊)
b. 3 属性、3～5 階層（全 45 パターン）
   i．住所:3 階層
   ii．生年月日:5 階層（0:生データ、0.25:生年月、0.5:生年、0.75:10 年ごと、1:＊）
   iii．医療機関コード:3 階層
c. 4 属性、2～3 階層（全 54 パターン）
   i．住所:3 階層
   ii．生年月日:3 階層
   iii．医療機関コード:3 階層
   iv．性別:2 階層
d. 4 属性、2～5 階層（全 90 パターン）
   i．住所:3 階層
   ii．生年月日:5 階層
   iii．医療機関コード:3 階層
   iv．性別:2 階層

4.1.2 レコード削除上限数による影響評価

レコード削除上限数を設定する集計実行方法（3.一般化レベルの低い加工方法の実行省略、4.双方向での実行省略）に対して、レコード削除上限数を変えて集計処理の実行時間を計測し、レコード削除上限数による集計処理時間への影響を評価した。具体的には、100 万件のデータに対してレコード削除上限数として、(a)0 件(0%)、(b)1,000 件(0.1%)、(c)1 万件(1%)、(d)10 万件(10%)、(e)50 万件(50%)を設定して測定した。測定には 4.1.2 の d の対象属性と一般化パターンを用いた。

4.2 実験用データ

実験には統計情報等を利用し実データに近くなるように作成したデータを使用した。データベースのテーブル内には、ICD10 コード、傷病名コード、性別、年齢、生年月日、医療機関コード、郵便番号、住所の 8 属性が含まれており、このうち医療機関、住所、性別、生年月日を一般化対象とした。一般化対象属性で定義した一般化レベルと、一般化レベルごとの属性値のパターン数を表1に示す。

表 1 評価用データ

属性	属性値パターン数				
	0	0.25	0,5	0,75	1
医療機関コード	88,929	-	97	-	1
住所	116,450	-	47	-	1
性別	2	-	-	-	1
生年月日	36,495	1200	100	11	1

4.3 評価環境

実験に使用した評価環境を以下に示す。

- CPU：Intel Core i7-4790 CPU @ 3.60GHz
- メモリ：16.0GB
- OS：Windows 7 Professional SP1
- DB：PostgreSQL 10

4.4 実験結果

4.4.1 加工方法の増加による影響評価結果

加工方法の増加による影響評価結果を図 1 に示す。提案した高速集計処理により、4.1.1 に示した a-d のすべての場合において集計処理時間は減少した。特に、レコード削除上限数を指定する方式の削減効果が高く、すべての集計用 SQL を実行する場合と比較して最大 72%処理時間が削減した。匿名化対象属性数や属性ごとの一般化ルールの増加により一般化パターンの組み合わせが増えれば増えるほど、集計時間の削減率はアップした。

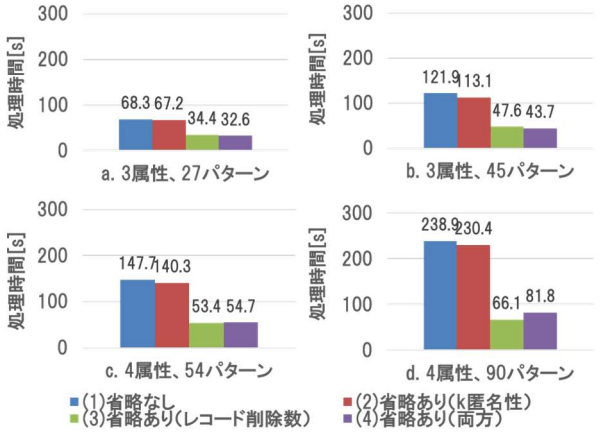

図 3 一般化パターンの増加による影響評価結果

4.4.2 レコード削除上限数による影響評価

レコード削除上限数による影響評価結果を図 2 に示す。レコード上限数を小さく設定するほど、集計処理時間は減少した。また、レコード削除上限数が少ない場合は、(3)一般化レベルの小さい方向の枝刈りの方が集計時間は短く、レコード削除上限数がある程度大きくなると(4)双方向枝刈りの方が集計時間が短くなった。

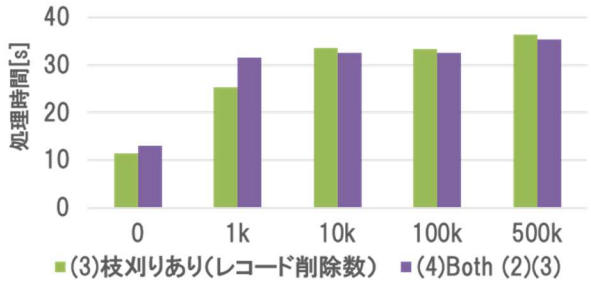

図 4 レコード削除上限数による効果

4.4.3 考察

集計処理の実行省略の有無による処理性能差について考察する。図 5 は 4.1.1 の d.の一般化対象、一般化レベル（4 属性、全 90 パターン）における各一般化パターンで集計処理を行った場合の属性値の組み合わせパターン数と集計処理時間との関係を表す。生データ（一般化を行わない場合）の集計時間は約 3.2 秒であり、最長は 4.3 秒（医療機関コード:1、生年月日:1、住所:0、性別:0）、最短 0.2 秒（全属性を「*」に一般化）であった。図 5 に示すように、一般化後の属性値の組み合わせの出現パターンが少ない一般化パターンほど、集計処理時間は短くなる傾向があった。集計処理は一般化を行う処理と一般化後の結果を集計する処理に分けられるが、データに対して一般化を行う時間よりも一般化した結果を集計する時間の影響が大きいといえる。

(2) 一般化レベルが高い方向の枝刈り方法は、実行時間の短い処理（データの組み合わせの出現パターンが少ない一般化パターンの集計）を削減するため効果が低く、逆は効果が高い結果となったといえる。

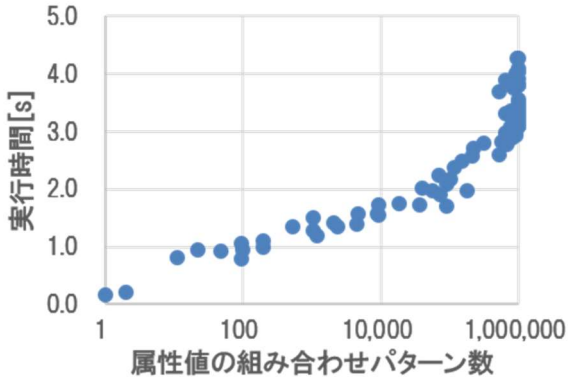

図 5 パターン数と実行時間の関係

5. まとめ

本稿では、RDB に格納されたデータに対する標準 SQL を用いた k-匿名化の集計処理の高速化について述べた。本稿で述べた高速化アルゴリズムを適用して、4 属性（属性の一般化パターン 3 通り、5 通り、3 通り、2 通りの計 90 パターン）100 万件のダミーデータを 2-匿名化した結果、一般化処理の処理時間は全パターンを処理する場合と比較して 50～72%削減できた。高速化方式は匿名化対象属性、また、匿名化対象属性の一般化パターンが増えるほど効果が高いことを確認した。特に、レコード削除上限数を指定することによる枝刈り効果が高いことを確認した。

本論では、加工方法として、一般化とレコード削除を対象としたが、トップボトムコーディングやセル削除など、ほかの加工方法も標準 SQL で実行が可能な匿名加工方法である。他の匿名加工方法を組み合わせた場合の効率的な集計作業の検討は今後の課題である。

参考文献

1) ARX,
[https://arx.deidentifier.org/]
2) Kristen LeFevre, David J. DeWitt, Raghu Ramakrishnan. Incognito: Efficient Full-Domain K-Anonymity. Proceedings of the 2005 ACM SIGMOD international conference on Management of data, 49-60