Vol.2011-ARC-196 No.9
2011/7/27

低遅延匿名化処理機構における情報損失度改善手法の提案

澤田純一[†1] 井上恒一[†1] 西宏章[†1]

近年の様々な web サービスの普及に伴いインターネット上に蓄積されつつある大量のデータはユーザの動向や趣味趣向といった情報を含み，サービスの提供やマーケティング，社会工学的研究において利用価値が高い．そのため，現在は個人情報保護の観点から非公開あるいは破棄されているこれらの情報を公開，共有し有効に活用する需要が高まっている．このような背景の中，問題となるプライバシ保護と情報公開との両立を可能とする技術が提案されており，その一つが匿名化手法である．一般に処理コストが大きいことで知られるが，先行研究にて専用のハードウェアアーキテクチャを提案・実装し，処理の高速化を実現した．しかし同時に情報の損失が大きくなることがわかった．本稿では情報の損失を改善するため，処理結果を保存し，以降の処理へ反映するキャッシュ機構を提案する．予備評価の結果，広域網である WIDE のトレースにおいて，ベストケースに向けて最大約 60%情報損失度を低減することができた．

Improving Information Loss on Privacy Preserving Architecture

Junichi Sawada,[†1] Koichi Inoue[†1] and Hiroaki Nishi[†1]

In experiencing the growth of quantity and value of data, data holders realize the importance to utilize information that is abandoned or concealed currently. In this situation, they face the difficulty of releasing data without revealing privacy information. One of the methods to protect privacy information in publishing data is known as a privacy preserving method based on a constraint termed k-anonymity and l-diversity. It enables users of the data to utilize published data as they like. However, it requires a large calculation cost. In precedence research, we developed TCAM-based architecture that significantly accelerates the method. This paper proposes an architecture using cache mechanism to improve information loss. The evaluation proves that the information loss can be reduced approximately 60% on WIDE trace.

1. はじめに

近年発展が著しい SNS やブログ，インターネットショッピングなどの web サービスの普及に伴い，利用者の生活情報がサーバやデータベースに蓄積されつつある．今後予想されるユビキタスデバイスの普及に伴い，この流れは加速していくと考えられる．これらの情報には消費者やサービス利用者の動向や趣味趣向といった情報が多分に含まれ，高度なマーケティングや豊かなサービスの提供，社会工学的な研究において価値が高いと考えられる．このような背景のもと，従来ほとんど活用されなかったこれらの情報を共有あるいは公開し，有効に活用する動きが出始めている．Yahoo! Japan は 2007 年より知識検索サービスのデータを研究コミュニティに対して提供しており[1)]，また楽天技術研究所も同様に研究促進を目的として商品，施設等のレビューデータを大学や研究機関に対して公開している[2)]．このように情報を公開し活用する試みが出始める一方，問題となるのが個人情報の保護である．従来のような外部からの不正利用を防ぐセキュリティ技術ではなく，プライバシを保護しつつデータを公開する技術が必要となる．そのような技術は PPDP(Privacy Preserving Data Publishing) と呼ばれ，代表的な手法の一つとして k-匿名性，l-多様性といった匿名化があり有望視されている．しかしながら匿名化処理は一般に処理コストが大きく，常時生成され続けるデータに対する匿名化や，データベースから読み出されるデータストリームに対する匿名化など処理スループットが求められる用途への適用が難しい．我々は先行研究において TCAM を利用した専用のハードウェアを提案し，処理の高速化を実現したが，同時にウィンドウサイズが限られるために不要な匿名化が行われ，トランザクション全体を対象に匿名化した場合に比べて情報の損失が大きくなることもわかった．本稿では情報の損失を改善するため，処理結果を保存し以降の処理へ反映するキャッシュ機構を提案する．過去の処理結果を現在のウィンドウにおける処理に反映することで，ウィンドウサイズ内にとらわれない匿名化を実現し，情報の損失を改善する．

†1 慶応義塾大学大学院理工学研究科
Graduate School of Science and Technology, Keio University

Vol.2011-ARC-196 No.9
2011/7/27

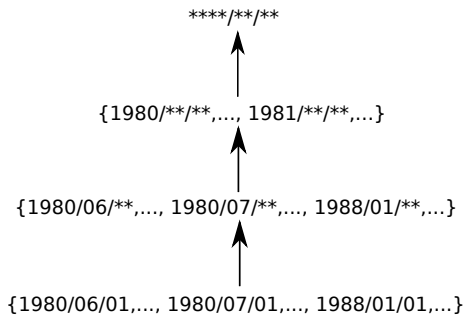

図 1 生年月日の属性一般化階層
Fig. 1 Birth date domain generalization hierarchy

表 1 匿名化処理前のデータテーブル
Table 1 Data table

Race	Birth	ZIP	Problem
Black	1963	02150	short breath
Black	1960	02140	chest pain
Black	1964	02138	chest pain
Black	1964	02138	obesity
Black	1964	02138	short breath

表 2 2-匿名性が達成されたテーブル例 (注目属性:Problem)
Table 2 2-anonymized table with sensitive attribute "Problem"

Race	Birth	ZIP	Problem
Black	196*	021**	short breath
Black	196*	021**	chest pain
Black	1964	02138	chest pain
Black	1964	02138	obesity
Black	1964	02138	short breath

2. 関 連 研 究

2.1 匿名化の定義

プライバシ情報を保護しつつデータから有用な知識を得る技術は PPDM(Privacy Preserving Data Mining) と呼ばれ，いくつかの手法が提案されている[3)4)]．PPDM は複数のデータベースにおいてそれぞれの機密を保ったまま，つまり機密情報を自身以外に公開せずに，相関関係や統計情報といった有用な情報を得る技術である．その中でもプライバシが保護された状態でのデータの公開を目的とする技術は特に PPDP(Privacy Preserving Data Publishing) と呼ばれ[5)]，情報の抽出ではなくデータの公開までを扱う点で PPDM よりも扱う範囲が限定的である．PPDP 技術の一つとして分類される匿名化手法ではデータを一般化することで情報を抽象化し，プライバシ基準である k-匿名性，l-多様性を達成したのちデータを公開する．

本稿で使用する用語を以下のように定義する．なお定義は 6)7) を参考にした．

属性一般化階層 DGH

匿名化はデータの一部を'*' で置き換える，または別の値で置き換えることでデータを抽象化する一般化によって行われる．このときの属性値の変化は値の階層で定義できる．例えば**図 1** は生年月日の一般化による値の変化を表している．これをその属性の属性一般化階層 DGH(Domain Generalization Hierarchy) と呼ぶ．

属　　性

データテーブルの列を属性と呼ぶ．属性のうち，直接プライバシ情報を特定できると考えられるもの，例えば名前のような属性を「直接識別子」と呼ぶ．また直接識別子ではなくても複数の属性を組み合わせることで個人を特定できる可能性がある．そのような属性を「準識別子 (Quasi-Identifier:QI)」と呼ぶ．

注 目 属 性

属性の中で，データ解析者にとって重要な項目であるため一般化を行わない属性を総称して「注目属性」と呼び，それ以外の属性を「非注目属性」と呼ぶ．また，非注目属性の値が一般化され，データの組み合わせが複数タプルで同一であったとき，その複数タプルのグループを $q*$-ブロックと呼ぶ．

2.2 k-anonymity

k-匿名性は「データテーブル中の各タプルにおいて，そのタプルの持つ準識別子の組み合わせと同一の準識別子の組み合わせを持つタプルが自身を含め k 個以上存在する状態」と定義される．

表 1 に匿名化処理前のデータテーブルを，**表 2** に一般化により 2-匿名性が達成されたデータテーブルの例を示す．k-匿名性が達成されたデータテーブルでは非注目属性が同一であるタプルが k 個以上存在する．したがって k-匿名性が達成されているデータテーブルでは，あるタプルは少なくとも他の $k-1$ 個のタプルと区別できない．

ただし k-匿名性ではプライバシの保護が十分でないケースがあり，例えば $q*$-ブロック中

の注目属性がすべて同一の場合，$q*$-ブロックの中から 1 つのタブルに特定できなくとも注目属性が特定できる．また非注目属性と注目属性の関係について何らかの背景知識を持っている場合，注目属性の値から非注目属性を推測し，注目属性の推測範囲を狭めることができる可能性がある．これらはいずれも注目属性の多様性の欠如に起因するため，より堅固なプライバシ基準として次に挙げる l-多様性が提案されている．

2.3 l-diversity

l-多様性は直感的には次のように定義される．

「データテーブル中のすべての $q*$-ブロックにおいて，出現する注目属性の値が少なくとも l 個の多様性を有する」

厳密には次のように定義される．データテーブル中の各 $q*$-ブロックにおいて出現する注目属性のうち，i 番目に多く出現する値の出現回数を r_i とする．ある $q*$-ブロック中に n 種類の注目属性が出現するとき，次式 1 が成り立てばこの $q*$-ブロックは l 多様性を満たしている．

$$r_1 \leq r_l + r_{l+1} + ... + r_n \quad (1)$$

2.4 匿名化手法

匿名化処理の高速化に関する研究として，クラスタリングを用いることでソフトウェア実行性能を向上させる提案がなされている[8]．しかしクラスタリングの効果は特定のケースに限定され，多くのケースではクラスタリングを用いない手法が最も計算時間が小さくなる．匿名化処理を行う際，あるタプルがプライバシ基準を満たしているか判定するためには他のタプルとの比較が必要である．このときの計算量はワーストケースで $O(n^2)$ となり，処理タプル数の増加に伴い計算時間は指数関数的に増加する．そこで先行研究 9) では TCAM を用いた専用のハードウェアを提案し，処理を並列化することで高速化を実現した．具体的には web 閲覧履歴 1000 タプルを処理コストの大きい l-多様性で処理した場合，検索処理がシリアライズされる RAM ベースのアーキテクチャと比較して約 10-50 倍のスループットを得た．参考までに同じアルゴリズムをソフトウェアで実行し，同じ情報損失度で処理を行った場合と比較すると，最大約 2000 倍のスループットを得た．しかし同時にウィンドウサイズが制限されるため，トランザクション全体を対象に匿名化した場合に比べ余分に一般化が行われてしまい，情報の損失が大きい．

処理スループットが求められる実行環境として，データベースから読み出される出力や常時生成され続けるデータ等のデータストリームに対する匿名化処理が挙げられる．研究 10) ではデータストリームに対する k-匿名性に基づく匿名化手法の提案がなされている．基本的な手法としては R 木を用いており，到着したタプルを k 個以上のグループにまとめ，それぞれをひとつの葉ノードとして扱う．葉ノード作成時あるいは分割時に設定された待ち時間が経過したのち，葉ノード内の全タプルが等しくなるまで一般化を行い，葉ノード内の全タプルを出力する．新たに到着したタプルによって葉ノードが分割される場合，分割後の葉ノードはより類似するタプルで構成されるようになり必要な一般化回数が少なくなるため，好ましい．そこでウィンドウサイズ内でデータの分散を計算し，新たに到着するタプルによって分割される可能性が低い葉ノードを優先的に出力する．出現頻度を反映することで入力から出力までに要する遅延を抑えつつ情報損失の低減を図っているが，この手法ではタプルの到着順を保存することは考慮していない．順番が重要でないデータであれば問題ないが，データストリームに対する処理では用途によってはタプルの順番が重要である．本稿で提案するアーキテクチャではタプルの到着順と出力順が同一であることを重視し，FIFO として実装する．またこのように動的かつ無限に生成されるデータを匿名化する場合，静的なデータの場合と異なりソフト・ハード関わらずウィンドウサイズを定めて処理を行わなければならない．したがって動的なデータを対象とする場合，本稿で提案するキャッシュ機構の意味はソフトと比較したハードのデメリットを補うものではなく，ソフト・ハード関わらず性能を改善する機能としての意味を持つ．

3. 匿名化処理機構

3.1 TCAM

匿名化処理をハードウェアで行うにあたり，必要となるのは以下の事項である．

- プライバシ基準を満たしていないタプルの高速な検索機能
- 一般化の概念の実装

提案アーキテクチャではこれらの要求事項を TCAM (Ternary Content Addressable Memory) によって達成する．CAM はデータを入力として受付，格納しているデータと比較し，一致するデータの位置を出力するメモリである．すべてのセルに比較器を実装し，並列処理による高速な検索機能を実現する．注目するタプルについて TCAM で検索することにより，プライバシ基準が満たされているかどうかの判定を $O(n)$ で行うことができる．また TCAM はマスク値を実装し，マスク値を don't care として扱うことで可変長の検索を可能とする．一般化として'*' で置き換える処理は**表 3** のようにマスク値を用いて実装する．

これにより一般化されていない部分のみの比較が可能となる．また**表 4** のようにマスク値をシフトさせることにより，データ末尾からの段階的な一般化を行う．

表 3 TCAM による一般化の実装
Table 3 Generalized values expressed in TCAM

値の意味	TCAM 内のデータ	
	データ	マスク
0011****	00110000	00001111
00111***	00111010	00000111

表 4 TCAM における一般化処理
Table 4 Gneralization process in TCAM

値の意味		TCAM 内のデータ	
非注目属性 1	非注目属性 2	データ	マスク
0000	0000	0000_0000	0000_0000
	↓一般化		
000*	000*	0000_0000	0001_0001
	↓一般化		
00**	00**	0000_0000	0011_0011

匿名化における一般化の概念はこのように TCAM の可変長の検索機能により実装する．

3.2 キャッシュ機構

匿名化処理が完了し，TCAM から出力されたデータはキャッシュとして保存され，以降の処理において参照される．データとマスク値の OR 演算出力を処理結果として，マスク数とセットで保存する．アクセスキーとして処理結果のハッシュ値を用い，ハッシュ関数にはハードウェア実装が比較的容易であることから CRC を応用する．TCAM 内で注目するタプルについて検索する際，同時にキャッシュを参照し，キャッシュされた値とのマッチングも行う．k-匿名性に基づく匿名化処理の場合，キャッシュされている処理結果はすべてプライバシ基準を満たした値であるため，キャッシュにヒットした場合，注目するタプルは即時プライバシ基準を満たしていると判断できる．そのためキャッシュされたマスクをそのまま TCAM 内の注目するタプルに適用し，そのタプルについては匿名化処理が完了したものとする．キャッシュの更新は TCAM 内の全データで匿名化処理が完了し，処理結果を出力する際に行われる．情報の損失が小さい結果を以降の処理に反映することで情報損失度が改善できると考えられるため，キャッシュ更新時にエントリが既存であった場合，マスク数が小さい値を優先して保存する．

3.3 アーキテクチャ

提案するアーキテクチャの概要図を**図 2** に示す． TCAM の各セルからマッチ線を出力

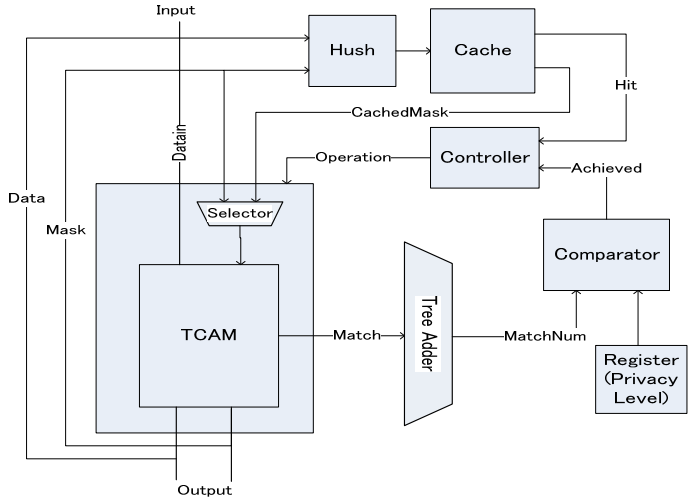

図 2 キャッシュ付き匿名化処理アーキテクチャ
Fig. 2 Architecture for the anonymization mechanism with cache module

し，ツリー加算器はマッチ数を数え上げ，総数を出力する．比較器はあらかじめ設定した k もしくは l とマッチ数を比較し，プライバシ基準が満たされているか否かを出力する．コントローラは書き込みアドレスや読み込みアドレス，検索タプルのアドレスを管理し，プライバシ基準が満たされていなければタプルを一般化し，注目するタプルを更新する．また同時に注目するタプルについてキャッシュを参照し，ヒットすれば TCAM 内の検索結果に関わらず，キャッシュされたマスク値で注目するタプルのマスク値を更新する．最終的に TCAM 内のすべてのタプルでプライバシ基準が達成されたらデータとマスク値を出力する．その際すべての出力タプルについてハッシュ値を求めてキャッシュを参照し，キーがない場合，もしくはキーが既に存在していても出力タプルのマスク値がキャッシュされたマスク値よりも小さければキャッシュする．以上のアーキテクチャにおいて，以下のような順序で匿名化を行う．

(1) データを TCAM に格納する
(2) 注目するタプルについて TCAM，キャッシュを検索する
(3) キャッシュにヒットすれば，キャッシュされたマスク値を TCAM の注目するタプル

Algorithm 1 Anonymization Algorithm

```
loop
  /*Input*/
  while TCAM is not full do
    write_tuple_to_TCAM;
  end while
  /*Anonymization*/
  repeat
    flag ← 0;
    for each tuple in TCAM do
      search_tuple;
      if tuple hits cache then
        set_cache_mask_into_TCAM
      else if l or k is not achieved then
        generalize_tuple;
        flag ← 1;
      end if
    end for
  until flag is 0
  /*Output*/
  while TCAM is not empty do
    read_tuple_from_TCAM;
    if (tuple does not hit cache ∥ (mask <
    cached mask)) then
      cache_tuple;
    end if
  end while
end loop
```

に上書きする

(4) キャッシュにヒットしなかった場合，プライバシ基準が満たされていなければ注目するタプルを一段階一般化し，注目するタプルを次へ

(5) 2〜4 を繰り返し，TCAM 内のすべてのタプルでプライバシ基準が満たされたらデータとマスク値を出力して 1 へ

詳細なアルゴリズムの疑似ソースを Algorithm 1 に示す．

表 5 キャッシュなしの場合とウィンドウの制限がない場合における IL (WIDE, k=2)
Table 5 IL of the method without cache and IL of the method without the constraint of window (WIDE, k=2)

キャッシュなし（ウィンドウサイズ 256）	ベストケース（ウィンドウなし）
22.7%	4.07%

4. 評　　価

ソフトウェア (C++) でキャッシュ機構のシミュレータを作成し，情報損失度について予備評価を行った．一般化により失われた情報の大きさを評価するため，評価指標として情報損失度 $InformationLoss : IL$ を導入する．テーブル T が与えられ，タプルを $t \in T = \{t1, ..., tN\}$ ，準識別子が $QI_T = \{A1, ..., AN_A\}]$ であるとき，関数 IL は以下のように定義される．

$$IL(T) = \frac{\sum_{t_j \in T} \sum_{A_i \in QI_T} \frac{h}{|DGH_{Ai}|}}{|T| \cdot |QI_T|} = \frac{\sum_{j=1}^{N} \sum_{i=1}^{N_A} \frac{h}{|DGH_{Ai}|}}{N \cdot N_A} \quad (2)$$

式中の h はタプル中のある属性値の属性一般化階層における高さを表し $\frac{h}{|DGH|}$ はそれが属性一般化階層全体の高さに対してどこに位置するか，つまりどれだけ匿名化されているかを表す．

アプリケーションとして web における動向調査を想定し，WIDE のトレース（Tran-P: アメリカとの直結ネット）を web 閲覧履歴として宛先 IP アドレスに対して k=2，k=5 で匿名化を行った．100,000 タプル処理し，キャッシュが安定してからの性能を評価するため，最初の 10,000 タプルを除いて IL およびキャッシュヒット率について評価した．同様に当研究室において取得したトラフィックに対しても k=30 で匿名化を行い，評価を行った．なお先行研究において FPGA に実装可能な TCAM の最大サイズが 256x256 であったため，ウィンドウサイズは 256 とした．

図 3 は k=2，**図 4** は k=5 で WIDE のトレースを匿名化したときに，各キャッシュエントリサイズにおいて計測された IL とキャッシュヒット率である． また，**表 5** はキャッシュ機構なし，ウィンドウサイズ 256 の TCAM で k=2 で匿名化を行った場合の IL と，ウィンドウサイズによる制限がないフルサイズで匿名化を行った場合のベストケースの IL である． k=2 ではキャッシュエントリサイズの増加に従い IL は約 10%に収束しており，同時にキャッシュヒット率は 100%弱に収束している．キャッシュなしの場合における IL が 22.7%，ベストケースの IL が 4.07%であることから，ベストケースに向けて約 60%IL を

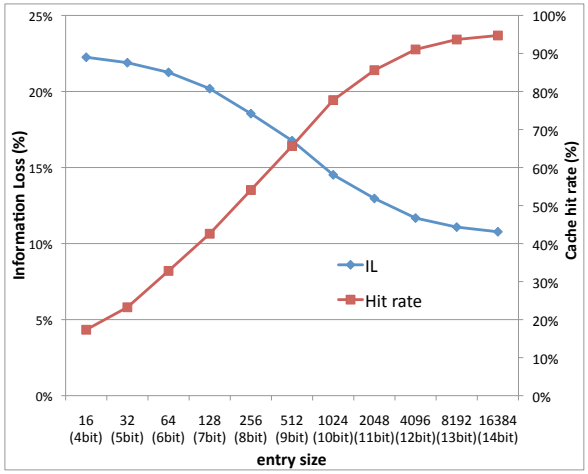

図 3 各エントリサイズにおける IL とキャッシュヒット率 (WIDE, k=2)
Fig. 3 IL and cache hit rate with cache entry size (WIDE, k=2)

低減することができた．同様に k=5 では IL は約 30%弱に収束している．どちらの場合においても，IL 低減幅が下がり始めるのはエントリサイズが 2K-4K 程度のときである．したがって，広域網である WIDE における匿名化処理では，実装規模とのトレードオフを考慮すると 2K-4K 程度のエントリサイズが適当であると考えられる．

図 5 は研究室のトレースを匿名化したときの IL とキャッシュヒット率である． 研究室のトラフィックではエントリサイズ 64 で IL が 4%弱，キャッシュヒット率が約 100%に収束している．研究室のトラフィックは小規模なネットワークであるため，出現する値のバリエーションが少なく，64 と小さいサイズで収束している．以上より広域網である WIDE では 2K-4K 程度のエントリサイズが適当であるといえる．

また，キャッシュによる IL の改善と TCAM のサイズを大きくした場合の IL の改善を比較するため，TCAM の各サイズにおける IL を求め，**図 6** に示す．これは WIDE のトレースを匿名化したときの値である． 図 3，図 4 と比較すると，k=2 において，キャッシュサイズ 4K に対し，TCAM のサイズが 2K でほぼ同等の IL となる．k=5 において，キャッシュサイズ 4K に対し，TCAM のサイズが 1.5K で IL がほぼ同等である．FPGA(Xilinx Virtex5: XC5VLX330T) に実装可能な TCAM の最大サイズは 256 だが，RAM を実装する場合はプリミティブとして用意されているメモリリソースである Block RAM を活用することができるため，大規模な RAM を実装することが可能である．したがって FPGA において提案アーキテクチャを実装する場合，今回の例ではエントリサイズ 4K のキャッシュを実装することにより，1.5K-2K 程度のサイズの TCAM を実装した場合と同等の IL で匿名化処理を行うことができることがわかった．また ASIC としての実装を考えた場合，RAM と TCAM のハードウェアコストを比較・検討する必要がある．

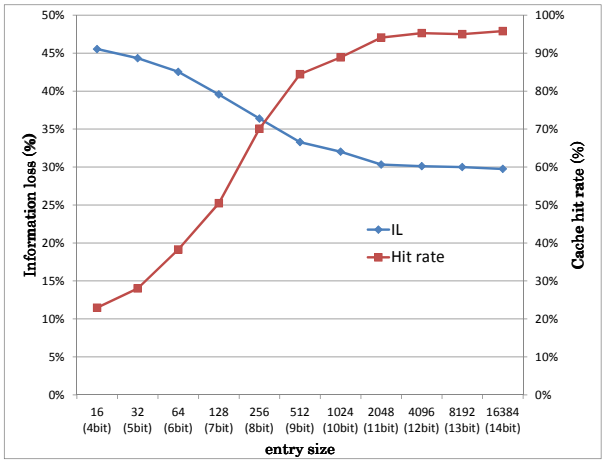

図 4 各エントリサイズにおける IL とキャッシュヒット率 (WIDE, k=5)
Fig. 4 IL and cache hit rate with cache entry size (WIDE, k=5)

5. 結　　論

匿名化処理ハードウェアにおける，ウィンドウサイズによる情報損失度の増加を低減するための機構としてキャッシュ機構を提案し，予備評価を行った．広域網である WIDE のトレースに対し匿名化を行った結果，k=2 ではキャッシュなしの結果に比べてベストケースに向けて約 60%IL を低減することができた．また実装規模とのトレードオフを考慮すると，キャッシュエントリサイズは 2K-4K 程度が適当であると考えられる．FPGA への実装を考えた場合，例として Xilinx Virtex5 (XC5VLX330T) に実装可能な TCAM の最大サイズは 256 だが，今回の例ではエントリサイズ 4K のキャッシュを搭載することにより，サイズが 1.5K-2K 程度の TCAM を実装した場合と同等の IL で匿名化処理を行うことができる．

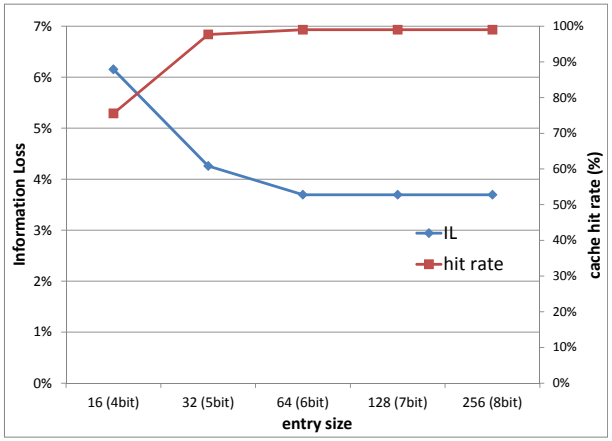

図 5 各エントリサイズにおける IL とキャッシュヒット率 (研究室，k=30)
Fig. 5 IL and cache hit rate with cache entry size (研究室，k=30)

一方研究室程度の小規模なネットワークではキャッシュエントリサイズは 64 程度で十分である．

静的なデータに対しては，ソフトウェアであればウィンドウサイズによる制限がないためフルサイズで処理を行うことができる．匿名化処理ハードウェアにキャッシュ機構を搭載することにより IL を低減し，ソフトウェアの処理結果に近づけることができる．一方動的に，無制限に生成されるデータに対しては，ソフト・ハード関わらずあるウィンドウサイズで処理を行わなければならない．したがってキャッシュ機能により，過去の処理結果を参照することで IL を改善することが可能となる．

今回はまずキャッシュの有効性について評価するため，基本的なプライバシ基準である k-匿名性について評価を行った．TCAM を利用したアーキテクチャは k，l 両方のアルゴリズムに対応しているため，柔軟なプライバシ基準を設定可能とするためにキャッシュ機構も両者に対応することが望ましい．l-多様性に基づく匿名化では注目属性の出現確率が基準値以下でなければならないため，キャッシュのメカニズムにさらなる工夫が必要であり，今後検討を進める予定である．

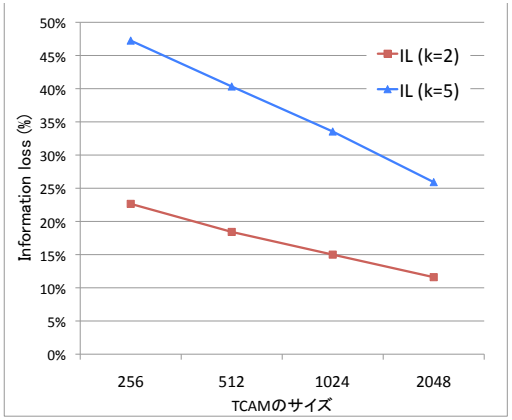

図 6 TCAM のウィンドウサイズによる IL の変化
Fig. 6 IL behavior with the size of TCAM

参 考 文 献

1) 国立情報学研究所：国立情報学研究所発表資料，国立情報学研究所（オンライン），% urljhttp://www.nii.ac.jp/userimg/yahoo_chiebukuro090603.pdf（参照 2011-06-01）．
2) 楽天技術研究所：楽天データ公開，楽天技術研究所（オンライン），入手先⟨http://rit.rakuten.co.jp/rdr/⟩（参照 2011-06-01）．
3) Aggarwal, C.C. and Yu, P.S.: *Privacy-Preserving Data Mining: Models and Algorithms*, Advances in Database Systems, Vol.34, Springer (2008).
4) 佐久間淳，小林重信：プライバシ保護データマイニング，人工知能学会誌， Vol.24, No.2, pp.283–294 (2009-03-01).
5) Chen, B.-C., Kifer, D., LeFevre, K. and Machanavajjhala, A.: Privacy-Preserving Data Publishing, *Found. Trends databases*, Vol.2, pp.1–167 (2009).
6) Sweeney, L.: Achieving K-Anonymity Privacy Protection Using Generalization and Suppression, *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, Vol.10, No.5, pp.571–588 (2002).
7) Machanavajjhala, A., Kifer, D., Gehrke, J. and Venkitasubramaniam, M.: L-diversity: Privacy beyond k-anonymity, *ACM Trans. Knowl. Discov. Data*, Vol.1 (2007).
8) 井上恒一，石田慎一，西　宏章：サービス指向型ルータに利用可能なプライバシー保護アーキテクチャ，情報処理学会研究報告. 計算機アーキテクチャ研究会報告，Vol.2010,

Vol.2011-ARC-196 No.9
2011/7/27

No.11, pp.1–9 (2010-07-27).
9) 澤田純一，井上恒一，西　宏章：低遅延匿名化処理機構の提案と実装，電子情報通信学会技術研究報告，Vol.110, No.473, pp.309–314 (2011-03-18).
10) Zhou, B., Han, Y., Pei, J., Jiang, B., Tao, Y. and Jia, Y.: Continuous privacy preserving publishing of data streams, *Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology*, EDBT '09, New York, NY, USA, ACM, pp.648–659 (2009).