Vol.2019-DPS-178 No.36
Vol.2019-CSEC-84 No.36
2019/3/5

非負精緻化を伴うPrivelet法演算効率化の試み

本郷 節之[1]　大加瀬 稔[1]　寺田 雅之[2]　鈴木 昭弘[1]　稲垣 潤[1]

概要：Privelet 法は，差分プライバシ基準に準拠しつつ，部分和精度にも優れており，プライバシが保護されたデータのスケーラブルな活用を可能にする．しかし「非負制約の逸脱」や「疎データの密度急増」という問題は回避できない．けれども，この Privelet 法に非負精緻化処理を組み込むと，高い部分和精度を維持しつつ，これらふたつの問題への対処も可能となる．

この手法の場合，非負精緻化を伴う逆 Wavelet 変換（Top-down 精緻化）部分に枝刈り処理を導入することで演算を効率化することができる．筆者らは以前，Top-down 精緻化の性質に着目した枝刈り実装法（水平型）を提案した．本報告では，先の提案とは異なる実装法（垂直型）を新たに提案する．さらに，先に提案した実装法との間での，演算効率化効果の比較評価も試みる．

キーワード：プライバシ保護，差分プライバシ，ウェーブレット変換，非負制約

Developing an Efficient Privelet with Non-negative Refinement

Abstract: Privelet is a data publishing technique that ensure ϵ-differential privacy while providing accurate answers for range-count queries. This technique is suitable for scalable utilization of privacy-preserved data. However, it has two problems which are “deviation from the non-negative constraint” and “abruptly increase of data-density”. Our non-negative refinement solves these two problems without losing the accuracy of the partial summation. In this method, it is possible to improve the efficiency of calculation by introducing pruning processing in the inverse Wavelet transform with nonnegative refinement - the top-down refinement. We have proposed a pruning implementation method - the horizontal type - focused on characteristics of the top-down refinement. In this report, we propose a new implementation method - the vertical type - different from the previous proposal. Additionally, we will try to compare and evaluate the efficiency improvement effect with the previously proposed implementation method.

Keywords: privacy-preserving data utilization, differential privacy, wavelet transform, non-negative constraint

1. はじめに

デジタルデータの蓄積が加速される今日，蓄積されたいわゆるビッグデータから抽出・加工された大規模集計データの有効活用を図ることは，新たな産業分野の創生に対する大きな足掛かりとなる可能性がある．しかし，あらゆる日常シーンが情報通信ネットワークと融合している現代においては，多方面から収集，抽出，蓄積された集計データが，人々の消費活動や，日常的な生活行動などと結びついたものであることも少なくない．そこで，大規模集計データの有効活用を考えるとき，プライバシ保護の観点に立った高い安全性を確保することが，極めて重要となってくる．

近年，ビッグデータの有効活用とデータに対するプライバシ保護の両立を可能にする技術の研究が盛んに行われている．様々な産業活動で計量・蓄積されているビッグデータを広く見渡すと，商品の数や生物の個体数，事象の発生数や人口など，自然数，すなわち，非負値から構成される集計データであることが少なくない．また，データが，人口密集地や商業地などのような固有の特性を有するエリアのみに集中するような，全体的には疎（sparse）な分布をとる傾向もしばしば見られ，そしてこの傾向は，特に，集計データの規模が大きくなるほど生ずる可能性が高まる．そこで本研究では，元のデータベースに含まれる個々のデー

[1] 北海道科学大学
Hokkaido University of Science
[2] 株式会社 NTT ドコモ
NTT DOCOMO, Inc.

タの集合体（個票）から，何らかの条件を満たすデータの個数を数えた数値データの集合体であり，さらに，全体的に疎な分布をとるような集計データを対象とする．

集計データに対するプライバシ保護に関しては，古くから検討が行われて来ている．これらは統計的開示制御（statistical disclosure control）[1][2] と呼ばれ，そこではセル秘匿基準や $n-k\%$ 基準などに基づく各種の秘匿方式が専門家によって注意深く適用されており，長年にわたって安全性が確保されて来た [3][4]．しかし，ビッグデータに基づく大規模集計データでは，値の小さな大量のセル値に対しても，切り捨てるのではなく，活用することが望まれる．そこで近年，プライバシを保護しつつも有用なデータを有効に活用するための，新たな基準や手法に対する様々な研究が盛んに進められている．こうした技術はプライバシ保護データ公開（PPDP）と呼ばれ，$k-$ 匿名性基準 [5] や $l-$ 多様性 [6]，$m-$ 不変性 [7] など，さまざまな技術が提案されている．

しかし，これらの PPDP 技術で前提とするところの，攻撃者の目的や能力，保有知識はそれぞれ異なっており，安全性を統一的に議論することは難しい．そうした中，近年，Dwork らが提案した差分プライバシ基準 [8], [9] が，高い安全性を実現するための基準として注目を集めている．これは，データベースから集計データを作成した際に，「ある特定のデータがデータベースに含まれているか否かを集計データから判別することが困難である」ことを安全性の根拠とするプライバシ保護基準である．この基準を満たす処理手法は，従来手法と異なり，背景知識や攻撃手法に依存しない数学的な安全性を備えることが保証されている．

この差分プライバシ基準を満たす代表的な手法に Laplace メカニズムがある．この手法は，集計データの各セル値に対して，平均値が 0 の Laplace ノイズ（Laplace 分布に従う独立な乱数）を付加するものである．しかしこの Laplace メカニズムを，大規模集計データに適用すると，「非負制約の逸脱」「部分和精度の劣化」「計算量の増大」といった問題への対処が必要となる [10][11]．「非負制約の逸脱」とは，Laplace メカニズムが適用されたデータに，本来の集計データには存在し得ない負値が多く含まれることである．本来存在し得ない負値が大量に含まれることで，プライバシが保護された集計データを利用する際の利便性が損なわれる事態が懸念される．一方，「部分和精度の劣化」とは，複数セルの部分和をとった際に，元データの値に対する誤差値が大きくなる現象をさす．これは，Laplace ノイズが付加されたセル値の部分和をとる際に，付加されたノイズが多重に作用することにより誤差値が増大してしまい，集計データの利用価値が低下するような事態をさす．部分和処理は，プライバシが保護された集計データのスケーラブルな利活用を行う上で，重要な特性といえる．また，「計算量の増大」とは，Laplace ノイズの付加により，集計データにおける非 0 値の割合（密度）が増大してしまう現象である．特に大規模な集計データにおいてはその影響が顕著であり，計算量やデータ量が現実的では無くなってしまう可能性も懸念される．なお本稿では，処理効率の改善手法とその効果を検討対象としていることから，「演算処理の効率」と「非 0 データの増大の問題」という異なるふたつの問題に対する混同が発生することによる混乱を避けるため，この現象（計算量の増大）については「疎データの密度急増」という，現象の本質部分に着目した呼称を用いることとする．

これら 3 点の課題を同時に解決する手法として，我々は非負精緻化を伴う Privelet 法を提案した [10][11]．これは，Xiao らによって提案された Privelet 法 [12][13] が有する，部分和精度が高いという性質を維持しつつも，「非負制約の逸脱」に対する回避と，「疎データの密度急増」の抑制を同時に実現する手法である．さらに我々は以前，Top-down 精緻化（非負精緻化を伴う逆 Wavelet 変換）部分の性質に着目した枝刈り実装法（水平型）を提案した [14]．この手法は，枝刈りによる演算の省略を行うことで効率化を図るものであり，データへの依存性はあるものの，北海道のメッシュ人口データを使った評価実験において，演算時間を 1/3 以下にまで抑制できることが確認できた [15]．しかし我々は，この枝刈り処理の動作を分析することで，先の実装法とは異なる，新たな実装法（垂直型）の可能性を見出した．本報告では，この，新たな枝刈り実装法について詳しく述べるとともに，先に提案した実装方法との比較評価も行う．

2. 非負精緻化を伴う Privelet 法

Privelet 法は，データ（長さ $n=2^H$ のベクトル列 $V=\{v_1,v_2,\ldots,v_n\}$）に対して Haar 基底に基づく離散 Wavelet 変換 (HWT)$\mathcal{H}$ を導入し，その Wavelet 係数に対して Laplace メカニズムを適用した上で，逆 Wavelet 変換 $\mathcal{H}^{-1}$ を施すことで，差分プライバシ基準を満たすデータ V^* を得る手法である．しかし，この方法で得られた集計データは，ノイズの影響により，非負制約を逸脱する．さらに，本来ゼロ値であった大量のデータが非ゼロ値となるため，疎データの密度急増も伴うことになる．

非負精緻化を伴う Privelet 法 [11] では，HWT により得られた Wavelet 係数に対して，Laplace メカニズムの適用，および，非負精緻化を伴う逆 Wavelet 変換を適用して，差分プライバシを満たすデータ V^+ を得ている．オリジナルの Privelet 法に，非負精緻化処理を加えることで，非負制約の逸脱を回避するとともに，疎データの密度急増を抑制している．なお，文献 [11] では，Top-down 精緻化の構成法として，直列構成法と並列構成法という 2 つの構成法を提案しているが，ここでは，枝刈り処理による演算効率化

が期待できる並列構成法を使用する．

2.1 Haar Wavelet 変換

はじめに，Haar Wavelet 変換部分の処理について概説する．基本となる処理は，n 個の入力データに Haar Wavelet 変換 $\mathcal{H}$ を適用して，$n/2$ 個の近似係数ベクトル cA，および，同じく $n/2$ 個の詳細係数ベクトル cD を得る，Haar 分解と呼ばれる処理である．まず，集計データベクトル $V = \{v_1, v_2, \ldots v_n\}$ を対象に，Haar 分解処理を適用する．

$$cA = \left(\frac{v_1 + v_2}{2}, \frac{v_3 + v_4}{2}, \ldots \frac{v_{n-1} + v_n}{2}\right),$$
$$cD = \left(\frac{v_1 - v_2}{2}, \frac{v_3 - v_4}{2}, \ldots \frac{v_{n-1} - v_n}{2}\right),$$

続いて，Haar 分解によって生成された $n/2$ 個の近似係数ベクトル cA に対して，再び Haar 分解を施すことで，$n/4$ 個の近似係数ベクトルと，同じく $n/4$ 個の詳細係数ベクトルが得られる．同様に，Haar 分解処理を再帰的に繰り返すことで，最終的に，$n-1$ 個の詳細係数ベクトルと，1 個の近似係数ベクトルが得られ，これらが HWT の出力 W となる．

2.2 Top-down 精緻化

次に，Top-down 精緻化を行って，差分プライバシ基準を満たす集計データを生成する．Top-down 精緻化では，HWT により得られた詳細係数ベクトル cD への Laplace メカニズムの適用，逆 HWT の適用，および，非負精緻化処理という 3 つの処理を行う．

2.2.1 Laplace メカニズムの適用

詳細係数ベクトル cD への Laplace メカニズムの適用では，次式に従って cD を構成する全ての要素へ Laplace ノイズを加算することにより，差分プライバシを満たす詳細ベクトル cD^* を得る [11].

$$cD^*_{h,x} = cD_{h,x} + Lap\left(\frac{1}{2^h\epsilon}\right)$$

ここで，$h = \{0, 1, \cdots, H-1\}$ は階層番号を，$x = \{0, 1, \cdots, 2^{H-h} - 1\}$ は階層内ノード番号を，$Lap(\cdot)$ は Laplace 分布に従う乱数を，また，ϵ は，Laplace 分布のスケールを規定するパラメータを表す．なお，最下層 $(h = 0)$ にあたる $n = 2^H$ 個の変数群はリーフと呼ばれ，秘匿対象および結果がここに格納される．

2.2.2 逆 HWT の適用

逆 HWT の適用については，図 1 を用いて説明する．HWT により得られた係数ベクトルの各要素は，図 1 のように 2 分木の各ノードに対応させて配置することができる．これらのうち，HWT の出力 W として保存されたのは，$cA_{H-1,0}$ および $n-1$ 個の $cD_{h,x}$ である．そしてこの段階では，Laplace メカニズムの適用により $cD_{h,x}$ には Laplace ノイズが付加されて，$cD_{h,x}$ の値は $cD^*_{h,x}$ へと変化している．

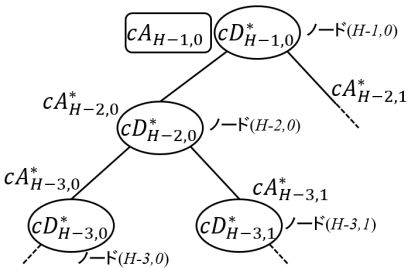

図 1 Top-down 精緻化
Fig. 1 The top-down refinement

逆 HWT 処理は，HWT 処理と逆のプロセスをたどる．すなわち，2 分木の上側からの処理となる．ノード $(H-1, 0)$ には，詳細係数 $cD^*_{H-1,0}$ と近似係数 $cA_{H-1,0}$ とが割り付けられているが，次式で表す演算により一階層下に位置する 2 つの近似係数を求める $(h = H-1, x = 0)$.

$$cA^*_{h-1,2x} = cA_{h,x} + cD^*_{h,x} \quad (1)$$
$$cA^*_{h-1,2x+1} = cA_{h,x} - cD^*_{h,x} \quad (2)$$

次に，ノード $(H-2, 0)$ に着目する．上記の演算により得られた近似係数 $cA^*_{H-2,0}$ の値と，ノード $(H-2, 0)$ に格納されている詳細係数 $cD^*_{H-2,0}$ とを用いて，同様の演算を行い，一階層下に位置する 2 つの近似係数を求める．同様に，ノード h, x に対応する近似係数 $cA^*_{h,x}$ と詳細係数 $cD^*_{h,x}$ とから，一階層下に位置する 2 つの近似係数 $cA^*_{h-1,2x}$ および $cA^*_{h-1,2x+1}$ を求める演算を再帰的に実行して行く．こうすることで，最終的に差分プライバシを満たす集計データベクトル V^* が得られる．

Privelet 法では cD に対して Laplace メカニズムが適用されているために，各ノードに対応する詳細係数 $cD_{h,x}$ が異なる値に変化してしまっていることから，逆 HWT の結果得られる集計データベクトル V^* は，入力と異なる値となり，データの秘匿が実現される．しかしその一方で，一連の処理に起因して，非負制約の逸脱や疎データの密度急増といった副作用が発生することになる．そこで，逆 HWT の過程に，非負精緻化処理を導入する．

2.2.3 非負精緻化の導入

近似係数 $cA_{h-1,2x}$ や $cA_{h-1,2x+1}$ の値は，それぞれ対応するノード $(h-1, 2x)$ および $(h-1, 2x+1)$ に集約される入力データの平均値であるから，HWT への入力データが非負値であった場合，必ず非負値となる．しかし，Laplace メカニズムの適用により，逆 HWT の過程で，$cA^*_{h-1,2x}$ や $cA^*_{h-1,2x+1}$ を算出する際に使用される $cD^*_{h,x}$ の値がもとの値と異なっていることから，負の値をもった $cA^*_{h-1,2x}$ や $cA^*_{h-1,2x+1}$ が生ずる場合が発生する．上述したように，$cA^*_{h-1,2x}$ の値も $cA^*_{h-1,2x+1}$ の値も，対応するノード $(h-1, 2x)$ および $(h-1, 2x+1)$ に集約される入力データの平均値であるから，$cA^*_{h-1,2x}$ や $cA^*_{h-1,2x+1}$ の値が負値

であることは，出力データに多数の負値が発生する事態を招く可能性がある．そこで，$cA^*_{h-1,2x}$ や $cA^*_{h-1,2x+1}$ の値に負値が生じないように一階層上の $cD^*_{h,x}$ の値を精緻化する．

ノード $(H-2,0)$ に着目する．式 (1, 2) と同様にして，一階層下の近似係数を求める $(h=H-2, x=0)$．$cA^*_{h-1,2x}$ または $cA^*_{h-1,2x+1}$ のいずれかが負値となった場合，$cD^*_{h,x}$ の値を，符号は変えずに，その絶対値を $cA^*_{h,x}$ へと置き換えることにより，非負精緻化が施された詳細係数 $cD^+_{h,x}$ を得る．

$$cD^+_i = \begin{cases} cD^*_{h,x} & (\text{if } cA^*_{h,x} \geq |cD^*_{h,x}|) \\ sign(cD^*_{h,x}) \cdot cA^*_{h,x} & (\text{otherwise}). \end{cases} \tag{3}$$

なお，$sign(\cdot)$ は，入力値の符号を返す関数とし，次式のように定義する．

$$sign(x) = \begin{cases} -1 & (\text{if } x < 0) \\ +1 & (\text{otherwise}) \end{cases} \tag{4}$$

そして，この非負精緻化を施した $cD^+_{h,x}$ の値を用いて改めて次式の処理を行うことにより，一階層下に位置する，非負精緻化が施された 2 つの近似係数を求め直す $(h=H-2, x=0)$.

$$cA^+_{h-1,2x} = cA^*_{h,x} + cD^+_{h,x} \tag{5}$$

$$cA^+_{h-1,2x+1} = cA^*_{h,x} - cD^+_{h,x} \tag{6}$$

ここで，$cA^+_{h-1,2x}$ および $cA^+_{h-1,2x+1}$ は，$cD^*_{h,x}$ に対して非負精緻化処理を施した結果に基づいて求められた近似係数の値を表す．上記に示す非負精緻化が施された $cD^+_{h,x}$ の効果により，$cD_{h-1,2x}, cD_{h-1,2x+1}$ の値は，ともに非負値となる．そしてその結果，最終的に出力される差分プライバシ基準を満たす集計データ V^+ の値も全て非負値となる．

3. 演算効率化の試み

3.1 枝刈り処理の概要

上述した非負精緻化を伴う Privelet 法では，一旦非負精緻化処理が発生すると，その片側子ノードから下は，全ての詳細係数の値が 0 となる．これは，非負精緻化処理を施したノードでは詳細係数と近似係数との絶対値が等しくなることから，式 (5) または式 (6) いずれかの近似係数の値 ($cA^+_{h-1,2x}$ または $cA^+_{h-1,2x+1}$) が必ず 0 になることに起因する．例えば，もしも近似係数の値が 0 のノードに対応する詳細係数が 0 以外の値をとると，またそこで非負精緻化処理が発生することとなり，そのノードの詳細係数の値は 0 へと精緻化される．そしてこうした非負精緻化処理が繰り返されることにより，詳細係数の値が 0 となったノードに連結される下層のノードでは，近似係数・詳細係数ともに全て 0 となり，その結果，当該ノードに連結されている出力値も全て 0 となる．

この性質に着目すると，近似係数 (および詳細係数) の値が 0 となるノードの演算を省略し，非負精緻化を伴う Privelet 法の処理を効率化することができる．ここではこの効率化処理を“枝刈り”と呼ぶことにする．

3.2 水平型実装

第 2.2.2 項で述べた通り，逆 HWT 処理は最上位層（層内のノード数が 1 の層）から，一層ずつ下りながら，順次演算を行って行く．説明のため，図 2 に示すように，最下層（リーフ）の層番号を $h=0$, 最上位層の層番号を $h=H-1$ とする．一方，各層では層内に含まれるノードを順次訪問しながら，非負精緻化が組み込まれた逆 HWT 処理を繰り返す．同一層内のノードの訪問順序に特に制約はないので，ここでは，図 2 に示すように，左側から連続したノード番号 x を付与し $(0 \leq x < 2^{H-h}-1)$，このノード番号の順番に則って層内各ノードの訪問を行うこととする．

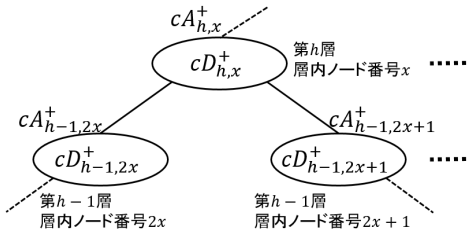

図 2 層番号と層内ノード番号
Fig. 2 The layer number and the node number in a layer

各ノードにおいて演算を省略するか否かは，近似係数 $cA^+_{h,x}$ の値が 0 であるか否かで判定する．上述した通り，近似係数 $cA^+_{h,x}$ の値が 0 であれば，非負精緻化処理の結果，詳細係数 $cD^+_{h,x}$ の値は必ず 0 となる．すなわち，式 (5) および式 (6) の値は，ともに 0 となる．したがって，近似係数 $cA^+_{h,x}$ の値が 0 の場合には，計算せずとも $cD^+_{h,x}=0$ となることが自明であり，この演算を省略することができる．さらに，HWT の場合，2 分木構造をもっていることから，階層間での対応が容易である．処理が一層ずつ下るにつれて，同一層内に含まれるノード数は 2 倍となり，また，第 h 層で x 番目に位置したノードに対応するノードは，第 $h-1$ 層では $2x$ および $2x+1$ 番目に位置することとなる．

図 3 に，階層間におけるノードの対応と演算省略ノード数メモリの様子を示す．一点鎖線より上側が第 h 層を，下側が第 $h-1$ 層をそれぞれ表している．楕円がノードを，長方形が演算省略ノード数メモリをそれぞれ表している．楕円内には層番号と層内ノード番号の組が，長方形内には演算省略ノード数メモリを表す変数ベクトル $Z=(z_{1,0}, z_{1,1}, z_{1,2}, \ldots, z_{H-1,0})$ の要素とその値が記されている．

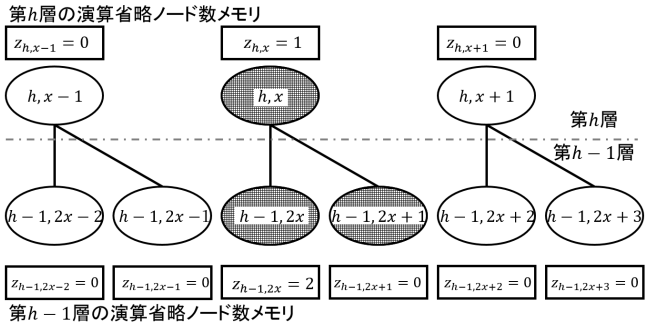

図 3 層間におけるノードの対応と演算省略ノード数メモリ
Fig. 3 Correspondence of nodes between layers and storage mechanism of the number of nodes to be omitted.

上述したような層間におけるノード関係から，いま，ノード (h,x) で $cA^{+}_{h,x}=0$ が検出され，演算が省略されるとき，第 $h-1$ 層ではそこに連結される 2 つの下位ノード $(h-1,2x)$ および $(h-1,2x+1)$ においても，$cA^{+}_{h-1,2x}=0$ および $cA^{+}_{h-1,2x+1}=0$ が検出されることとなり，一層下るごとに，省略できるノード数が 2 倍になることがわかる．そこで，Top-down 精緻化処理において $cA^{+}_{h,x}=0$ を検出した層内ノード番号 x と演算省略ノード数 $z_{h,x}$（初期値は 1）を記録し，さらに一層下るごとに，層内ノード番号 x ならびに演算省略ノード数 $z_{h,x}$ を 2 倍して行けば，非負精緻化を組み込んだ逆 HWT 演算の過程で記録された層内ノード番号 x の位置から $z_{i,x}$ ノード分の演算を一気に省略可能であることが期待できる．ただし，最下層（出力層）においてのみ，演算を省略する部分に相当するノードにゼロ値を格納する必要がある点には注意が必要である．

3.3 垂直型実装

2.2 節で説明した通り，非負精緻化を伴う逆 Wavelet 変換処理は，各層の各ノードにおいて，自ノードの近似係数 $cA_{h,x}$ の値と詳細係数 $cD_{h,x}$ の値から，直下ノードの近似係数 $cA_{h-1,2x}$ および $cA_{h-1,2x+1}$ の値を求めて行く．これらの処理は，同一層内では他のノードとの相互作用が存在せず，また，下層側から見て層間相互作用を有する上位ノードはひとつ（すなわち複数の上層側ノードとの相互作用は存在しない）という構造をもっている．この性質に着目すると，この処理は階層ごとに行う必要はなく（すなわち同一層内の全てのノードの処理を終えてから次の階層内のノードの処理に移行する必要はなく），層間結合の範囲のみを対象に深さ方向（層間方向）に処理を進めて行って構わないことになる．そこで，この処理を，最上位層から深さ方向に進めて行くことを考える．

このような処理は，再帰呼出し方式での実装に適している．再帰呼出しは，手続き型プログラミングにおいて，あるまとまった処理ブロックをモジュール化（例：関数化）し，そのモジュール内で自分自身を呼び出す方式である．本実装方式における非負精緻化を伴う逆 Privelet 変換処理の流れを，図 2 を参照して説明する．

本実装では，ひとつのノード（例：ノード (h,x)）における処理をモジュール化する．この処理モジュールでの基本的な演算処理は，自ノードの近似係数 $cA^{*}_{h,x}$ と詳細係数 $cD^{*}_{h,x}$ を用いて一階層下の近似係数 $cA^{*}_{h-1,2x}$ および $cA^{*}_{h-1,2x+1}$ を求めることである．

$$cA^{*}_{h-1,2x}=cA^{*}_{h,x}+cD^{*}_{h,x} \tag{7}$$

$$cA^{*}_{h-1,2x+1}=cA^{*}_{h,x}-cD^{*}_{h,x} \tag{8}$$

ここで，$cA^{*}_{h-1,2x}$ または $cA^{*}_{h-1,2x+1}$ のいずれかに負値が発生した場合には，2.2.3 項で説明した非負精緻化処理を行った上で，改めて式 (7) および式 (8) に従って，ふたつの近似係数の値を求め直す．

再帰呼出しによる実装を行う上では，上記の処理を終えたところで，左下のノード $(h-1,2x)$ および右下のノード $(h-1,2x+1)$ に対するこの処理モジュール自身の呼出しを行う．こうすることで，上位層の処理を終えた直後に，直下の 2 ノードの処理へと降りて行く．

本再帰呼出しではふたつの終了条件を設定する．ひとつ目の終了条件は，枝刈り適用に対する判定条件である．第 3.2 節でも述べた通り，枝刈りの適用条件は，$cA^{+}_{h,x}$ の値が 0 であるか否かである．もうひとつの終了条件は，最下層ノードの判定条件である．再帰呼出しの引数に階層番号のカウンタを含め，最下層ノードを超えた場合 $(h\leq 0)$ には処理を行わないように設定する．なお，これらの終了条件の判定は，このモジュールにおける一連の処理の最初に設けておく．非負精緻化を伴う Privelet 法の枝刈り処理を再帰呼出しによって実装することで，水平型実装に不可欠な演算省略ノード数メモリの管理が不要となり，プログラミングの労力を軽減できることが期待できる．

4. 評価

4.1 評価方法

本評価では，異なる複数のエリアにおける人口分布データに対して Privelet 法による秘匿処理を適用し，非負精緻化を組み込んだ，水平型および垂直型それぞれの実装方法による逆 HWT 処理部分の処理時間を計測する．“(α) 枝刈りあり”と“(β) 枝刈りなし”との間の処理時間の差を“(β) 枝刈りなし”の処理時間で正規化した値 $(\beta-\alpha)/\beta$ をここでは“時間短縮率”と呼ぶこととし，演算量抑制効果の指標とする．

本評価では，平成 22 年度国勢調査に基づく地域メッシュ人口（1km メッシュ）のデータに対して，一次元 Haar Wavelwt 型の，非負精緻化を伴う Privelet 法を適用し，差分プライバシを規定するパラメータの値は $\epsilon=0.1$ とした．また，条件をそろえた形で水平型実装と垂直型実装との実行時間の比較を行うため，予め摂動値データを作成しておき，各ノードには同じ摂動値を加えるようにした．日本全

Vol.2019-DPS-178 No.36
Vol.2019-CSEC-84 No.36
2019/3/5

国（2^{11}メッシュ × 2^{11}メッシュ）のデータから,(1) 北海道（$2^9 \times 2^9$）,(2) 四国（$2^8 \times 2^8$），(3) 関東（$2^8 \times 2^8$）の各エリアを切り出して評価用のデータとした．また，他エリアとの比較用に，隣接する縦横 2 × 2 メッシュの人口をひとまとめにした (4) 北海道 1/4（$2^8 \times 2^8$）も評価用データに加えた．

なお，本評価を行うにあたって，二次元上に配置された地域メッシュ人口データを，一次元化する方式による差異についても確認するべく，(a) ラスター方式，(b) ソート方式（降順），(c)Morton 方式 [11]，(d) ランダム方式という 4 方式によって一次元化を行い，各々に対して評価を行った．(a) ラスター方式では，2 次元に配置されているメッシュ人口データを，X 軸方向に取り出す操作を，Y 軸方向に繰り返すことでデータの 1 次元化を行っている．(b) ソート方式のデータは，(a) ラスター方式で得られた 1 次元データを，その値の大きい順（降順）に並べ替えたものである．(c)Morton 方式のデータは，2 次元に配置されているメッシュ人口データに Morton 写像を施したものである．Morton 写像は，多次元空間から一次元空間への全単射を行う写像であり，元の空間上における距離の遠近が写像先の空間における距離の遠近に反映される性質を持つ，局所性保存写像の一種である．(d) ランダム方式のデータは，(a) ラスター方式で得られた 1 次元データに対して一様乱数を用いた並べ替え処理を施したものである．なお，(b) ソート方式は差分プライバシ基準を満たさないが，他 3 方式と比較する目的で加えている．

評価には Intel Core i7-875K CPU (2.93GHz)，実装メモリ 4GB のデスクトップ PC を使用した．また，同一処理を 100 回繰り返した時間を計測して 1/100 し，計測時間の精度向上を図った．

4.2 評価結果

表 1 に，水平型実装による処理時間の計測結果を示す．4 方式間での時間短縮率の傾向を見ると，ソート方式において最も時間短縮率が高く，ランダム方式において最も低い．また，同一メッシュ数をもつ 3 エリア間での時間短縮率の傾向については，北海道 1/4 の値が最も高く，ランダム方式以外関東の値が最も低い．さらに，メッシュ数の多い北海道エリアは，著しく高い時間短縮率を示している．

表 2 に，垂直型実装による処理時間の計測結果を示す．4 方式間での時間短縮率の傾向は水平型実装による処理時間と同様，ソート方式において最も時間短縮率が高く，ランダム方式において最も低い．また，同一メッシュ数をもつ 3 エリア間での時間短縮率の傾向についても，北海道 1/4 の値が最も高く，関東の値が最も低い．さらに，メッシュ数の多い北海道エリアについても同様に，著しく高い時間短縮率を示している．

表 3 に，非負精緻化発生回数とゼロ値含有比率の値を示す．すべての条件において，垂直型実装の方が，水平型実装よりも時間短縮率が高い値となっている．また，処理結果のゼロ値含有比率は，ソート方式の値が最も高く，ランダム方式で最も低い．そして，同一メッシュ数をもつ 3 エリア間での処理結果のゼロ値含有比率を比較すると，北海道 1/4 の値が最も高くて関東の値が最も低く，元データの

表 1 水平型実装の処理時間
Table 1 Processing time of horizontal implementation

一次元化方式	エリア	処理時間 (100 ループの平均) (α) 枝刈あり [ms]	(β) 枝刈なし [ms]	処理時間比率 [%] α/β	時間短縮率 [%] $(\beta-\alpha)/\beta$
(a) ラスター方式	北海道	23.7	45.1	52.5	47.5
	四国	10.2	11.3	90.2	9.9
	関東	10.6	11.2	94.2	5.8
	北海道 1/4	8.4	10.8	77.7	22.3
(b) ソート方式	北海道	4.3	44.5	9.6	90.4
	四国	3.2	11.2	28.4	71.7
	関東	5.2	11.1	46.7	53.3
	北海道 1/4	1.8	10.7	16.9	83.1
(c) Morton 方式	北海道	15.2	45.4	33.6	66.4
	四国	7.6	11.2	67.9	32.1
	関東	8.7	11.1	78.0	22.0
	北海道 1/4	5.8	10.7	54.7	45.3
(d) ランダム方式	北海道	35.9	44.7	80.2	19.8
	四国	14.9	11.3	131.6	-31.6
	関東	14.9	11.6	128.8	-28.2
	北海道 1/4	12.1	11.1	109.0	-9.0

表 2 垂直型実装の処理時間
Table 2 Processing time of vertical implementation

一次元化方式	エリア	処理時間 (100 ループの平均) (α) 枝刈あり [ms]	(β) 枝刈なし [ms]	処理時間比率 [%] α/β	時間短縮率 [%] $(\beta-\alpha)/\beta$
(a) ラスター方式	北海道	5.5	36.9	15.0	85.0
	四国	3.5	9.1	37.9	62.1
	関東	4.6	9.0	51.5	48.5
	北海道 1/4	2.3	9.2	24.4	75.6
(b) ソート方式	北海道	1.3	36.9	3.6	96.4
	四国	1.6	9.1	17.4	82.7
	関東	3.1	8.8	35.1	64.9
	北海道 1/4	0.6	9.1	6.8	93.2
(c) Morton 方式	北海道	3.7	37.0	10.1	89.9
	四国	2.6	9.0	28.2	71.8
	関東	4.2	8.8	47.2	52.8
	北海道 1/4	1.7	9.2	18.2	81.8
(d) ランダム方式	北海道	9.0	37.0	24.3	75.7
	四国	5.6	9.1	61.5	38.5
	関東	7.5	9.0	83.8	16.2
	北海道 1/4	3.6	9.2	38.8	61.2

ゼロ値含有比率の傾向を反映している．

表 3　水平型実装と垂直型実装の時間短縮率比較
Table 3　Time reduction rate of horizontal and Vertical implementation

一次元化方式	エリア	元データのゼロ値含有比率 [%]	水平型実装の時間短縮率 [%]	垂直型実装の時間短縮率 [%]	処理結果のゼロ値含有比率 [%]
(a) ラスター方式	北海道	95.0	47.5	85.0	92.7
	四国	78.7	9.9	62.1	72.9
	関東	61.2	5.8	48.5	55.4
	北海道 1/4	90.3	22.3	75.6	85.3
(b) ソート方式	北海道	95.0	90.4	96.4	96.9
	四国	78.7	71.7	82.7	83.3
	関東	61.2	53.3	64.9	64.8
	北海道 1/4	90.3	83.1	93.2	93.0
(c) Morton 方式	北海道	95.0	66.4	89.9	94.3
	四国	78.7	32.1	71.8	76.2
	関東	61.2	22.0	52.8	58.1
	北海道 1/4	90.3	45.3	81.8	87.9
(d) ランダム方式	北海道	95.0	19.8	75.7	87.9
	四国	78.7	-31.6	38.5	56.6
	関東	61.2	-28.8	16.2	30.5
	北海道 1/4	90.3	-9.0	61.2	86.2

5. 考察

5.1 一次元への写像方式と時間短縮率の関係

今回，二次元データから一次元データへ写像する複数の方式を評価している．4 つの写像方式の時間短縮率を見てみると，いずれの実装形態でも，第 4.2 節でも述べた通り，ソート方式において時間短縮率が最も高く，次いで Morton 方式，ラスター方式となっており，ランダム方式において最も低いことが表 1 および表 2 からわかる．これは，データの局所性が高いほど，本演算効率化手法による効率向上の効果が高いことを示唆している．

ソート方式は，人為的にデータの局所性を最大に高めていることから，最大の効果がもたらされたといえる．この方式は，第 4.1 節でも触れた通り，差分プライバシを満たさないことから実用に供するものでは無いものの，本演算効率化手法の適用効果に関する理論的な最大値を示している．

Morton 方式は，ソート方式に次いで効率向上の効果が高い結果を示している．これは，元々のデータが「人口分布」という地理的偏在性が高いデータであり，Morton 写像が，写像元の二次元空間上で「近い」距離にある点を，写像先の一次元空間でもなるべく「近い」距離に配置するように写像する，局所性保存写像の性質を備えているためと考えられる．無論，元の二次元データに偏在性が無ければ局所性保存写像を用いても効果は無いが，地理空間データなどの実世界のデータはロングテール性を持つ，すなわち，偏在性が高い傾向にあるため，局所性保存写像の適用は人口分布に限らず，他の実データへの適用にも広く有効であることが期待される．

ラスター方式は，Morton 方式とランダム方式の間に位置している．これは，ラスター方式の場合，X 軸方向の局所性が保持され，Y 軸方向の局所性は基本的に保持されないことに起因していると考えられる．この結果は．ラスター方式の場合，データの局所性，すなわち，一次元配列上における隣接データ間で差分値の小さい部分が，ランダム方式ほどではないものの，細分化された状態で大量に分布していることを示唆している．

ランダム方式は，ソート方式とは逆に「最悪」のケースに対する評価となっている．ランダム方式の評価結果の場合，垂直型実装においては全ての条件で効率化の効果がみられるものの，データの局所性を失わせることで，水平型実装のように，本演算効率化手法による効率向上の効果がほとんど得られないばかりか，悪化する場合さえあることを示している．このことは，データが持つ局所性を保存する形で一次元データへの写像を行った上で本演算効率化手法を適用することが重要であることを裏付けている．

5.2 重み付き枝刈り発生回数と時間短縮率の関係

次に．枝刈り処理の発生回数と時間短縮率の相関について考察を行う．第 2 章で述べた通り，本手法は 2 分木構造を基盤としている．すなわち，ひとつの最下層ノードで枝刈りが発生した場合，省略される演算数は 2（2 分木で接続される左右各ひとつずつの最終結果：リーフ値）であるが，その一階層上で枝刈りが発生した場合に演算が省略されるノード数はその 2 倍の数が加算された値となる．したがって，階層 h を起点とする枝刈りの発生回数を p_h とするとき，枝刈りにより省略される近似係数演算の総数（ここでは“重み付き枝刈り発生回数”と呼ぶ）の値 WP は，次式によって求めることができる．

$$WP = \sum_{h=1}^{H-1} (p_h \cdot \sum_{i=1}^{h} 2^i)$$

図 4 に，重み付き枝刈り発生回数と時間短縮率の関係を示す．グラフから，両者の間に高い正の相関関係のあることがわかる．

以上の考察結果から，より高い階層を起点とする枝刈りが発生するほど，演算が省略されるノード数の値が大きくなること，すなわち，演算が省略されるノード数が増えると考えられる．一般に，均一な値が広い範囲で連続している場合，上位階層まで詳細係数 $cD_{h,x}$ の値が 0 近傍の値となり，ノイズの付加によって負値が発生し，非負精緻化が起こり，枝刈りが発生する可能性が高くなる．したがって，均一な値が広く分布するようなデータに本手法を適用した

場合に，大きな時間短縮率が期待できることが示唆される．

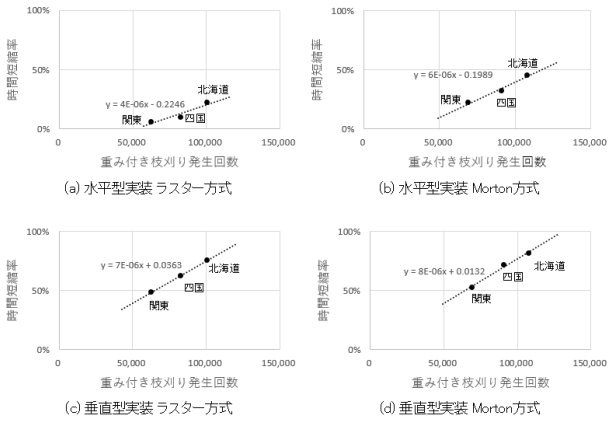

図 4　重み付き枝刈り発生回数と時間短縮率の相関

Fig. 4　Correlation between weighted pruning count and time reduction rate

5.3　水平型実装と垂直型実装の時間短縮率比較

さらに，水平型実装と垂直型実装の時間短縮率を比較する．表 3 から，すべての条件において，垂直型実装の方が，水平型実装よりも時間短縮率が高いことがわかる．特に，ランダム方式において顕著な差異が見られる．ランダム方式の場合，まとまった省略が発生しにくいことから，水平型実装に必要となる，省略するノード数の処理によるオーバーヘッドを上回る効率化が難しいことに起因しているものと考えられる．

6.　おわりに

本報告では，新たに垂直型の枝刈り実装法の提案を行った．一次元への写像方式と時間短縮率の関係は，先に提案した水平型の実装法と同様の性質を示した．また，重み付き枝刈り発生回数と時間短縮率の関係についても，水平型と同様に，高い相関がみられた．さらに，時間短縮率においては，水平型実装よりも，大きな時間短縮率が得られることが示された．今後は，例えば処理結果の誤差率や部分和精度など，時間短縮率以外の，より詳細な性質の検討を行うことが課題と考える．

謝辞　本研究は日本学術振興会科学研究費補助金基盤研究 (C)（課題番号：15K00190）による助成を受けて行なわれたものである．

参考文献

[1] Hundepool, A., Domingo-Ferrer, J., Franconi, L., Giessing, S., Lenz, Longhurst, J., Nordholt, E. S., Seri, G., and Wolf, P. -P.: *Handbook on Statistical disclosure control*, Statistics Netherlands (2010).

[2] Hundepool, A., Domingo-Ferrer, J., Franconi, L., Giessing, S., Nordholt, E. S., Spicer, K., and Wolf, P. -P.: *Statistical Disclosure Control*, John Wiley & Sons (2012).

[3] 統計センター：統計データ開示制御に関する用語集（改訂版），製表関連国際用語集，No.2 (2005).

[4] 集計表におけるセル秘匿問題とその研究動向，統計数理，Vol.51，No.2，pp.337-350 (2003).

[5] Sweeney, L.: *k-anonimity: A model for protecting privacy*, Intl. J. Uncertainty, Fuzziness and Knowledge-Based Systems, Vol.10, No.5, pp.557-570 (2002).

[6] Machanavajjhala, A., Kifer, D., Gehrke, J. and Venkitasubramaniam, M.: *l-diversity: Privacy Beyond k-anonymity*, ACM Trans. Knowledge Discovery from Data (TKDD), Vol.1, No.1 (2007).

[7] Xiao, X. and Tao, Y.: *m-Invariance: Towards Privacy Preserving Re-publication of Dynamic Datasets*, Proc. 2007 ACM SIGMOD Intl. Conf. Management of Data, pp.689-700, ACM (2007).

[8] Dwork, C.: *Differential Privacy*, Proc. 33rd Intl. Conf. Automata, Languages and Programming - Volume Part II, Bugliesi, M., Preneel, B., Sassone, V. and Wegener, I.(Eds.), Lecture Notes in Computer Science, Vol.4052, pp.1-12, Springer (2006).

[9] Barak, B., Chaudhuri, K., Dwork, C., Kale, S., McSherry, F., Talwar, K. and Berkeley, U. C.: *Privacy, accuracy, and consistency too: a holistic solution to contingency table release*, Proc. 26th ACM SIGMOD-SIGACT-SIGART symp. Principles of database systems - POD'S07, ACM Press, 273282 (2007).

[10] 寺田，竹内，齊藤，本郷: 差分プライバシー基準に基づく情報秘匿手法の一考察, マルチメディア，分散，協調とモバイル *(DICOMO2014)* シンポジウム, pp.224-233 (2014).

[11] 寺田，鈴木，山口，本郷: 大規模集計データへの差分プライバシの適用, 情報処理学会論文誌, Vol.56, No.9, pp.1801-1816 (2015).

[12] Xiao, X., Wang, G. and Gehrke, J.: Differential privacy via wavelet transforms, *Proc. 26th Intl. Conf. Data Engineering (ICDE2010)*, pp.225-236, IEEE(2010).

[13] Xiao, X., Wang, G., Gehrke, J. and Jefferson, T.: Differential Privacy via Wavelet Transforms, *IEEE Trans. Knowledge and Data Engineering*, Vol.23, No.8, pp.1200-1214 (2011).

[14] 本郷，手塚，寺田，稲垣: Top-down 精緻化を伴う Privelet 法における演算効率化手法の検討, マルチメディア，分散，協調とモバイル (DICOMO2018) シンポジウム講演論文集, pp.460-466 (2018).

[15] 本郷，大加瀬，手塚，寺田，稲垣，鈴木: 集計データへの差分プライバシ適用における特性の一考察 III, 暗号と情報セキュリティシンポジウム講演論文集, No. 1C1-1, pp.1-8 (2019).