時空間データ分析のための差分ヒストグラム構築手法

趙　セイ† 石川　佳治† 杉浦　健人† 脇田佑希子†

† 名古屋大学大学院情報科学研究科　〒 464–8603 愛知県名古屋市千種区不老町

E-mail: †{zhao,sugiura}@db.ss.is.nagoya-u.ac.jp, ††ishikawa@is.nagoya-u.ac.jp, †††wackie@nagoya-u.jp

あらまし　科学分野における観測データやシミュレーション結果のデータの分析においては，同じ対象物に関する複数のデータを比較して，どのような差異があるかを検出したいという要求がしばしば見られる．例えば，ある空間領域に対する異なる時間における観測データやシミュレーションデータの比較，異なったパラメータによる空間的なシミュレーションの比較などが考えられる．そこで本論文では，時空間的な観測データやシミュレーションデータなどを蓄積する時空間データウェアハウスにおける差分演算について提案する．それに加えて，差分を分析するための要件を整理し，その実装に関するアプローチを示す．

キーワード　データウェアハウス, 差分演算, 時空間データベース, シミュレーションデータ

1. はじめに

さまざまな分野においてビッグデータが注目されている今日では，データベースにおける大量のデータの高度な分析処理を行うための**データアナリティクス**（data analytics）について，研究が盛んに行われている [6]．**時空間データベース**（spatio-temporal database）においても，行動データ，移動軌跡データ，科学分野のデータなどのさまざまな領域において大規模な時空間データの分析が求められている．我々の研究グループでは，特に時空間的な大規模シミュレーションの結果を蓄積し，対話的な分析を可能とするための**時空間データウェアハウス**（spatio-temporal data warehouse）に関する研究を進めており，その考え方を**シミュレーションデータウェアハウス**（simulation data warehouse）と呼んでいる．特に津波・震災シミュレーションデータの分析に関して研究を進めている [5]．

本稿では，時空間データウェアハウスにおける基本的な分析要求の一つとして考えられる**差分**（もしくは**差異**）（difference）に着目する．時空間的なデータでは，特に時間に関する変化をどのように検出するかが重要な要求の一つとして考えられる．また，パラメータが異なるシミュレーションデータや観測状況が異なる観測データが与えられたとき，異なるデータ間にどのような違いがあるかを検出することも求められる．例として，以下のような避難シミュレーションデータの分析の例を考える．

> 対象となるシミュレーションの空間領域において，時区間 $T_1 = [8:00, 9:00]$ および $T_2 = [20:00, 21:00]$ における避難者の数の分布について，顕著な差異がある領域を報告せよ．

この分析例では，時間帯 T_1 と T_2 において，避難者の数の分布にどのような違いがあるかを求めている．

差分演算のイメージを図 1 に示す．例えば，左図と中図はそれぞれ時区間 T_1 と T_2 における集計結果を表している．各セルの濃淡は集計値の大きさ（その時間帯にそのセルに属する移動ユーザの人数）に対応している．なお，実際の利用においては，二つの時区間 T_1, T_2 における集計値をそのまま用いるか，総数で割った正規化された集計値を用いるかという選択肢があり，これは対象データや応用に依存する．ここでは簡単のため，ユーザが前者の方式を選んだものとして進める．

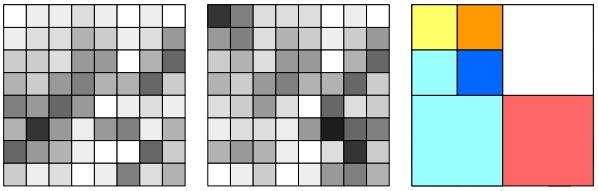

図 1　差分演算のイメージ

図 1 の右図は，時区間 T_1, T_2 における入力データに対し，T_1 の集計結果に対する T_2 の集計結果の差分を近似的に表したイメージを示している．ヒートマップ表現を用い，赤い色が強い領域ほど T_2 の時区間において T_1 時区間と比較して集計値が大きいことを示す．一方，青い色が強いほど，逆の傾向があることを意味する．また，大まかな差分の傾向を表現するため，差分のトレンドが同じようなセルが隣接する場合にはそれらをまとめて一つのセルとして表現する．ここでは四分木（quadtree）のような空間分割を想定し，セルの辺の大きさが 2 のべき乗の長さになるようにしている．

このような出力結果の提示により，ユーザは，二つの時区間における変化を容易に把握することができる．しかし，差分演算としてどのようなものを考えればいいかは明らかではなく，対象のデータの性質や応用目的に依存してさまざまなものが考えられる．一方で，大量のデータの中から顕著な変化を検出することは容易ではなく，最新のデータベースシステム技術を効果的に活用した効率的なアルゴリズムの開発が必要となる．このような背景に基づき，本研究では:

（1） 時空間データウェアハウスから差分・差異を検出するための汎用的な演算を定義して，ヒストグラムに基づく手段を用いて時空間データの顕著な差分を検出することを目指す．

（2） 大規模配列データの管理および問合せに特化した配列指向型 DBMS である SciDB [2], [3], [4] を活用して差分演算を実装する．SciDB に時空間シミュレーションデータが分析用に蓄積されているものとし，必要に応じてそれに対して差分演算が適用されるというのが想定シナリオである．

本論文の構成は以下の通りである．2 章では差分演算の定義について述べる．3 章は提案手法の説明である．節 3.1 と節 3.2 は差分ヒストグラムの構築手法について述べ，節 3.3 では，大規模なデータを取り扱うとき計算コストを減らすため，要約情報に基づく差分誤差の推定手法を提案する．4 章で SciDB における実装について述べて，5 章は評価実験である．最後に，6 章でまとめと今後の課題について議論する．

2. 問題の定義

ここでは，本研究で想定する典型的な時空間データについて述べる．2 次元のユークリッド空間を細粒度で何分割するかというパラメータ n が与えられるものとし，空間を $2^n \times 2^n$ 分割する．一方で，時間の間隔 τ が与えられており，時間は等間隔の時区間に分けられる．

[定義 1] **時空間配列** $\mathcal{A}_{\mathrm{st}}(X, Y, T, Atr)$ は次元と属性で構成される．次元については，具体的には，$2^n \times 2^n$ 空間グリッドにマッピングされる空間次元 X, Y，と等間隔の時区間で構成される時間次元 T となる．なお，$X = \{1, 2, ..., 2^n\}$, $Y = \{1, 2, ..., 2^n\}$, $T = \{1, 2, ..., T/\tau\}$．時空間配列において，各要素 $e_{i,j,k}$ （$x_i \in X$, $y_j \in Y$, $t_k \in T$）は次元によって位置付けられて，属性の集合 Atr を有する．本研究では，属性については，数値データの集計値を対象として想定する．例えば，避難のシミュレーションデータにおいて，属性は避難の人数の集計値となる．

具体的な集計値の計算方法としては，まず，その時区間および対象セルに関するレコード数をカウントすることが考えられる．選択肢として，分布のパターンを見たいなら，全レコード数で割った頻度分布とすることも考えられる．また，他の種類の対象データに関しては，AVG や MAX などの集計関数の利用も考えられる．

時空間配列データを分析するため，時間に関する変化を検出することや空間上の大幅な分布を把握することは，重要な要求の一つとして考えられる．そこで本研究では，差分・差異を検出するための汎用的な演算を定義する．

[定義 2] 与えられた時空間配列 $\mathcal{A}_{\mathrm{st}}$ において，時区間 t_a と t_b における属性 v の集計結果の差分を**差分配列** $\mathcal{A}_{\mathrm{diff}}(X, Y, \delta(t_a, t_b))$ と呼ぶ．差分の定義（差，比例など）は分析のシナリオによって変わっているが，ここでは，時区間 t_a と t_b の集計値の差を考慮する．

[定義 3] **差分演算** $Diff(\mathcal{A}_{\mathrm{st}}, t_a, t_b, v, B)$ は，差分配列 $\mathcal{A}_{\mathrm{diff}}(X, Y, \delta(t_a, t_b))$ を**最良な差分ヒストグラム** H で表現する．差分ヒストグラム $H = \{b_1, b_2, ..., b_B\}$ は対象となる空間領域 (X, Y) をカバーする B 個のセルで構成される．各セル $b \in H$ は $\{x, y, \delta(t_a, t_b)\}$ （$x \in X, y \in Y$）を含め，空間セル (x, y) における時区間 t_a と t_b の属性の集計値の差分 $\delta(t_a, t_b)$ を表す．また，空間領域がセル b にカバーされる差分配列の要素の集合を

$$cov(b) = \{e \mid e \in \mathcal{A}_{\mathrm{diff}} \wedge contain(area(b), area(e))\} \quad (1)$$

で示す．ただし，area は XY 平面における領域を返す関数で，contain は前者が後者を包含するときに真になる関数である．

最良な差分ヒストグラムは誤差が最小なもので定義される．ここでは，二乗誤差を用いて誤差の尺度とする．すなわち，差分ヒストグラム H の誤差を

$$\mathrm{Error}(H) = \sum_{b \in H} \sum_{e \in cov(b)} \left(e(v) - \delta(t_a, t_b)\right)^2 \quad (2)$$

で定義する．つまり，差分配列 $\mathcal{A}_{\mathrm{diff}}$ における要素の属性値 $e(v)$ と差分ヒストグラムのセルの値の差の二乗である．

3. 提 案 手 法

最良な多次元ヒストグラムを構築する問題は NP 困難と証明されている [1]．それで，本研究では，近似的な方針として，四分木の構造に基づき，ボトムアップに統合を行い差分ヒストグラムを構築する．主に二つの手法が考えられる．まず，貪欲的な統合手法は，ボトムアップに誤差が最小となる統合可能なセル群を見つけて，ヒストグラムのサイズの制約（B）を満たすまで統合を行うものである．一方で，閾値に基づく統合手法は，閾値を用いて，誤差が閾値以下のセルを統合し，ボトムアップにヒストグラムを構築する手法である．両者について，以下で具体的に説明する．

なお，差分ヒストグラムの構築においては，差分配列を直接の入力とする考え方をとることもできるが，大規模なデータを取り扱う際には，最初に配列データをスキャンして差分配列を計算するコストが高いという問題がある．一方で，必ずしも差分配列を先に計算せずに，差分ヒストグラムを構築しながら必要となる差分を計算する手法も考える．よって本研究では，統計情報に基づく手法も提案する．グリッド構造に対して事前に集計・計算した統計情報にアクセスして差分誤差の区間を推定することによって，条件を満たさないセルをフィルタリングすることができる．これによって，必要のない配列データをチェックするコストを減らして，差分を計算しなくても条件を満たすセルを見つけることもできる．

3.1 ヒストグラム構築手法 1：貪欲的な統合手法

ヒストグラム構築手法 1 として，貪欲な統合手法に基づくアルゴリズムについて考える．基本方針は，隣接したセルをボトムアップに統合していくというものである．このアルゴリズムをアルゴリズム 1 に示す．

入力の $\mathcal{A}_{\mathrm{diff}}$ は，対象となる差分配列である．入力のもう一つは，差分ヒストグラムのセルの総数 B である．$B = 1$ の場合は一切分割しないため処理は自明であるので，このアルゴリズムでは $B > 1$ の場合を扱う．また，差分ヒストグラムでは視覚的な提示を想定しているため，結果となる差分ヒストグラムの細粒度のセルの大きさは元の配列データの細粒度の大きさである必要はない．そこで，ユーザあるいはシステム側で対応

Algorithm 1: Greedy-based Merging

Input: $\mathcal{A}_{\text{diff}}$: 差分配列, B : 差分ヒストグラムのセルの総数, m : 初期レイヤのパラメータ

Result: H: 差分ヒストグラムのセル集合

```
1  H ← merge(A_diff, 2^m, 2^m) ;      // 差分ヒストグラムを初期化
2  curL ← 0 ;                          // ヒストグラムのレイヤを初期化
3  while H.size > B do
4      if curL == c_opt.layer then
5          CL ← merge(H, 2, 2) ;       // 候補セルリスト生成
6          curL++;
7      end
8      c_opt ← get_min_error(CL)  ;  // 誤差が最小なセルを抽出
9      H ← c_opt ∪ (H \ children(c_opt)) ;  // ヒストグラムを更新
10     CL ← CL \ c_opt ;               // 候補リストを更新
11 end
12 return H;
```

するパラメータ m を与えることを想定する．差分配列が 2 次元の空間上の $N \times N$ のグリッドとしたとき，差分ヒストグラムの細粒度を $\frac{N}{2^m} \times \frac{N}{2^m}$ の空間グリッドと設定する．

本手法では，まずパラメータ m を用いて，$2^m \times 2^m$ のグリッドサイズで差分ヒストグラムを初期化し，すべてのグリッドセルを一時的な結果 H に入れる (1 行目)．例えば，図 2(a) では，m が 0 と指定された場合の初期ヒストグラムは入力とする差分配列のデータと一致する．対応するレイヤ $curL$ を 0 にする．1 行目と 5 行目に出てくる **merge**() 関数では，指定された配列の領域に対してグリッドサイズ $2^m \times 2^m$ でセルを統合して，対応する配列の属性を集計する．ここでは，差分値を表す属性の平均をとって集計を行う．**merge**() 関数によって，ヒストグラムの初期化と候補セルリストの生成を行う．ボトムアップに統合を行うので，新たに結果に入れたセルは一番上のレイヤに属する限り，新たな候補セルを生成する（4-7 行目）．

差分ヒストグラムのセル数が B より大きくなくなるまで，貪欲的に誤差が最小となるセルをヒストグラムに入れる（8-10 行目）．8 行目の get_min_error(CL) は，与えられた候補セルリスト CL において，誤差の値が最小となるセルを返す関数である．9 行目の children(c_{opt}) は，セル c_{opt} の領域にカバーされる子のセルからなる集合を返す関数である．これらによって，ヒストグラムを更新する．

3.1.1 計算量の分析と効率化

このアルゴリズムでは，while ループを 1 回実行するたびに，4 個のセルが統合されて 1 個のセルになることから，結果的に 3 個のセルが減少することになる．$H.size$ の初期値が C であるため，$C - 3p \geqq B$ を満たす最大の p が繰り返しの回数となり，p の値は

$$p = \left\lfloor \frac{C - B}{3} \right\rfloor \tag{3}$$

で与えられる．なお，C の大きさはパラメータ m の設定に依存しており，可視化の際の応答性と結果の精度のバランスを考慮する必要がある．m が大きいほど計算コストが少なくなり，可視化処理における応答が良くなるが，結果の精度が低くなる．逆に，m が小さいほど結果の精度は高くなるが，あまりに小さい設定では，視覚的には有効ではない小さいセルが差分ヒストグラムに残る可能性がある．

一方，処理の過程で何度も呼ばれる誤差の計算については，実際のデータの集計が必要となることから，大規模なデータに対して支配的な影響を与えうる．これについては，後述のようにデータベースシステムの集計機能を活用することや，計算済みの値を保持するなどの工夫で実際のコストを削減できる．具体的には，各レイヤで計算済みの統計情報を保持して，上のレイヤのセルの誤差を計算する際に再利用される．つまり，以下の式を用いて，一個下のレイヤにおけるセルの統計情報（例，平均値など）を用いて，新たに生成される候補セルの誤差を計算できる．

$$n = \sum_j n_j \tag{4}$$

$$m = \frac{\sum_j m_j n_j}{n} \tag{5}$$

$$E(c) = \sum_j E_j + \sum_j m_j^2 n_j - m^2 n \tag{6}$$

m，n，$E(c)$ はそれぞれセル c の平均値，含まれている細粒度のセルの数と誤差である．m_j，n_j，E_j はそれぞれセル c の子のセル c_j $(j = 1, 2, \ldots)$ の平均値，含まれている細粒度のセルの数と誤差である．例えば，図 2(c) では，左下のセルの誤差を計算するとき，カバーしている領域に存在する細粒度のセルをチェックする必要はなく，一個下のレイヤ（図 2(b)）で保持された統計情報と式 (4),(5),(6) を用いて計算できる．これらによって，ボトムアップに誤差を計算しながら統合を行うことができる．

3.1.2 差分ヒストグラムの構築例

差分配列と $B = 10$ が与えられたときの，差分ヒストグラムの構築例を図 2 に示す．パラメータ m が 0 の場合，初期ヒストグラムは与えられた差分配列と一致する．図 (b) に示したように，一回目の統合では誤差が最小（0）となるセルを差分ヒストグラムに入れる．そして，差分ヒストグラムのセル数が B 以下となるまで，この方針で統合を行う．ボトムアップに統合を行いながら，計算済みの統計情報を保持して，式 (6) によってセルの誤差を計算できる．結果となるヒストグラムを図 2(c) に示す．

3.2 ヒストグラム構築手法 2：閾値に基づく統合手法

ヒストグラム構築手法 1 では，候補セル数が多い場合，計算コストも高くなる．そこで，閾値に基づく手法も提案する．基本的なアイデアは次のようになる．統合を行うとセルの誤差が大きくなるが，分布が比較的に均一である（誤差が小さい）セルを統合すると，結果の誤差も一般に小さくなる．ここでは，$B - 1$ 番目に大きい誤差値 E^{B-1} を閾値として用いる．統合の方針は，四分木の構造に基づき，誤差が E^{B-1} より小さい 4 つの兄弟のセルを統合して，対応する親セルを結果に入れるというものである．このアルゴリズムを 2 に示す．

手法の流れはアルゴリズム 1 と類似し，以下のように三つの

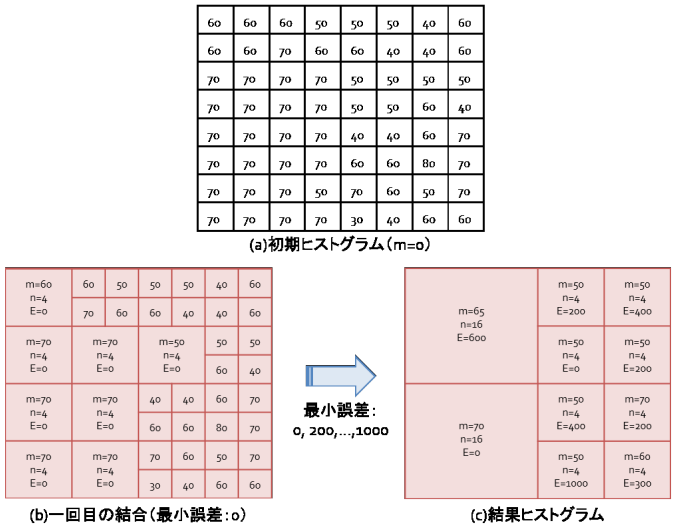

図 2　実行例 1 ：貪欲的な統合手法（$B = 10$）

Algorithm 2: Threshold-based Merging

Input: $\mathcal{A}_{\text{diff}}$: 差分配列, B : 差分ヒストグラムのセルの総数, m : 初期レイヤのパラメータ

Result: H: 差分ヒストグラムのセル集合

```
1  H ← merge(A_diff, 2^m, 2^m);
2  curL ← 0, trd ← 0, triggerM = 0;
3  while H.size > B do
4      if triggerM == 1 then
5          CL ← merge(H, 2, 2) ;          // 候補セルリスト生成
6          curL++, triggerM = 0;
7      end
8      trd ← get_trd(CL)   ;              // 閾値を計算
9      for c ∈ CL do
10         if ∀c_child ∈ children(c), Error(c_child) < trd then
11             H ← {c} ∪ (H \ children(c)) ; // ヒストグラム更新
12             CL ← CL \ {c} ;            // 候補リスト更新
13             if c.layer == curL then
14                 triggerM = 1
15             end
16         end
17     end
18 end
19 return H;
```

ステップとなる．

- ステップ 1 ：初期化（1,2 行目）

指定された初期レイヤのパラメータ m を用いて対象領域を $\frac{N}{2^m} \times \frac{N}{2^m}$ のグリッドに分けて，すべてのグリッドセルを差分ヒストグラム H に入れる．$m = 0$ の場合全てのセルの誤差は 0 となり，統合の必要はないため，$m \geqq 1$ を想定する．図 2 で示した例と同じ差分配列と B が与えられたとき，初期ヒストグラムは図 3(a) に示す 4×4 のグリッドとなる．

- ステップ 2 ：統合（3-17 行目）

$merge()$ 関数を用いて，候補結果となるセルリスト CL を生成する．4 行目に出てくる $triggerM$ は統合を行うかどうかを指示するパラメータである．結果に入れた候補セルが一番上のレイヤに属する限り，新たな候補セルリストを生成する（4-7 行目，13-15 行目）．候補セルの誤差については，節 3.1.1 に述べたように，式 (6) に基づき，子のセルの統計情報を用いて親セルの誤差を計算する．

候補セルリストにおいて，10 行目の条件を満たすセルを結果に入れる．つまり，候補セル c の 4 つの子のセルの誤差が閾値より低い場合，c をヒストグラム結果に入れて，すべての子のセルを結果から削除する．図 3 の例では，一回目の統合処理において，左下の 4 つのセルの誤差が閾値（200）より小さいので，統合を行って，親のセルを結果に入れる．このアルゴリズムでは，閾値は定数ではなく可変であり，閾値を更新しながら，条件を満たすセルを見つけてヒストグラムを構築する．

- ステップ 3 : セル数が B 以上であるならば，ステップ 2 を繰り返す．統合できるセルが存在しない場合，閾値を緩めて，$[E^{B-2}, E^{B-3}, ..., E^1]$ の区間で調整することが考えられる．図 3 では，一回目の統合結果に対して，閾値 E^{B-1} は 200 であるので，条件を満たす統合は存在しない．しかし，ヒストグラムのサイズの制約 B を満たすため，閾値を調整して 300 とすると，図 3(c) の結果ヒストグラムが得られる．

実行処理の回数については，アルゴリズム 1 と同様の分析が行える．ただし，実際の計算においては，1 回の繰り返しの中で複数の統合処理を実施できるので，繰返し数を減らすことができる．アルゴリズム 1 の場合と同様，パラメータ m の設定については，計算コストと結果の精度のトレードオフが存在する．

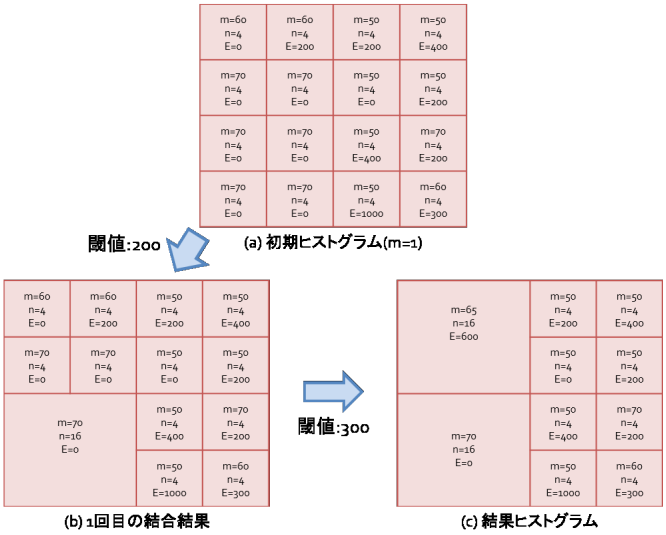

図 3　実行例 2 ：閾値に基づく統合手法（$B = 10$）

3.3　改善手法：要約情報に基づくアプローチ

上記の二つの差分ヒストグラム構築手法では，差分配列を入力としてヒストグラムを構築するが，I/O コストと計算量が高いことは問題となる．そこで，本節では，事前に計しておいた要約情報に基づいて，差分ヒストグラムを構築するアプローチを提案する．

3.3.1 要約情報に基づく誤差推定

［定義 4］ **要約情報** $S(\mathcal{A}_{\mathrm{st}}, t, m)$ は対象となる空間領域をグリッドに分けて，与えたれた時間帯 t （$t \in T$）で各グリッドセル内の配列データを集計して（例：sum），対応する統計情報を保持する．グリッドのレイヤはパラメータ m によって決まる．つまり，$S(\mathcal{A}_{\mathrm{st}}, t, m)$ の解像度は $\frac{N}{2^m} \times \frac{N}{2^m}$ となる．各要約情報セル sc では，平均値，最大・最小値，セル数と二乗誤差を格納する．例えば，図 4 では，配列 $\mathcal{A}_{\mathrm{T_a}}$ と $\mathcal{A}_{\mathrm{T_b}}$ はそれぞれ右図の 2×2 解像度の要約情報に対応する．

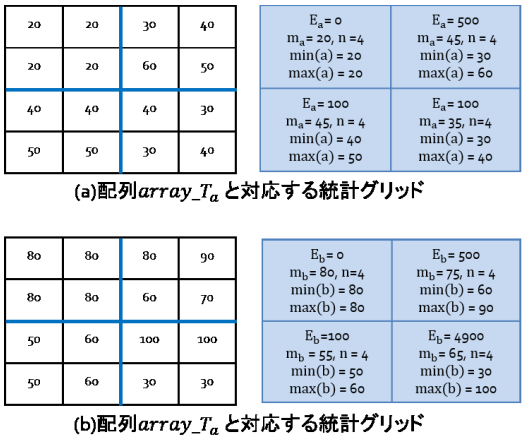

図 4　要約情報の例

与えられた配列 $\mathcal{A}_{\mathrm{T_a}}$ と $\mathcal{A}_{\mathrm{T_b}}$ の要約情報によって，対応する差分ヒストグラムの誤差の区間を推定できる．式 (2) により，差分ヒストグラムのセル b_i の誤差は $E(b_i) = \sum_{c \in b_i}[(v_b(c) - v_a(c)) - (m_b - m_a)]^2$ で計算される．なお，$v_a(c)$ と $v_b(c)$ はそれぞれセル c の時間 T_a と T_b の属性値であり，m_a と m_b はそれぞれセル b_i における時間 T_a と T_b の属性の平均値である．また，配列 $\mathcal{A}_{\mathrm{T_a}}$ と $\mathcal{A}_{\mathrm{T_b}}$ のセルの誤差は式 $E_a(b_i) = \sum_{c \in b_i}(v_a(c) - m_a)^2$ と式 $E_b(b_i) = \sum_{c \in b_i}(v_b(c) - m_b)^2$ で計算される．これらによって $E(b_i)$ の式を書き換えると，次の式となる．

$$E(b_i) = E_a(b_i) + E_b(b_i) + 2m_a m_b n - \sum_{c \in b_i} v_a(c)v_b(c) \quad (7)$$

ここでは，式の後ろの要素 $\sum_{c \in b_i} v_a(c)v_b(c)$ を var で示す．さらに，要約情報に保持される属性値の最大値，最小値，平均値など統計情報によって， var の下限値と上限値は次の式：

$$var_{lb} = \max(\min(v_a(c)) \times m_b n, \min(v_b(c)) \times m_a n) \quad (8)$$

$$var_{ub} = \min(\max(v_a(c)) \times m_b n, \max(v_b(c)) \times m_a n) \quad (9)$$

で推定される．そして，セル b_i の誤差の区間は以下の式で計算される．

$$E_{lb}(b_i) = E_a(b_i) + E_b(b_i) + 2m_a m_b n - var_{ub} \quad (10)$$

$$E_{ub}(b_i) = E_a(b_i) + E_b(b_i) + 2m_a m_b n - var_{lb} \quad (11)$$

3.3.2 要約情報に基づくヒストグラムの構築手法

配列データの要約情報を事前に計算しておけば，差分ヒストグラムを構築するときには全ての配列データにアクセスして差分配列を計算する必要はなく，要約情報によって最適解となる可能性のないセルをフィルタリングすることはできる．本手法は節 3.1 と節 3.2 で記述した差分ヒストグラムの構築手法に拡張できる．ここでは，貪欲的な方針に基づいて，手法の概観をアルゴリズム 3 に示す．

Algorithm 3: Statistic-based approach

Input: 要約情報 S_{T_a}, S_{T_b}, 差分ヒストグラムのセル総数 B, 初期レイヤのパラメータ m

Result: 差分ヒストグラムのセル集合 H

```
1  H ← apply(join(S_Ta(m), S_Tb(m), A_diff), delta, v_b − v_a) ; // 要約情報を用いて初期ヒストグラムを生成
2  CL ← get_error_interval(S_Ta(m − 1), S_Tb(m − 1)) ; // 候補セルリスト生成
3  while H.size > B do
4      c_opt←∅; CR ← CL ;                    // CR : 候補結果リスト
5      ub_min ← ∞ ; // 貪欲的なアプローチ : 誤差の上限値を初期化
6      for c ∈ CL do
7          if E_lb(c) ≧ ub_min then
8              CR ← CR \ {c} ; // 貪欲的なアプローチ : 誤差の下限値は最小な上限値より高いセルをフィルタリング
9          end
10         else if E_ub(c) < ub_min then
11             ub_min ← E_ub(c);             // 上限値を更新
12         end
13     end
14     c_opt ← c_opt ∪ get_opt(CR);   // 検証 : 実際の誤差値を計算し，条件を満たすセルを結果に入れる
15     update H, CL;
16 end
17 return H;
```

まず，入力の要約情報 S_{T_a} と S_{T_b} を用いて初期ヒストグラム H を生成する．本研究では，配列指向型 DBMS である SciDB で実装を行うため，SciDB のビルトイン演算を用いて初期ヒストグラムを構築する．つまり，1 行目に出てくる SciDB のビルトイン演算 $\mathbf{apply}(\mathbf{join}(S_{T_a}(m), S_{T_b}(m), array\_D), delta, v_b - v_a)$ を用いている．**join** 演算を使って配列 $S_{T_a}(m)$ と $S_{T_b}(m)$ を統合して，**apply** 演算で属性値の差 $v_b - v_a$ を新たな属性 $delta$ に入れる．

2 行目の $\mathrm{get\_error\_interval}(S_{T_a}(m-1), S_{T_b}(m-1))$ は初期ヒストグラムの一個上のレイヤのセルを候補セルリストに入れて，式 (10) と式 (11) を用いて対応する差分誤差の区間を計算する．次に，得られた差分誤差の区間によって，チェックする必要のないセルをフィルタリングする．

- フィルタリング手法 1（貪欲的なアプローチ）（6-13 行目）：誤差の下限値が最小の上限値よりも高いセルをフィルタリングする．誤差がセル c の誤差の下限値よりも小さいセルが存在するので，セル c は最小の誤差ではないことが分かる．そ

のため，セル c の実際の誤差をチェックする必要はなく，候補結果リスト CR から削除する．

- フィルタリング手法 2（閾値に基づくアプローチ）：$B-1$ に番目大きい誤差を閾値として，誤差が閾値より小さいセルを統合する方針であるので，$B-1$ 番目に大きい誤差の上限値 ub_{max}^{B-1} を使ってフィルタリングを行う．フィルタリング手法 2 は次のアイデアに基づく：セル c の誤差の上限値は ub_{max}^{B-1} よりも小さい場合，セル c の誤差は必ず閾値より小さいことが分かるため，実際の誤差を計算する必要はない．

フィルタリングを行って得られた候補結果リストに対して，検証を行う（14 行目）．具体的な配列データにアクセスして実際の差分誤差を計算して，条件を満たすセルを結果に入れる．

3.3.3 要約情報に基づく手法の実行例

貪欲的な手法を例として，要約情報に基づくヒストグラム構築の過程を図 5 に示す．図 5(a) は図 4 の要約情報によって推定された差分誤差の区間であり，図 5(b) は実際の配列データにアクセスして計算された差分の誤差値となる．アルゴリズムの実行例を図 5(c) に示す．

sc_1 E_1: [0,0]	sc_2 E_2: [0,6400]
sc_3 E_3: [0,2000]	sc_4 E_4: [2400,7600]

(a) 統計グリッドによる差分誤差の推定区間

E_1 = 0	E_2 = 900
E_3 = 100	E_4=5000

(b) 実際の差分誤差

ループ	CL	CR	実際の誤差値	c_{opt}
1	sc_1, sc_2, sc_3, sc_4	sc_1		sc_1
2	sc_2, sc_3, sc_4	sc_2, sc_3	E_2 = 900 E_3 = 100	sc_3
3	sc_2, sc_4	sc_2		sc_2
4 B=7: end				

(c) 実行例

図 5　実行例 3：要約情報を用いた貪欲的な手法

最初に，4 つの要約情報のセル $\{sc_1, sc_2, sc_3, sc_4\}$ は候補セルリスト CL に入っている．セル sc_1 をスキャンすると，ub_{min} は 0 となる（10,11 行目）．次に，フィルタリング手法 1 によって，sc_2, sc_3, sc_4 をフィルタリングすると，最適解 c_{opt} は sc_1 となる（7,8 行目）．2 回目のループでは，sc_4 はフィルタリング手法 1 により候補結果リストから削除されるので，sc_2 と sc_3 を比較する必要がある．そのため，実際の誤差を計算して，誤差が最小となる sc_3 を結果に入れる．3 回目のループでは，同様に sc_4 をフィルタリングして，sc_2 を結果に入れる．ここでヒストグラムのセル数の制約（$B=7$）を満たしたので，アルゴリズムを終了する．

この例では，実際に配列データにアクセスして差分と対応する誤差を計算したのは E_2 と E_3 のみとなる．E_1 と E_4 は要約情報に基づき誤差の区間を推定できるので，提案したフィルタリング手法によって，誤差を実際に計算することなく判断できる．

4. SciDB における実装

本研究では，配列指向型 DBMS である SciDB を活用して差分演算を実装する．SciDB では，大規模な配列データは複数のチャンクに分けられて，分散して SciDB クラスタに格納される．SciDB クラスタは複数のインスタンスで構成されており，インスタンスごとに複数のチャンクが格納される．このようなデータ構造によって，SciDB は並列分散処理を行う [3]．問合せ処理の過程では，あるインスタンスがコーディネータ（coordinator）として動作し，問合せプランを生成してほかのワーカインスタンスに割り当てる．次に，ワーカインスタンスは関連するチャンクデータを処理して中間的な結果を生成する．最後に，コーディネータはワーカインスタンスの結果を統合して，最後の結果を出す．

SciDB では，さまざまなビルトイン演算が提供されている一方，特定な要求に対応するためのユーザ定義演算（UDOs）もサポートされている．SciDB のプラグイン仕組みを用いることで，差分演算を UDO として実装する．差分演算は C++言語で記述する．

上記の SciDB の機能を活用して，本論文で提案したヒストグラム構築手法を実装する．チャンクは SciDB における最小の I/O 単位であるため，ボトムアップに統合を行うときにチャンクを単位として処理が行われる．チャンクごとにローカルな最適解を見つけて，最後にこれらのローカルな最適解をまとめて，グローバルな解を計算する．

5. 評 価 実 験

5.1 対象となるデータセット

本研究で使っているデータセットは，大規模地震の発生時の，高知市における人流の避難シミュレーションデータである．このデータは東京大学関本研究室からの提供による．今回実験で用いるサンプル人流は，地震発生後に約 4 万人の人が避難する状況を，パーソントリップデータに基づいてシミュレーションしている．

シミュレーションデータは $(id, time, x, y)$ という形式のレコードの集まりである．id はユーザ ID，$time$ は時刻，x, y は時刻 $time$ におけるユーザの位置である．シミュレーションデータは，シミュレーション後には変化しない静的なデータである．本研究が目的とするデータウェハウス的な問合せ処理においては，静的なデータであることの利点を生かし，前処理を行うことで対話的処理を効率化することが一般的である．そこで，今回のデータについては，空間的には $4,096 \times 4,096$ のレベルの最大粒度の分割を行う．この空間の細粒度の分割に基づき，各セル内で 1 分ごとにその中の避難者数を集計する．前処理の結果得られた集計データを SciDB にロードし，これを差分演算の問合せ処理の対象とした．また，対象となる配列データ Kouchi_eva のデータスキーマを以下に示す．

```
Kouchi_eva<Num:double>
[X=0:*,4096,0,Y=0:*,2048,0,T=0:*,256,0]
```

配列 Kouchi_eva では，座標 X,Y と時間 T という三つの次元があり，属性 Num は対応するセルの避難人数の総数である．

6. まとめと今後の課題

本稿では，時空間データの高度な分析機能を実現するため，差分演算に着目し，差分ヒストグラムの構築手法と効率化の手法を提案した．また，効率性と有効性を考慮した評価実験と，配列指向型 DBMS の機能をフルに活用した実装技術についても開発を行う予定である．

本論文で示した差分演算では，2 つの時区間 T_1, T_2 がユーザにより指定されることを想定していた．しかし，このようなアプローチは，事前にユーザが着目すべき時区間についてすでにアイデアを有しているときにしか使えないことから，より多くのケースに対応することが必要である．さらに今後の課題としては，差分演算のセマンティクスの拡張が考えられる．今回は単純に差の大小のみに着目したが，増加傾向にあるか，減少傾向にあるか，また増加・減少の速さはどの程度かといった情報に着目した差分演算も開発したいと考えている．このような差分を考える場合，どのように可視化するかもポイントとなることから，可視化手法についても検討を行う．

謝　　辞

本研究の一部は，科研費（16H01722, 26540043），CREST「大規模・高分解能数値シミュレーションの連携とデータ同化による革新的地震・津波減災ビッグデータ解析基盤の創出」，および文部科学省「実社会ビッグデータ利活用のためのデータ統合・解析技術の研究開発」による．避難シミュレーションデータを提供いただいた東京大学関本研究室に感謝します．

文　　献

[1] S. Muthukrishnan, V. Poosala, and T. Suel. On rectangular partitionings in two dimensions: Algorithms, complexity, and applications. In C. Beeri and P. Buneman, editors, *ICDT*, volume 1540 of *Lecture Notes in Computer Science*, pages 236–256. Springer, 1999.

[2] Paradigm4: Creators of SciDB a computational DBMS. `http://www.paradigm4.com/`.

[3] M. Stonebraker, P. Brown, A. Poliakov, and S. Raman. The architecture of SciDB. In *SSDBM*, volume 6809 of *LNCS*, pages 1–16, 2011.

[4] M. Stonebraker, P. Brown, D. Zhang, and J. Becla. SciDB: A database management system for applications with complex analytics. *IEEE Computational Science & Engineering*, 15(3):54–62, 2013.

[5] J. Zhao, K. Sugiura, Y. Wang, and Y. Ishikawa. Simulation data warehouse for integration and analysis of disaster information. *Journal of Disaster Research*, 11(2):255–264, 2016.

[6] 石川佳治. 大規模データアナリティクスに関する研究動向と展望. **電子情報通信学会論文誌** *D*, J97-D(4):718–728, 2014 年 4 月.