探索的データ解析におけるエンジンの効率化

松本　拓海† 山室　健†† 小笠原麻斗††† 佐々木勇和††† 鬼塚　真†††

† 大阪大学工学部電子情報工学科　〒 5650871 大阪府吹田市山田丘 1-5
††† 大阪大学大学院情報科学研究科　〒 5650871 大阪府吹田市山田丘 1-5
†† 日本電信電話株式会社　〒 1808585 東京都武蔵野市緑町 3-9-11
E-mail: †{matsumoto.takumi,yamamuro.takeshi,ogasawara.asato,sasaki,onizuka}@ist.osaka-u.ac.jp

あらまし　マーケティングデータなどのデータ分析を対象として, 近年, 有用性の高い分析結果を自動的に検出する探索的データ分析の技術が注目されている．探索的データ分析では, 多様な OLAP クエリあるいは多様な部分データを総当たりして有用性の高い分析結果を探索するため, 膨大な時間を要する問題がある．この問題に対して従来のクエリ最適化の技術として, 複数クエリの共有化手法および有用な分析結果の上位 k 件検索による探索空間の削減手法が存在するが, 前者はプランニング時の最適化技術であり実行プランが静的であることが必要であるのに対して, 後者は実行中の最適化技術であり動的に実行プランを変更する必要があるため, 両者の共存したものが存在していない．本稿では, プランニング時の最適化技術である複数クエリ共有化を実行中の探索空間の削減技術に複数クエリを同時実行する形で組み込み両立を図る. データをストリーム処理して複数クエリを同時に処理するとともに, 中心極限定理に基づいて有用性の低いデータキューブを推定して枝刈りするフレームワークを提案する．更に, 大規模データに対して異常検知の分析を行った結果を報告する．評価実験により, 大域例外データを探索する場合, 最大で 2 倍の高速化が可能であることを確認した．

キーワード　OLAP, 探索的データ解析, Data Cube

1. はじめに

大規模かつ多様性に富んだデータであるビックデータを資源と捉え, 有益な情報を抽出する技術が重要である．そのため, OLAP (Online Analytical Processing) によるデータ分析, 相関マイニング, クラスタリング, 回帰分析などの技術が研究されてきた．特に企業などが所持しているマーケティングデータの分析において, データに多次元的な集約処理を行いデータの特徴や異常を発見する OLAP 型の分析手法が頻繁に利用されている．分析結果を得たユーザはその特徴や異常なデータの外的要因を踏まえ有効な知見であれば企業の施策作成に役立てる．この分析手法において, データ分析とは仮説 · 検証の繰り返し作業である．しかし, 仮説を立てるためにはデータに関する深い知識が必要である．また, データの多様化により多数の仮説が必要である事とデータ量の増加が分析者にとって大きな負担となっている．

このような負担を排除するため, 分析を自動化する探索的データ分析 (Exploratory Data Analysis) の研究 [1] [2] [3] [4] [5] [6] [7] [8] が注目されている．SEEDB [3] [2] [4] では分析結果として高い有用性を有する分析パターンを探索し, 水野らの研究 [8] では有用性の高い部分データを探索する．ここで, 分析パターンとは次元属性, 集計属性, 集計関数のことであり, 部分データとはデータセットから特定の条件 (例, 都道府県=' 大阪府') に合致するデータ集合のことを指す．探索的データ分析では，必要なデータを抽出するのクエリを機械的に大量生成 · 実行し，そのクエリ結果から分析結果の有用性を測る評価関数を使用し，有益な分析結果を自動探索する．

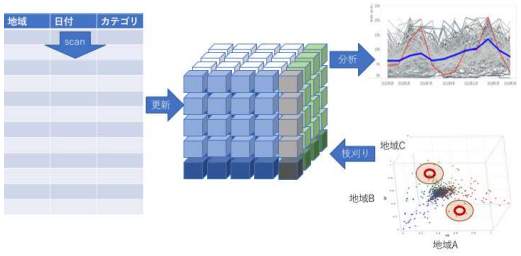

図 1　提案フレームワークの適用例

このように探索的データ分析技術は, 有用な分析結果を自動的に探索することができる強力な技術であるが, 多様な OLAP クエリあるいは多様な部分データを総当たりして有用性の高い分析結果を探索するため, 膨大な時間を要する問題がある．この問題を解決する有効なアプローチとして, 複数クエリを同時に処理することで処理コストを削減する方法と, 探索空間を枝刈りすることで有用性の高い上位 k 件を効率的に特定する方法が存在する．この複数クエリの共有化 [9] [10] においては, クエリ実行前に共有化プランを生成するためクエリの実行プランが静的であることが必要である．一方で, 枝刈りによる処理の高速化においては, クエリを実行中に打ち切るため, クエリ実行プランが動的に変化することになる．クエリ結果を取得する際に主に使われているツールとして一般的である DBMS では複数クエリの共有化と探索空間の削減手法は共存できない技術である．

本稿では, 自動的に OLAP クエリを発行する探索的データ分析ツールに適応可能な, 複数クエリの同時実行と上位 k 件の候補に成り得ない部分データの枝刈りを両立したフレームワークを提案する. プランニング時の最適化技術である複数クエリ共有化を実行中の探索空間の削減技術に複数クエリを同時実行する形で組み込み両立を図る. 図 1 は本提案フレームワークの適用例を示している. まず, 複数クエリを同時実行する方法は, データを逐次で処理する方法 [11] [7] が広く知られているので, これを採用し , クエリ結果をデータキューブの形式で保持する. そして, 上位 k 件検索による探索空間の削減手法に関しては, Online Aggregation の技術を適用する. 具体的には統計的信頼区間の技術 [12] [13] を利用することで, サンプルデータをもとに全体データに対するクエリ結果を推定する. 推定結果を用いて, 有用性の上位 k 件の候補に成り得ないデータキューブを枝刈りする. 本稿では, 提案フレームワークのプロトタイプを Spark の UDAF (User Defined Aggregation Function) を用いて実装し , 実データを用いた評価実験を行った. 評価実験により , 95%の信頼水準において大域例外データを探索する場合, 最大で 2 倍の高速化が可能であることを確認した.

本稿の構成は, 以下の通りである. 2. 章にて事前知識について説明し , 3. 章において提案フレームワークの詳細を示し, 4. 章でフレームワークの実装を説明する. 5. 章にて評価実験について説明し , 6. 章にて関連研究について述べ, 7. 章にて本稿をまとめ, 今後の課題について論ずる.

2. 事 前 知 識

本章では, 提案フレームワークに用いた信頼区間と信頼区間の推定に用いた確率不等式, 分析アプリケーションについて説明する. 2.1 節では中心極限定理 (CLT: Central Limit Theorem) について, 2.2 節では中心極限定理を用いた OLAP クエリ結果の区間推定について説明する. そして 2.3 節では本稿で使用した分析アプリケーションに関する問題定義や有用性の定義を説明する.

2.1 中心極限定理

確率不等式は, 任意の確率分布を有する母集団の確率変数に対して, 期待値や分散などの統計的な情報だけに基づいて, それらの確率変数の和あるいは平均に関する上限確率の上界を評価するものである.

中心極限定理とは, 母集団がどのような分布の確率変数でも母集団から無作為に n 個の標本を抽出することで得られる標本平均と真の平均との誤差が, サンプルサイズを大きくすると近似的に正規分布に従うという定理を表している.

ここで, 集約属性に対する実数値関数 v の集合を $\mathbb{F}$, データセットのサイズを m, サンプルサイズを n とすると, 真の集約値平均値 μ と標本集約値平均 $\bar{Y_n}$ は以下のように表わされる.

$$\mu = \frac{1}{m}\Sigma_{i=1}^{m} v(i) \ \ , v \in \mathbb{F} \tag{1}$$

$$\bar{Y_n} = \frac{1}{n}\Sigma_{i=1}^{n} v(i), v \in \mathbb{F} \tag{2}$$

具体的に, 集約関数 AVG, SUM である場合に限定して議論を進める. 集約値を計算する集合の要素数が n とすると, 集約関数が AVG の場合 $v(i) = i$ であり, SUM の場合 $v(i) = n * i$ である. 真の集約値平均値 μ, 標本集約値平均 $\bar{Y_n}$ と標本集約値分散 θ_2 を用いると中心極限定理は以下のように表される.

$$\frac{\sqrt{n}(\bar{Y_n} - \mu)}{\sqrt{\theta_2(v)}} \simeq N(0, 1), \quad n \to \infty, \ f \in \mathbb{F} \tag{3}$$

ただし, $N(0, 1)$ は標準正規分布 (平均が 0, 分散が 1) である.

2.2 中心極限定理を用いたクエリ結果の区間推定

信頼区間とは, 母数の値がどのような範囲に存在するかを確率的に表す方法である. 式 (3) より, 真の集約平均値 μ の信頼区間の式を導出する. $\bar{Y_n} - \mu$ が負の数になる場合も考慮すると, 以下のように表される.

$$Pr(\epsilon) = Pr[|\bar{Y_n} - \mu| \leqq \epsilon] \tag{4}$$

また, サンプルサイズ n が大きい場合, 分散 θ_2 の値は不変分散 $T_{n,\,2}$ 値に近似する.

$$T_{n,\,q}(v) = \frac{1}{n-1}\Sigma_{i=1}^{n}(v(i) - \bar{Y_n})^q \tag{5}$$

式 (3,5) を用いると , 式 (4) は以下のように表される.

$$P\{|\bar{Y_n} - \mu| \leqq \epsilon\} = P\{|\frac{\sqrt{n}(\bar{Y_n} - \mu)}{\sqrt{T_{n,\,2}(v)}}| \leqq \frac{\epsilon\sqrt{n}}{\sqrt{T_{n,\,2}(v)}}\} \tag{6}$$

$$\approx 2\Phi(\frac{\epsilon\sqrt{n}}{\sqrt{T_{n,\,2}(v)}}) - 1.$$

但し, $\epsilon \geqq 0$ であり, Φ は標準正規分布の確率変数の累積分布関数である. 母集団の平均値 μ の信頼係数が p である確率点 z_p は累積分布関数を使用すると, $\Phi(z_p) = \frac{p+1}{2}$ で計算することができる. 以上より, クエリ結果の信頼区間は式 (7) で求められる.

$$\bar{Y_n} - \epsilon_n \leqq \mu \leqq \bar{Y_n} + \epsilon_n, \qquad \epsilon_n = \sqrt{\frac{z_p^2 T_{n,\,2}(v)}{n}} \tag{7}$$

2.3 分析アプリケーション

有用性の定義に基づく評価関数を作成することで, ユーザが発見したい分析結果を定量的に評価することが可能となる. ここで全体データを D とし, D の部分集合を部分データ S で表現し, 部分データの集合を $\mathbb{S}$ とする. 全体データ D はレコード $d_1, d_2, \cdots, d_{|D|}$ ($|D|$ は全体データのレコード数) から構成される. レコード $d_i (1 \leqq i \leqq |D|)$ は集約属性と次元属性の集合で構成される. 集約属性の集合 A は, 集約属性 $a_1, a_2, \cdots, a_{|A|}$ ($|A|$ は集約属性の総数), 次元属性の集合 B は, 次元属性 b_1 , $b_2, \cdots, b_{|B|}$ ($|B|$ は次元属性の総数) で構成される.

次元属性 $b_i (1 \leqq i \leqq |B|)$ の値が Y である部分データ S と部分データ集合 $\mathbb{S}$ は, 関係代数における選択演算 σ を使用すると以下の式で定義される.

$$S := \sigma_{b_i = Y}(D) \tag{8}$$

$$\mathbb{S} := \bigcup_{i=1}^{|B|} \{\sigma_{b_i = Y}(D) \ | \ Y \in b_i\} \tag{9}$$

単一の集約属性 a_i と次元属性 b_i から成る OLAP クエリ q

は以下のように定義される．

$$q := {}_{b_i}G_{w=f(a_i)} \tag{10}$$

但し，G は各レコードを次元属性 b_i でグループ化し 各グループ毎に集約関数 f を a_i に適用する処理，w はその集約値である．つまり，${}_{b_i}G_{v=f(a_i)}$ は部分データ S に集約 · グループ化処理を実行する OLAP クエリであり，その結果はグループ化の値とその集約値を組としたシーケンス型であり，次式で表現できる．

$$q(S) = [(V_1, W_1), (V_2, W_2), \cdots, (V_N, W_N)] \tag{11}$$

但し，$V_i (1 \leqq i \leqq N)$ はグループ化対象の次元属性 b_i が持つ各属性値，$W_i (1 \leqq i \leqq N)$ は各集約値である．N は取りうるグループ化属性の数を表している．

2.3.1 大域例外データの特定

ユーザが単一の集約属性 a_i と次元属性 b_i，次元属性を指定し，有益な上位 k 件のデータを探索する具体的な分析例として，大域例外部分データの探索タスクが存在する [14]．このタスクでは，任意の条件から得られる全ての部分データと全体データの分析結果の乖離度を数値化する必要がある．そこで，有用性が高い分析結果を生み出す部分データとは，全体データの分析結果との乖離が大きいと仮定し，部分データ集合 $\mathbb{S}$ から乖離度の大きい上位 k 件の部分データ S を求める．この探索タスクでは，部分データ集合と単一の集約属性，グループ化属性を事前にユーザが指定する．複数クエリ結果間の乖離度を数値化する関数として，本問題では乖離度の数値化関数としてユークリッド距離を採用する．評価関数として式 (12) を使用すると，上位 k 件の部分データを特定する大域例外データの探索タスクは以下のように定義される．

定義 1. 部分データ集合 $\mathbb{S}$，OLAP クエリ q，特定する大域例外部分データの件数 k を指定し，部分データ集合 $\mathbb{S}$ に属する全ての S の中で $\mathbb{D}(q'(S))$ 上位 k 件の S を特定する．

$$\arg\max_{S \in \mathbb{S}}^{k} \mathbb{D}(q(S))$$

ただし，$\mathbb{D}(q(S))$ は以下の式で表される．

$$\mathbb{D}(q(S)) = |q(S) - q(D)|,\ S \in \mathbb{S}$$

3. 提案フレームワーク

本章では，複数クエリの同時実行と上位 k 件検索による探索空間の削減手法を両立したフレームワークを説明する．このフレームワークは， Apache Spark[注1] 上に UDAF (User Defined Aggregation Function) を用いて実装した．複数クエリを同時実行する方法としてデータを逐次で処理する方法を採用する．そして，上位 k 件の枝刈りに関しては，Online Aggregation の技術を適用する．具体的には統計的信頼区間の技術 [15] [12] [13] [16] を利用することで，サンプルデータをもとに全体データのクエリ結果を推定する．推定結果を用いて，上位 k 件の候補に成り得ないデータキューブ を枝刈りする．上位 k 件の枝刈り判定は分析アルゴリズムによって異なるため，枝刈り判定自体は分析アルゴリズムが行い，フレームワーク側はその判定に従いデータキューブ の更新をスキップする．3.1 節でフレームワークの動作について，3.4 節では本研究で用いた分析アルゴリズムについて説明する．

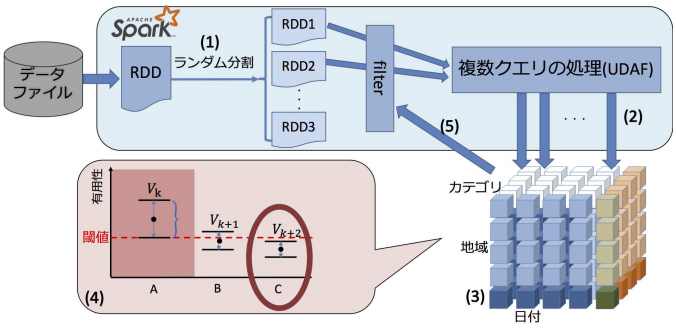

図 2　フレームワークの概要

3.1 フレームワークの概要

探索的データ解析では，様々なパターンで集約グループ化処理を実行するため，OLAP クエリの実行回数が膨大になる．そのためデータを読み込みながらそのレコードが対象となる複数 OLAP クエリを同時に実行するストリーム処理を行い，その結果を用いてデータキューブを作成 · 更新する．これによりデータを 1 スキャンする事で複数の OLAP クエリの結果を同時に作成することができる．また，有用性の高い分析結果上位 k 件を分析者に対し提示することが分析アプリケーションの目的であるため，統計的信頼区間推定技術を適用し上位 k 件の候補になり得ない OLAP クエリを早期に特定し，その OLAP クエリに関する処理を枝刈りすることで高速化を図る．

提案するフレームワークの処理の流れは以下の通りである．図 2 は実装したフレームワークの概要を表しており， 図の番号と処理の流れの番号が対応している．

（ 1） 統計的信頼区間の技術を適用するためにはデータのランダムサンプリングが必要である．ランダムサンプリングを疑似的に行うため，入力データを複数[注2] に分割する．

（ 2） 分割したデータ集合ごとに，レコードを逐次的に読む込むことで複数クエリを同時実行し，データキューブを差分更新する．

（ 3） データキューブを構成するベースキューボイド に対して統計的信頼区間推定技術を適用し，クエリ結果の上限値と下限値を推定する．

（ 4） 推定したクエリ結果を用いて有用性の上限値と下限値を計算し，上位 k 件に成り得ないクエリを特定する．

（ 5） 上位 k 件に成り得ないクエリを枝刈りを行う．

（ 6） 上位 k 件が確定するか，分割したデータ集合を全て読み終わるまで 2 ～ 5 を繰り返す．

（ 注1）: https://spark.apache.org/

（ 注2）: 分割個数はパラメータで指定する

3.1.1 データ構造

フレームワークではデータキューブの形式で OLAP のクエリ結果の内部データを保持する．全データを保持していると計算途中のデータサイズが膨大になるため，クエリ結果の区間推定 (式 (2.2)) を行うのに必要な標本数と標本平均，標本分散のみをデータキューブで保持する．

保持するデータキューブの最小単位であるベースキューボイドはデータキューブを構成するすべての属性に対し 1 つの要素を条件として持つ．そしてキューボイドとはデータキューブの部分集合であり，あるキューボイドは OLAP クエリの結果を保持している．具体的には，データキューブを構成する属性が (都道府県, カテゴリ, 月) であり，集約関数 op，集約属性 $price$ とすると，OLAP クエリ $q_0 = SELEC\ op(price)\ FROM\ TABLE\ WHERE$ 都道府県 = 東京都 AND カテゴリ $= A\ AND$ 月 $= 12$ はベースキューボイドの 1 つを表わしている．

3.2 データキューブの更新

標本数と標本平均，標本分散に関して，データブロックごとに計算して以下のように差分更新する．データブロック A, B のそれぞれの要素数を n_a, n_b，標本平均を $mean_a, mean_b$，標本分散を var_a, var_b とすると，データブロック $A \cup B$ の要素数 n_{ab} と平均 $mean_{ab}$，分散 var_{ab} は以下のように計算する事ができる．この性質を用いることで中心極限定理の計算に必要な統計情報を差分更新する．

$$n_{ab} = n_a + n_b \tag{12}$$

$$mean_{ab} = \frac{mean_a n_a + mean_b n_b}{n_{ab}} \tag{13}$$

$$var_{ab} = \frac{n_a(var_a + mean_a^2) + n_b(var_b + mean_b^2)}{n_{ab}} - mean_{ab}^2 \tag{14}$$

集約値の信頼区間の計算を行う時に，中心極限定理の式 (7) を適用する．具体的には，データキューブが保持しているデータ (標本数 N, 標本平均 $Average$, 標本分散 Var) から信頼区間は以下のように表される．但し，$\bar{Y}$ が推定したい集約値である．

$$Y\_interval = [\bar{Y} - \epsilon, \bar{Y} + \epsilon], \qquad \epsilon = z_p\sqrt{\frac{Var}{N-1}} \tag{15}$$

3.3 フレームワークのアルゴリズム

Algorithm1 を用いて，本システムのフレームワーク部の動作の詳細を説明する．探索する件数 k，信頼係数 z_p，データセットを分割する割合 $x\_array = [x_0\%, x_1\%, \cdots, x_l\%]$，データキューブを作成する際に使用する属性集合 $Dimension$，使用するデータセット $DataSet$ を入力とし，上位 k 件の部分データを出力とする．入力データセットをユーザが事前に指定した割合 x_array のレコードを保持する l 個のデータ集合に分割する (1 行目)．分割されたデータセット毎に枝刈りされた部分データに該当するレコードをフィルタリングする (6 行目)(注3)．

Algorithm 1: フレームワークの動作

Input : k, z_p, x_array, Dimension, DataSet
Output: Results

```
1  DataRDDs ← RandomSampling(Dataset, x_array)
2  CUBE ← ∅
3  Subset_key ← ∅
4  foreach records ∈ DataRDDs do
5      if Subset_key.size > k or Cube is empty then
6          records.filterling(Subsetkey)
7          Cube ++ = QueryExecute(records)
8          foreach (subsetkey, value) ∈ Cube do
9              Cube1[subsetkey].Upper = value + ϵ
10             Cube1[subsetkey].Lower = value - ϵ
11         end
12         Results, Subset_key ← ComputeUtility(Cube1)
13         foreach (subset,value) ∈ Cube do
14             if subset ∉ Subset_key then
15                 Cube.Remove(subset)
16             end
17         end
18     end
19 end
```

入力となる各レコードごとに UDAF を適用することで複数クエリを同時に処理してその結果を用いてデータキューブを差分更新する (7 行目)．データキューブのすべてのキューボイドに対して上限値と下限値を計算する (9, 10 行目)．上限値と下限値を保持したデータキューブを用いて，有用性の計算 · 枝刈り判定を行う (12 行目)．有用性の計算 · 枝刈り判定は分析アプリケーションが決定する (詳細は 3.4 節で述べる)．枝刈り判定の結果をもとにデータキューブから有用性の低いクエリ部分の削除を行う (13 行目から 16 行目)．

3.4 大域例外データの分析例

本研究で使用する分析アプリケーションとして 2.3.1 で述べた大域例外部分データの特定の例を用いて有用性の計算方法を説明する．また集約関数は AVG か SUM に限定して議論を進める．大半の部分データの分析傾向は, 全体データの分析結果の傾向と類似しているため大域例外データの上位 k 件の候補に成り得ないため処理の途中において枝刈りすることが望ましい．そのため, 統計的信頼区間推定技術を適用し上位 k 件の候補になり得ない部分データを早期に特定し, その部分データの探索を枝刈りすることで高速化を図る．各標本に対し統計的信頼区間の技術を適用したクエリ結果を用いて，全体データからの乖離度の上限値と下限値を推定し, 上位 k 件の候補になり得ない部分データの枝刈りを行う．

データキューブから大域例外部分データを特定する部分データとグループ化属性の粒度のキューボイドを作成し，全体データに関する OLAP クエリ結果のデータキューボイドを複数のベースキューボイドを合算し作成する．部分データ集合 $\mathbb{S}$, OLAP クエリ q，特定する大域例外部分データの件数 k，閾値

(注3)：但し，最初のデータセットでは枝刈り条件が無いためフィルタリングを行わない．

Algorithm 2: 大域例外データの特定部分

Input : k, DataCube // k: 探索する件数，DataCube: クエリ結果の信頼区間を保持するデータキューブ

Output: Subset_key

```
1  DevianceArray ← ∅ // 乖離度を保持する変数
2  All ← All_GroupByAggregate(DataCube)
3  SubsetCube ← Subset_GroupByAggregate(DataCube)
4  foreach (key, XY) ∈ SubsetCube do
5      deviance ← (0, 0) //乖離度の信頼区間を導出
6      foreach (x, y_interval) ∈ XY do
7          deviance.Upper + = Large( CalucuDistance( All[x],
           y_interval ) )
8          deviance.Lower + = Small( CalucuDistance( All[x],
           y_interval ) )
9      end
10     DevianceArray[key] ← deviance
11 end
12 threshold ← GetTop_k(DevianceArray, k)
13 Subset_key ← {}
14 foreach (key, deviance) ∈ DevianceArray do
15     if threshold ← ≦ deviance.Upper then
16         Subset_key + = {key}
17     end
18 end
```

は乖離度の上限値が k 番目に大きい部分データの乖離度の下限値 $threshold = Top\_k(\mathbb{D}(q(S)).lower)$ とすると，上位 k 件に成り得る部分データの集合 $N_k(S)$ を以下のように定義する．

$$N_k(S) := \{S \in \mathbb{S} \mid \mathbb{D}(q(S)).upper \geqq threshold\} \quad (16)$$

Algorithm2 を用いて探索空間の削減手法の詳細について説明する．Algorithm2 では，データキューブ $DataCube$，探索する部分データの件数 k を入力 (Input) として，上位 k 件に成り得る部分データ名を出力 (Output) する．入力として与えられるデータキューブの値は既に統計的信頼区間推定の技術を適用したクエリ結果の上限値と下限値を保持している状態である(注4)．まず，データキューブから全体データに関するクエリ結果を保持するデータキューブを作成し (2 行目)，同様に部分データに関するデータキューブを使用する粒度のデータキューブを作成する (3 行目)．部分データ毎に乖離度の上限値と下限値を計算する (4 行目から 11 行目)．ここで全体データ，部分データ共に集約結果に上限値と下限値を保持しているためそれぞれの集約結果区間を用い，各グループ化属性の値での乖離度の最大値と最小値を算出する．そして，乖離度の上限値が k 番目に大きい部分データの乖離度の下限値を閾値に設定する (12 行目)．上位 k 件の候補集合を探索し (14 行目から 18 行目)，候補集合 $N_k(S)$ と乖離度を出力する．

(注4)：部分データに関して，グループ化属性値の中に集約対象のレコードが存在しない場合，そのグループ化属性値の集約値を 0 とすることで，グループ化属性値の数を統一する．

4. Spark での実装

データキューブの更新で最もコストが大きいのは入力データに対するベースキューボイドの特定処理である．これは，入力データとデータキューブの結合演算と等価であることから，従来のリレーショナルデータベースの結合演算の最適化技術を適用し，データキューブをハッシュテーブルとしたハッシュ結合を行う．step.2 では UDAF を用いてデータキューブとレコードをハッシュ結合して更新するべきキューボイドを特定してから，キューボイドを差分更新する．フレームワークの step.1 では，統計的信頼区間を推定することで枝刈りを行うため入力データ D をランダムの順にレコードを格納することでデータを分割すること (これをランダム分割と呼ぶ) が必要である．入力データを RDD (Resilient Distributed Dataset) で読み込み，RDD の機能である RandomSplit を使用することでランダム分割を行う．そして，分割された RDD 毎にフレームワークの step.2 から step.5 を行う．枝刈りされていない $q(S), S \in \mathbb{S}$ に合致しないレコードをスキップするため，枝刈り情報を基に RDD に filter メソッドを行う．UDAF 内ではデータを永続化することやレコード以外を入力とすることが出来ないため，UDAF 外で分割された RDD に対する step.2 の処理結果を保持しておく必要がある．step.2 終了時に分割された RDD 毎に作成されたデータキューブを UDAF 外のデータキューブにハッシュ結合して全データ D に関するキューブを維持する．フレームワークの step.5 では，有用性の区間から枝刈りされる部分データを特定し，枝刈り情報 (fliter) を更新する．

5. 評価実験

本章では提案したエンジンにおいて複数クエリの同時実行と探索空間削減手法を両立したフレームワークの有効性を評価した．

5.1 実験環境

提案フレームワークの有効性を検証するために，複数クエリの同時実行のみを行った場合と複数クエリの同時実行と枝刈りを両立した場合の実行時間と分析の出力結果の精度を計測する実験を実データを用いて行った．本稿で使用した実データは，経営科学系研究部会連合協議会主催平成 29 年度データ解析コンペティションで提供されたデータである．また，分析結果の精度の指標として，ランキングの評価指標の一つである nDCG (normalized Discounted Cumulative Gain) を採用した．

本実験には，CPU が Intel(R) Core(TM) $i5-6600$, クロック周波数は 3.30GHz, コア数は 4，メモリは 16GB の PC を使用する．

5.2 実験結果

OLAP クエリ: 使用したクエリは，部分データ集合のためのディメンション属性を **月**，集約関数を AVG，集約属性を **会計税込売上**，グループ化属性を **(店 and 誕生年代)** とした．

パラメータ: 分析者の入力パラメータは，サンプリングにおいて 95% 信頼区間を用い (すなわち，Z_p は 1.96 となる)，データの分割割合を 10%のデータ集合 10 個とした．

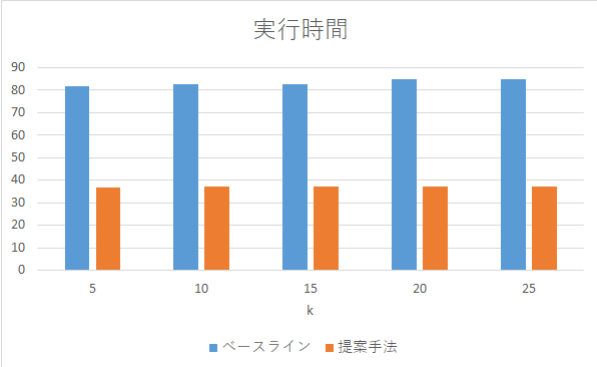

図 3　高速化性能の結果

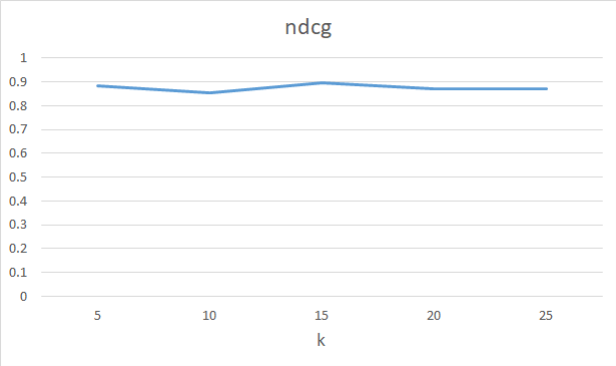

図 4　分析精度の結果

図 3 は，複数クエリの同時実行のみを適用した手法 (**ベースライン**) と複数クエリの同時実行と探索空間の削減技術を両立した手法 (**提案フレームワーク**) に関して，探索する部分データの件数 k を $k = [5, 10, 15, 20, 25]$ と変化させ全体の実行時間を計測した結果である．最大 43.5%の実行時間の削減が確認できた．また，探索する件数を増加させたとしても高速化ができていることが図 3 からわかる．図 4 は，**ベースライン**手法と**提案フレームワーク**の探索する部分データの件数 k を変化させ分析 (出力) 結果の精度を計測した結果である．探索件数を変化させても高精度の分析結果を出力できていることがわかる．精度が少し低下しているのは，枝刈りした部分データに関するレコード分だけ全体データのクエリ結果が正確でない事が原因であると考えられる．また，データ件数の少ない部分データが誤差への影響が大きいと考えられる．

6. 関 連 研 究

6.1 分析自動化ツール

データ分析を自動化する研究が行われている [5] [4] [9]. SEEDB [2] は，分析クエリを実行した際の分析結果と全体平均の乖離が最大となる分析パターンを自動で探索するシステムである．ユーザは着目したい部分データを指定することで全体平均と異なる分析結果を得ることができる．また, SEEDB では仮説指向計画法を導入している．これにより , 着目した部分データの階層構造を考慮した分析が可能になっている．Zenvisage [9] は，複数クエリの共有化プランの最適化を主とした高速化を図る分析自動化ツールである．EKI [5] は, ユーザからの入力を必要としない複数の分析に関して全部分空間から上位 k 件の部分データ, 分析種類 · データの抽出方法を自動で探索するシステムである．トレンド分析と全体平均からの例外値分析のような異なる分析同士の比較するために, 分析種類を基に帰無仮説を立て, その仮説の下で実際にデータから計算された統計量よりも極端な統計量が観測される確率を使用した定式化関数を提案している．また, 集約関数として SUM, AVG, COUNT, MIN, MAX だけでなくランキングを求める関数や次元属性が順序を持つ場合に増減値 (Δ_{prev}) 求める関数を実装することでより複雑な分析を可能にしている．そして, これらの集約関数を複数回組み合わせデータを抽出する機構を作成している．これにより ,「ブランド毎の売上金額の前年比増減の順位」を求めることができる．全部分空間の探索を行うため複数の最適化技術を使用している．MuVE [17] は, 有用な分析結果を数的な次元でのグループ化処理を可能にするための評価関数を用いて有用性の定量化を行い, 有用性の大きい OLAP クエリを特定する．これらの研究では地域性や時期性などの局所性を考慮できていないのに対し , 小笠原らの研究 [6] は, LOF (Local Outlier Factor) 値を用いることで局所的な例外部分データを特定する．

6.2 データ分析ツール

データ分析ツールには, Tableau, Spotfire, Google Fusion Tables などが存在する．Tableau は, データ分析に関する専門的な知識を理解していないユーザを対象にした商業用のデータ分析可視化ツールである．ドラッグアンドドロップ操作などのマウスジェスチャーのみで様々な可視化を行うことができる．Tableau は, データソースの結合や新たなデータ列の作成も行うことが可能である．また, データセットに最適な可視化設定を自動的に選択するが, ユーザがグラフの種類や軸の種類 · 可視化位置などを変更できるため, 可視化したい分析結果を表示することが可能である．さらに, 複数の可視化を一つに統合し出力できる．Spotfire は, 散布図をベースとしたインタラクティブな可視化システムである．Google Fusion Tables は, 複数のデータソースからデータの収集 · 結合 · 可視化の処理を自動化する技術の一つである．分析者が必要としている公開されている web 上のデータを簡単に発見できるシステムを提供することが目的である．

将来的な OLAP クエリ結果可視化ツールである Ermacs [18] は，データベースの最適化機能とデータベースの可視化機能を統合するための新しい宣言的な可視化言語を有している．また，データの事前読み込みを用いた多次元データキューブ 分析に関する研究 [19] [20] では，OLAP Cube から特異的な単一のセルを探索する手法を提案している．

6.3 データ分析の高速化技術

データ分析の高速化は 2 つの手法に大別することができる．まず一つ目は, 事前計算を取り入れることである．あらかじめ指定されたデータキューブ [19] などのフォーマットに処理計算を行っておくことで, 対話的なクエリの応答を可能にするものである．しかし , この手法は巨大な Cube を必要とするため高次元データに対応していないことが問題である．また, データの追加や修正などが行われると再計算に時間を要することも問

題である．

もう一方は, データのサンプリングに基づく高速化手法である．サンプリングは, 計算が高速で柔軟性があるので高次元データにも対応可能である．しかし, 計算結果の正確性と実行時間は二律背反である．サンプリングを用いた OLAP クエリ結果の区間推定に関する研究 [12] [13] や推定結果を用いた分析の研究 [6] [8] [21] などが行われている．

7. ま と め

本研究では, 探索的データ解析におけるエンジンの効率化を目的としたストリームエンジンを作成した．エンジンのフレームワークは複数クエリを同時実行する技術と有用性の上位 n 件に成り得ない探索パターンを処理の途中で足切りする技術を両立することで高速化する．評価実験の結果，提案フレームワークは，高精度の分析結果の出力を可能でかつ高速化性能が確認できた．今後の課題として，LOF の発見 [6] や分析パターンの探索 [3] などの分析アプリケーションの拡充がある．また，枝刈り率により高速化性能が大きく変動する問題点が存在する．

8. 謝　　辞

本研究は JSPS 科研費 JP16K00154 の助成を受けたものです．

文　　献

[1] S. Idreos, O. Papaemmanouil, and S. Chaudhuri, “Overview of Data Exploration Techniques,” Proceedings of the ACM SIGMOD, 2015.

[2] A. Parameswaran, N. Polyzotis, and H. Garcia-Molina, “SeeDB: Visualizing Database Queries Efficientl,” Proceedings of the VLDB Endowment, vol.7, no.4, pp.325–328, 2013.

[3] M. Vartak, and S. Madden, “SEEDB : Automatically Generating Query Visualizations,” Proceedings of the 40th International Conference on Very Large Data Bases, vol.7, no.13, pp.1581–1584, 2014.

[4] M. Vartak, S. Rahman, S. Madden, A. Parameswaran, and N. Polyzotis, “SEEDB : Efficient Data-Driven Visualization Recommendations to Support Visual Analytics,” Proceedings of the VLDB Endowment, 2015.

[5] B. Tang, S. Han, M. Lung, Y. Rui, and D. Dongmei, “Extracting Top-K Insights from Multi-dimensional Data,” Proceedings of the ACM SIGMOD, 2017.

[6] 小笠原麻斗，水野陽平，佐々木勇和，鬼塚真，“局所例外部分データの自動探索,” Proceedings of the DEIM，2017.

[7] N. Kamat, P. Jayachandran, K. Tunga, and A. Nandi, “Distributed and interactive cube exploration,” Proceedings of the International Conference on Data Engineering, pp.472–483, 2014.

[8] M. Yohei, S. Yuya, and O. Makoto, “Efficient Data Slice Search for View Detection,” Proceedings of the DOLAP, 2017.

[9] T. Siddiqui, A. Kim, J. Lee, K. Karahalios, and A. Parameswaran, “Effortless Data Exploration with zenvisage: An Expressive and Interactive Visual Analytics System,” Proceedings of the VLDB Endowment, 2016.

[10] D. Suciu, M. Fern, A. Morishima, M. Fern, and D. Suciu, “SilkRoute Efficient Construction of Materialized XML Views with SilkRoute,” Proceeding of the Information Processing Society of Japan, pp.421–428, 2001.

[11] B. Mozafari, “Approximate Query Engines: Commercial Challenges and Research Opportunities,” Proceedings of the ACM SIGMOD, 2017.

[12] J.M. Hellerstein, P.J. Haas, and H.J. Wang, “Online aggregation,” Proceedings of the ACM SIGMOD, 1997.

[13] P.J. Haas, and S. Jose, “Large-Sample and Deterministic Confidence Intervals for Online Aggregation,” Proceedings of the SSDBM, 1997.

[14] 水野陽平，鬼塚真，“統計的信頼区間を用いた特徴的な部分データの効率的探索,” Proceedings of the DEIM，2016.

[15] J.F. Naughton, “Technical Perspective: Optimized Wandering for Online Aggregation,” Proceedings of ACM SIGMOD, vol.46, 2017.

[16] F. Li, B. Wu, K. Yi, and Z. Zhao, “Wander Join: Online Aggregation via Random Walks,” Proceedings of the ACM SIGMOD, 2016.

[17] H. Ehsan, M.A. Sharaf, and P.K. Chrysanthis, “MuVE: Efficient Multi-Objective View Recommendation for Visual Data Exploration,” Proceedings of the ICDE, 2016.

[18] E. Wu, L. Battle, and S.R. Madden, “The case for data visualization management systems,” Proceedings of the VLDB Endowment, no.10, 2014.

[19] S. Sarawagi, R. Agrawal, N. Megiddo, G.V.A.V. Univ Politecn Valencia, and E.T.H.Z.O.S.S.I. Edbt Fdn, “Discovery-driven exploration of OLAP data cubes,” Proceedings of the EDBT, 1998.

[20] S. Sarawagi, “User-adaptive exploration of multidimensional data,” Proceeding of the VLDB, 2000.

[21] K. Zeng, S. Agarwal, and I. Stoica, “iOLAP: Managing Uncertainty for Efficient Incremental OLAP,” Proceedings of the 2016 International Conference on Management of Data, pp.1347–1361, 2016.