情報処理学会論文誌 Vol.58 No.2 481–494 (Feb. 2017)

モバイルセンサデータベースにおける効率的なTop-k検索結果の多様化手法

横山 正浩[1,a)] 原 隆浩[1,b)]

受付日 2016年5月1日, 採録日 2016年11月1日

概要：近年のセンサ技術の発展にともない，モバイルセンサ端末が普及しており，これらの端末から収集されたモバイルセンサデータを活用する研究に注目が集まっている．モバイルセンサデータは，環境属性値を有する位置情報付きデータであり，たとえば周辺の環境属性値に比べて極端な値を持つデータを取得することで，ホットスポットの地理的分布に関する知識が得られる．このような応用に対して，地理空間上の Top-k 検索結果の多様化が有効である．しかし，データに対するスコアはユーザごとに異なり，クエリごとに計算しなければならない．また，大量のモバイルセンサデータに対して単純な手法を用いると，計算コストがきわめて大きくなってしまう．本論文では，事前にモバイルセンサデータに対しクラスタリング処理を施すことによる，効率的な Top-k 検索結果の多様化手法を提案する．単純な手法では，検索結果として最適なデータを探索するために，検索範囲内のすべてのデータを走査する必要があるが，提案手法ではクラスタリング情報に基づき走査するデータの数を削減することで，高速に検索結果が得られる．評価の結果，提案手法を用いることで，計算時間を短縮でき，かつディスク IO コストを削減できることを確認した．

キーワード：モバイルセンサデータ，ホットスポット検出，参加型センシング

An Efficient Top-k Result Diversification Method for Mobile Sensor Database

Masahiro Yokoyama[1,a)] Takahiro Hara[1,b)]

Received: May 1, 2016, Accepted: November 1, 2016

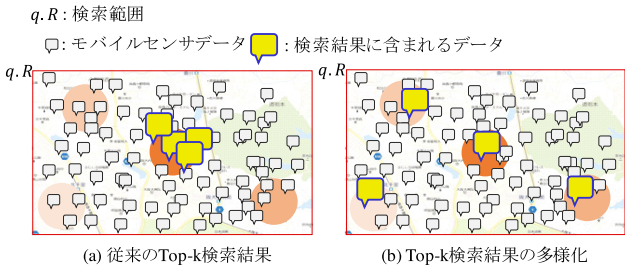
Abstract: Due to recent developments in sensor technologies, mobile sensor device use has become widespread, and many researchers have been attempting to leverage data collected by these devices. Mobile sensor data are geo-referenced data with environmental attribute values; and they enable us to determine the geographical distribution of hot spots by retrieving data with comparatively extreme environmental attribute values. Top-k result diversification in geographical space is valid for applications of this sort. However, the preference scores for data items are different from each user's interest, and must be calculated for each query. In this case, the computational cost of a naive method is excessively high when the amount of mobile sensor data is very large. In this paper, we propose an efficient top-k result diversification method for mobile sensor data. In a naive method, it is necessary to scan all data existing in a given query range when seeking the best data. Our proposed method, however, can reduce the amount of scanned data by exploiting cluster information, and the query result can thereby be returned much more rapidly. Experimental results show that our proposed method involves short computation time and reduces the disk IO cost.

Keywords: mobile sensor data, hot spot detection, participatory sensing

[1] 大阪大学大学院情報科学研究科
Graduate School of Information Science and Technology, Osaka University, Suita, Osaka 565–0871, Japan
[a)] yokoyama.masahiro@ist.osaka-u.ac.jp
[b)] hara@ist.osaka-u.ac.jp

1. はじめに

近年，スマートフォンをはじめとして，様々なセンサデバイスを搭載したモバイル端末が広く普及している．この



(a) 従来のTop-k検索結果　(b) Top-k検索結果の多様化

図 1 ホットスポットと検索結果
Fig. 1 Hot spots and query results.

ようなモバイル端末によって，端末保持者付近の物理現象や環境の変化を観測しデータ収集を行う手法は，参加型センシング [6], [17] と呼ばれ，現在注目が集まっている．このようにして収集されたデータは**モバイルセンサデータ**と呼ばれ，多くの研究者によってモバイルセンサデータを利活用する研究が行われている．モバイルセンサデータは，騒音や大気汚染指数といった，観測した現象に関する環境属性値を持った位置情報付きデータである．モバイル端末は，実世界のセンシング情報を継続的に生成するため，莫大なデータがモバイルセンサデータベースに蓄積される．

現在，位置情報付きデータに対する検索について，検索範囲を指定し範囲内のデータを取得する，時空間範囲検索が主流である．しかし，一般的にユーザはモバイルセンサデータの時空間分布を事前に知りえないため，単純な時空間範囲検索では，検索結果にユーザにとって必要のないデータが非常に多く含まれてしまう．ユーザは多くの場合，町中において高い気温や大気汚染指数を示すデータといった，極端なデータに興味があると考えられる．検索範囲内に多くのデータが存在する場合，分析の目的に応じたユーザの注目する環境属性（気温，湿度，騒音指数，大気汚染指数など）によりデータにスコアを割り当て順位付けし，上位 k 個のデータを取得する Top-k 検索が有効である [9], [18]．Top-k 検索によって，ユーザはそのときの分析目的に応じた，有用であると考えられるデータを取得できる．しかし，一般的に環境情報は，空間的に近くに位置するデータは互いに似た属性値をとる確率が高いという，空間的自己相関と呼ばれる特徴を有する [19]．そのため，Top-k 検索結果に含まれる大部分のデータは，**図 1** (a) に示すように，ある特定の領域から得られたデータである可能性が高い．ここでは，各円形領域は高いスコアを示すデータが分布する領域であり，ホットスポットであるとする．ホットスポットにおけるデータは，その周辺のデータと比較して高いスコアを示すため，Top-k 検索結果はこれらすべてのホットスポットから得られる可能性がある．しかし，$k = 4$ とした場合，上位 4 個のデータは単一のホットスポットからしか取得できない．

図 1 (b) に示すように，ホットスポットにおいて生成されたデータをより効率的に取得するためには，Top-k 検索結果の多様化 [7], [10] が有効である．この処理は，ユーザの検索基準に基づいたデータのスコアが高く，かつ検索範囲内でより分散するようなデータの組合せを取得することを目的としている．したがって，このような検索はモバイルセンサデータベースにおいてホットスポット検出に応用できる．

ここで，モバイルセンサデータに対して Top-k 検索結果の多様化を適用する際の課題について述べる．一般的に検索結果の多様化は，検索結果として選択されるデータの関連度および多様性の両者を用いて目的関数がモデル化され，目的関数を最大化することによって解かれる．既存の多様化の枠組みをモバイルセンサデータベースに適用する場合，データの関連度はデータのスコアに，多様性はデータ間の空間距離に相当する．Top-k 検索とは異なり，空間距離の計算のためにデータ間の比較が必要であるため，多様化における目的関数は個々のデータから計算できない．既存の多様化手法は，検索範囲内のすべてのデータと，すでに検索結果として選択されているデータを比較し，最適なデータを繰り返し選択することで最終的な検索結果を算出する．また，多くの手法はデータのスコアがすでに与えられている状態を仮定している．これらの既存手法をモバイルセンサデータの検索に適用した場合，最適なデータを取得するためにデータセット全体を走査する必要があるが，データセットサイズがきわめて大きい場合，計算コストも比例して大きくなる．

そこで本論文では，モバイルセンサデータベースにおける効率的な Top-k 検索結果の多様化手法を提案する．著者らが知る限り，モバイルセンサデータベースにおける検索結果の多様化に関する課題に取り組んでいる研究は，これまでに存在しない．提案手法は，少数のデータにのみアクセスすることで計算時間を短縮し，かつデータのスコアおよびデータ間の空間距離に関して最適な検索結果を保証することを目的とする．提案手法は，オフライン事前クラスタリング処理とオンラインクエリ処理からなる．オフライン事前クラスタリング処理では，空間位置が近く，かつ環境属性値が似ているデータどうしをクラスタ化し，クラスタ内の特定のデータから中心データおよび代表データを 1 つずつ選択する．これらのクラスタは構造化してファイルに格納され，オンラインクエリ処理時に利用される．オンラインクエリ処理では，各クラスタの中心データおよび代表データのみを走査し，中心データのスコアとクラスタ半径の情報から，クラスタ内のデータがとりうる評価値の上界を推定する．空間位置および環境属性値の両面から近接性を考慮してクラスタリングすることで，クラスタ内のデータがとりうる評価値の上界をできる限り小さく，短時間で推定できる．これにより，推定された評価値が十分に小さいクラスタ内のデータを走査対象から除外することで，最適なデータを短時間で探索できる．結果として，走査する

データ数を大幅に削減しつつ，単純な手法における検索結果と同一の検索結果を取得できる．また，提案手法は検索結果の多様化に関する種々の最適化問題に対しても同様に適用可能である．シミュレーション実験の結果から，提案手法は単純な手法に比べて計算時間およびディスク IO コストを大幅に削減できることを確認した．

以下では，2 章で想定環境を紹介し，本論文の問題を定義する．3 章で単純なグリーディアルゴリズムによるベースライン手法を紹介し，4 章で提案手法について説明する．5 章でシミュレーション実験の結果を示し，6 章で関連研究について述べる．最後に，7 章で本論文をまとめる．

2. 問題定義

モバイルセンサ端末は，周期的に付近の大気汚染指数，気温，湿度などの物理現象についてセンシングするものとする．ユーザの検索クエリを q，検索範囲を $q.R$ としたとき，検索範囲内に分布するデータ集合を O で表す．$q.R$ は Top-k 検索結果の多様化処理を行う時空間範囲であり，図 1 (a) および (b) における矩形領域が該当する．以降では，$q.R$ の範囲外に存在するデータは無視し，データ集合 O のみを検索対象とする．データ $o \in O$ は，データ ID $o.id$，観測時刻 $o.t$，位置情報 $o.\boldsymbol{loc}$，環境属性値 $o.\boldsymbol{att}$ を保持している．$o.\boldsymbol{loc}$ は，経度 $o.loc_x$ と緯度 $o.loc_y$ によって表される 2 次元平面内の点とし，$o.\boldsymbol{att}$ は m 次元のベクトル $o.att_i$（$i = 1, ..., m$）で表される．たとえば，モバイルセンサ端末が時刻 t_n に位置 $\boldsymbol{loc}$ の環境属性値 $\boldsymbol{att}$ をセンシングしデータ ID が id のデータを生成した後，別の位置 $\boldsymbol{loc}'$ に移動し時刻 t_{n+1} にその位置の環境属性値 $\boldsymbol{att}'$ をセンシングしデータ ID が id' の別のデータを生成したとする．このとき，データベースには 2 つのタプル $(id, t_n, \boldsymbol{loc}, \boldsymbol{att})$ および $(id', t_{n+1}, \boldsymbol{loc}', \boldsymbol{att}')$ が格納され，各カラムの値は不変で更新は行われない．

各データのスコアは，クエリ q に基づいて決定される．ユーザはクエリ q に対して，各環境属性に対する興味の度合いを示す重み付け係数 $q.\boldsymbol{w}$ を付与する．クエリ q における，データ o のスコア $p(q, o)$ は，以下の式に従って計算される．

$$p(q, o) = \sum_{i=1}^{m} q.w_i \cdot o.att_i \tag{1}$$

式 (1) 中の w_i は i 番目の環境属性に対する重みを示す．高いスコアを示すデータは，検索結果に含まれる可能性が高い．以降では文脈上明らかな場合は，$p(q, o)$ を $p(o)$ のように略記する．

2 つのデータ間の多様性は $d : O \times O \to R^+$ によって表され，完全に一致する場合には値は 0 となる．ここでは，単純に d を空間距離とし，$d(u, v)$ はデータ u，v の位置情報から算出される u，v 間のユークリッド距離であり，以下の式に従って計算される．

$$d(u, v) = \sqrt{(u.loc_x - v.loc_x)^2 + (u.loc_y - v.loc_y)^2}. \tag{2}$$

上述した環境属性値から算出されるデータのスコア，およびデータ間の位置情報から算出される空間距離に基づいて，モバイルセンサデータベースにおける Top-k 検索結果の多様化を以下のように定義する．

定義. クエリ $q = \{R, k, \lambda, \boldsymbol{w}\}$ が与えられたとき，データ集合 O を検索範囲内で観測されたデータ集合 $O = \{o_i \mid o_i \in q.R\}$ とする．このとき，以下の式で与えられる最適化問題を解くことによって，最適な検索結果 S_k^* が得られる．

$$S_k^* = \underset{S_k \subseteq O, |S_k| = k}{\arg\max} f(S_k, q, p(\cdot), d(\cdot, \cdot)). \tag{3}$$

ここで，$f(S_k, q, p(\cdot), d(\cdot, \cdot))$ は目的関数である．以降では文脈上明らかな場合は，$f(S_k, q, p(\cdot), d(\cdot, \cdot))$ を $f(S_k)$ のように略記する．Top-k 検索結果の多様化を達成するために，これまで様々な最適化問題が提案されている．たとえば，Maxmin [8]，Maxsum [3]，Maximal Marginal Relevance（MMR）[2], [7] の目的関数は，それぞれ以下の各式で示される．

$$f_{min}(S) = \min_{u \in S} p(u) + \lambda \min_{u,v \in S} d(u, v) \tag{4}$$

$$f_{sum}(S) = (k-1) \sum_{u \in S} p(u) + 2\lambda \sum_{u,v \in S} d(u, v) \tag{5}$$

$$f_{mmr}(S) = (1-\lambda) \sum_{u \in S} p(u) + \lambda \min_{u,v \in S} d(u, v). \tag{6}$$

いずれの目的関数についても，正解集合 S 内のデータのスコアが大きいほど目的関数の値は大きくなり，また，正解集合 S 内の任意のデータ間の距離が大きいほど目的関数の値は大きくなる．目的関数中の λ は，ユーザの検索における地理的多様性についての重要性を表しており，λ が大きいほど地理的多様性を重視してデータを要求し，地理的により分散した結果が得られる．特に，$\lambda = 0$ のときはデータの地理的多様性を完全に無視し，データのスコアのみが考慮されるため，最終的な正解集合は純粋な Top-k 検索結果と等しくなる．

上記の組合せ最適化問題を解くことは，NP 困難であることが示されており，検索範囲内のデータセットサイズ N が大きいときにすべての部分集合候補について総当たりで探索するのは，計算時間の観点から現実的ではない．そこで，本論文が対象とするモバイルセンサデータベースのような大規模なデータセットにおける最適化問題の解を得るために，何らかのヒューリスティックな手法を用いる必要がある．様々なヒューリスティックな手法の中でも，グリーディアルゴリズムは得られる正解集合の質および計算時間の観点から効果的であることが知られており，様々な

目的関数に応じた手法が提案されている [3]．そこで，本論文においてもグリーディアルゴリズムをベースラインとする．

3. ベースライン手法

ベースラインとなる単純なグリーディアルゴリズムを，Algorithm 1 に示す．文献 [3], [7] において，正解集合の初期化処理は，得られる検索結果の質に対して大きな影響を与えないことが確認されている．そこで，1，2 行目の初期化処理は文献 [7] に従い，データセット内で最大のスコアをとるデータを正解集合に追加することとした．3 行目から 6 行目の反復により，正解集合の大きさが k となるまで，繰り返しデータを正解集合に追加する．4 行目の $d'(\cdot, S)$ は，データのスコアとデータ間の空間距離から算出される評価値である．評価値は，それぞれの最適化問題について以下のように定義される．

$$d'_{min}(y, S) = \min_{u \in S} \left\{ \frac{1}{2}(p(y) + p(u)) + \lambda d(y, u) \right\} \quad (7)$$

$$d'_{sum}(y, S) = \sum_{u \in S} \{p(y) + p(u) + 2\lambda d(y, u)\} \quad (8)$$

$$d'_{mmr}(y, S) = \min_{u \in S} \{(1 - \lambda)p(y) + \lambda d(y, u)\}. \quad (9)$$

以下の式に示すように，4 行目で追加されるデータ y^* は，評価値 $d'(\cdot, S)$ を最大化すると同時に，目的関数 $f(S \cup y)$ を最大化する．

$$y^* = \arg\max_{y \in O \setminus S} d'(y, S) = \arg\max_{y \in O \setminus S} f(S \cup y). \quad (10)$$

このアルゴリズムの計算量は，データセットサイズ N に依存する．初期化処理は，データのスコアが最大のデータを探索するため，単純にデータセット全体の走査が必要となり，計算量は $O(N)$ である．また，3 行目から 6 行目の反復については，反復回数が k，各反復につき最大 $k(N-k)$ 回の評価値の計算が必要となるため，全体の計算量は $O(k^2N)$ となる．そのため，データセットサイズが大きくなると計算時間が長くなってしまう．そこで本論文では，効率的な Top-k 検索結果の多様化手法，すなわち走査するデータ数を削減し，かつベースライン手法と同一の正解集合を取得する手法を提案する．

Algorithm 1 Algorithm for the Optimization Problems

Input: Data set $O, k, \lambda, \boldsymbol{w}$

Output: Set $S(|S| = k)$ that maximizes $f(S)$

1: Initialize the set $S = \emptyset$
2: Find $x^* = \arg\max_{x \in O} p(x)$ and set $S = \{x^*\}$
3: **while** $|S| < k$ **do**
4: Find $y^* \in O \setminus S$ such that $y^* = \arg\max_{y \in O \setminus S} d'(y, S)$
5: Set $S = S \cup \{y^*\}$
6: **end while**

4. 提案手法

本章では，本論文の提案手法について説明する．ベースライン手法では，アルゴリズム中の各反復で最適なデータを探索するために，正解集合に含まれない全データ $O \setminus S$ を走査する必要があり，計算時間が長くなる．この際の計算コストを削減するために，提案手法では環境属性値に関する空間的自己相関と呼ばれる特徴 [19] を利用する．この特徴を考慮すると，空間的に近くに存在するセンサデータは，互いに似た環境属性値を有する可能性が高い．そしてそのようなデータは，ユーザの環境属性値に対する関心が異なる，すなわちスコアリング関数の重み付け係数が異なる場合でも，互いに似たスコアをとる．また，空間的に近くに存在するため，正解集合内のデータとの空間距離も近い値となる．よって，空間的に近いデータは，評価値も互いに似た値となる可能性が高い．

以上を考慮し，オフライン事前クラスタリング処理では，空間的に近いデータをクラスタ化し，クラスタ中心のデータをそのクラスタの中心データおよび初期代表データとする．オンラインクエリ処理では，最初に全クラスタの中心データと代表データについてのみ，評価値を計算する．この際に走査するデータの数は，全体のデータセットサイズ N に比べて大幅に少ない．クラスタ内の他のデータの評価値について，その上界は中心データの評価値およびクラスタ半径から推定できる．このとき，中心データの評価値が十分に小さい場合，そのクラスタ内のデータは正解集合に追加されえないと判断でき，走査するデータ数を削減できる．

まず，4.1 節において，オンラインクエリ処理で利用するクラスタを作成するためのオフライン事前クラスタリング処理，作成されたクラスタの管理方法，およびクラスタのメンテナンス方法について説明する．次に，4.2 節において，作成されたクラスタファイルセットを利用したオンラインクエリ処理について説明する．

4.1 オフライン事前クラスタリング処理

具体的なクラスタリングアルゴリズムを，Algorithm 2 に示す．3，4 行目で，いずれのクラスタにも属していないデータを見つけた場合，そのデータを新たなクラスタの中心データかつ代表データとする．ここで，5 行目の retrieveNeighbors(o_i, r_1, r_2) は，データ o_i に空間位置が互いに近く，環境属性値が互いに似ているデータを返す操作である．具体的には，データ o_i の空間位置ベクトル $o_i.\boldsymbol{loc}$ を中心とした半径 r_1 の円内に存在し，かつ，データ o_i の環境属性値ベクトル $o_i.\boldsymbol{att}$ を中心とした半径 r_2 の超球内に存在するデータを返す操作である．m 次元の環境属性値について，半径 r_2 の超球内に存在するデータ集合 O_i は，ユークリッド距離を用いた以下の式で表される．

Algorithm 2 Algorithm for Clustering

Input: Data set O, r_1, r_2

Output: Set of clusters $\boldsymbol{C} = C_1, C_2, ..., C_k$

```
clusterLabel = 1
for i = 1 to N do
  if o_i is not in any clusters then
    Mark o_i as the center and initial representative of the current cluster
    X = retrieveNeighbors(o_i, r_1, r_2)
    for j = 1 to |X| do
      if o_{i,j} is not in any clusters then
        Mark o_{i,j} with current clusterLabel
      end if
    end for
    clusterLabel++
  end if
end for
```

$$O_i = \left\{ o \mid \sqrt{\sum_{j=1}^{m} (o_i.att_j - o.att_j)^2} \leq r_2 \right\}. \qquad (11)$$

retrieveNeighbors(o_i, r_1, r_2) で取得したデータのうち，いずれのクラスタにも属していないデータに対し，現在作成中のクラスタのラベルを付与する．クラスタ間でのデータの共有はないものとし，すべてのデータがいずれかのクラスタに割り当てられるまでクラスタを生成する．

ここで，空間位置だけでなく，環境属性値についてもデータの近接性を考慮するのは，以下の理由による．センサデータは，互いに空間距離が近くても，観測した属性値に誤差を含んでいる場合や，観測時刻が他のデータと離れている場合がある．そのようなデータは，空間位置は近いが，環境属性値に差が生じることでスコアが大きく異なり，互いに評価値も大きく異なる可能性がある．そこで，環境属性値に関して近接性を考慮することで，このようなデータを互いに別々のクラスタに分割できる．

このアルゴリズムで最も計算量の大きい操作は，クラスタメンバ候補を取得する retrieveNeighbors(o_i, r_1, r_2) である．ここでは，データセットサイズが N で，データが構造化されていない場合を考える．retrieveNeighbors(o_i, r_1, r_2) では，o_i とそれ以外のデータ間の空間位置に基づく距離および環境属性値に基づく距離を計算する必要があり，合計 $2(N-1)$ 回の距離計算が行われる．ここで最悪の場合は，すべてのデータが別々のクラスタに分離されてしまう場合である．このとき retrieveNeighbors(o_i, r_1, r_2) はすべてのデータ $o_i \in O$ に対して実行されるため，N 回実行される．よって，このアルゴリズムの最悪計算量は $O(N^2)$ であり，クエリ到着前にオフラインで実行されることから，あらかじめオフライン事前クラスタリング処理を実行するための時間は十分確保できると考えられる．また，retrieveNeighbors(o_i, r_1, r_2) によるクラスタメンバ候補の取得は，R*木 [1] のような多次元インデックス構造を用いた範囲検索により，効率化できる．

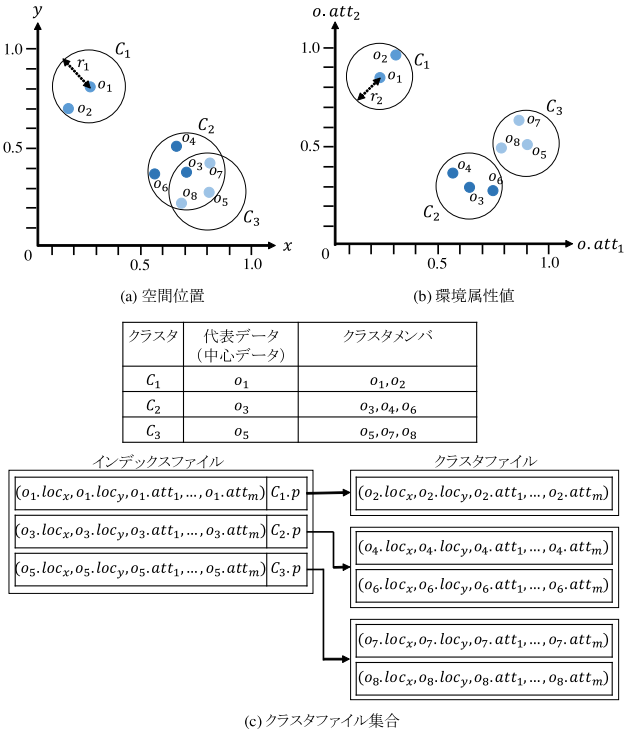


図 2 クラスタリング例

Fig. 2 An example of clustering.

4.1.1 クラスタリング例

図 2 (a) および (b) に示す具体例を用いて，環境属性の次元数 $m = 2$ の場合の，オフライン事前クラスタリング処理について説明する．まず，データ o_1 を中心としてクラスタ C_1 を生成する．データ o_2 はデータ o_1 を中心とした空間距離 r_1 の円内に存在し，かつデータ o_1 を中心とした環境属性値空間で半径 r_2 の円内に存在する．よって，データ o_2 は，データ o_1 と同じクラスタに割り当てる．次に，データ o_3 を中心としてクラスタ C_2 を生成する．残りのクラスタ未割当てのすべてのデータは，データ o_3 を中心とした空間距離 r_1 の円内に存在する．しかし，環境属性値空間では，データ o_3 を中心とした半径 r_2 の円内に存在するのはデータ o_4 とデータ o_6 の 2 つのみである．よって，クラスタ未割当てのデータのうち，データ o_4 とデータ o_6 のみをクラスタ C_2 に割り当てる．同様に，データ o_5 を中心としてクラスタ C_3 を生成すると，残りのデータ o_7 およびデータ o_8 はクラスタ C_3 に割り当てられる．結果，すべてのデータはそれぞれ 3 つのクラスタのうちのいずれかに割り当てられる．

4.1.2 クラスタのファイル管理方法

ここで，作成されたクラスタのファイル管理方法について説明する．提案するオンラインクエリ処理アルゴリズム

に適するようにクラスタファイルを作成することで，アルゴリズム全体におけるディスク IO コストを削減できる．

オンラインクエリ処理では，まず最初にクラスタの中心データおよび代表データのみ走査される．詳細は後述するように，クラスタの代表データの評価値が十分に小さい場合，そのクラスタは最適なデータを含みえない．そのため，中心データと代表データを除き，そのようなクラスタのデータはディスクから読み込む必要はない．一方，初期状態では中心データと代表データは一致しているため，ディスクから読み込むデータは各クラスタにつき 1 つのみとなる．

構造化されたクラスタは，図 2 (c) に示されるようなインデックスファイルと，関連するクラスタ数分のクラスタファイルとして保存される．インデックスファイルは，各クラスタの中心データ（代表データ）からなる．インデックスファイルの各レコードは，代表データの位置情報，環境属性値，そして当該クラスタファイルへのポインタからなる．これらのポインタを用いることで，最適なデータを含んでいると考えられるクラスタのクラスタファイルを読み込み，走査できる．

既存手法は，データのスコアを計算するために，検索範囲内のデータをすべてファイルから読み込まなければならない．しかし，提案手法では，上述したファイル管理方法を用いることで，大部分のクラスタファイルをディスクから読み込むことなく正解集合が得られる．さらに，検索対象データセットが一度構造化されれば，たとえユーザごとに環境属性に関する興味が異なり，クエリのスコアリング関数が変化したとしてもつねに効率的に対応できる．また，クエリの時間範囲内のデータが構造化されていれば，どのようなクエリも提案するオンラインクエリ手法が適用できる．

4.1.3 クラスタのメンテナンス方法

オフライン事前クラスタリング処理の後にも新たにデータは生成されるが，これらのデータを既存のクラスタに組み込むことで，クラスタのメンテナンスが可能である．ここで，新たに生成されたデータを o_{new} とする．クラスタのメンテナンスアルゴリズムを，Algorithm 3 に示す．4.1 節で述べたとおり，中心データの空間位置ベクトルを中心とした半径 r_1 の円内に存在し，かつ，中心データの環境属性値ベクトルを中心とした半径 r_2 の超球内に存在するデータをクラスタメンバとして，各クラスタは構成されている．よって，既存のクラスタの中から，o_{new} がクラスタメンバとして適切なクラスタを探索すればよい．このようなクラスタ集合は，既存のクラスタの中心データ集合を対象として，Algorithm 2 で用いた retrieveNeighbors(o_{new}, r_1, r_2) を実行することで取得できる（1 行目）．取得したクラスタ集合 $\boldsymbol{C}^*$ 内のクラスタは，いずれも o_{new} をクラスタメンバとすることができるため，集合内でランダムに選ばれたクラスタ C^* のクラスタファイルに o_{new} を挿入する（3，4 行目）．o_{new} がクラスタメンバとして適切なクラスタが存在しない場合は，o_{new} を新たなクラスタの中心データかつ代表データとし，インデックスファイルに挿入する（6 行目）．

Algorithm 3 Algorithm for Maintenance of Clusters

Input: Set of clusters $\boldsymbol{C}$, r_1, r_2, o_{new}

1: $\boldsymbol{C}^*$ = retrieveNeighbors(o_{new}, r_1, r_2)
2: **if** $\boldsymbol{C}^*$ is not NULL **then**
3: C^* = random($\boldsymbol{C}^*$)
4: Mark o_{new} with C^*.clusterLabel
5: **else**
6: Mark o_{new} as the center and initial representative of the new cluster
7: **end if**

表 1 4.2 節で用いる記号のリスト
Table 1 A list of symbols used in Section 4.2.

記号	意味
$o_{i,rep}$	クラスタ C_i の代表データ
$o_{i,cen}$	クラスタ C_i の中心データ
r_1	クラスタの空間半径
r_2	クラスタの環境属性値半径
$\boldsymbol{C'}$	走査対象クラスタ集合
$\overline{d'(C_i, S)}$	クラスタ C_i の上界
v_i	クラスタ C_i 内の仮想データ

一般的に，環境属性値の分布が大きく変化しなければ，すでに環境属性値が似たデータが存在するため，新たに生成されたデータは既存のクラスタに組み込まれる．環境属性値の分布が変化したとしても，次第にその分布に応じた新しいクラスタが生成されるため，以降に生成されるデータはそれらの新しいクラスタに組み込まれる．

この処理では，クラスタ数が M のとき，M 個の中心データとの空間位置に基づく距離および環境属性値に基づく距離を計算する必要があり，合計で $2M$ 回の距離計算が行われる．この処理の計算量は $O(M)$ であり，さらにクラスタ数 M はデータセットサイズ N に比べて小さいため，新たに生成されたデータによるクラスタのメンテナンスは短時間で行うことができる．

4.2 クラスタを利用したオンラインクエリ処理

次に，クラスタを利用したオンラインクエリ処理について説明する．具体的なアルゴリズムを Algorithm 4 に示し，**表 1** に本節で用いる記号をまとめる．ここでは特に，走査データ数削減の要点である，Algorithm 4 の反復部分について説明する．まず，全クラスタの代表データを走査し，評価値の最大値をとる代表データ o^*_{rep} を探索する（4

Algorithm 4 Algorithm for Optimization Problems Leveraging Clusters

Input: $\boldsymbol{C}, k, \lambda, \boldsymbol{w}, r_1, r_2$
Output: Set $S(|S| = k)$ that maximizes $f(S)$

1: Initialize the set $S = \emptyset$
2: Find $x^* = \arg\max_{x \in O} p(x)$ and set $S = \{x^*\}$
3: **while** $|S| < k$ **do**
4: Find o^*_{rep} such that $o^*_{rep} = \arg\max_{o_{i,rep} \in C_i} d'(o_{i,rep}, S)$
5: Initialize the set $\boldsymbol{C'} = \{C \mid o^*_{rep} \in C\}$
6: **for all** $i = 1$ to $|\boldsymbol{C}|$ **do**
7: Estimate upper bound of each cluster $\overline{d'(C_i, S)} = \max_{v_i \in C_i} d'(v_i, S)$
8: **if** $d'(o^*_{rep}, S) \leq \overline{d'(C_i, S)}$ **then**
9: $\boldsymbol{C'} = \boldsymbol{C'} \cup \{C_i\}$
10: **end if**
11: **end for**
12: Find $y^* \in \boldsymbol{C'} \backslash S$ such that $y^* = \arg\max_{y \in \boldsymbol{C'} \backslash S} d'(y, S)$
13: Set $S = S \cup \{y^*\}$
14: **if** y^* is representative data of C_i **then**
15: Select new representative data for C_i
16: **end if**
17: **end while**

行目）．またこのときに，中心データの評価値も計算し，記憶しておく．これは，本節の後半で説明するように，クラスタ内データがとりうる評価値の上界を推定する際に必要となるためである．評価値が最大の代表データ o^*_{rep} を含むクラスタは，正解集合に追加する最適なデータを含む可能性が高い．よって，このクラスタを走査対象クラスタ集合 $\boldsymbol{C'}$ に追加する．次に，全クラスタについて，各クラスタ内のデータの評価値がとりうる値の上界 $\overline{d'(C_i, S)}$ を推定する（7行目）．各クラスタの上界の推定値と，最初に計算した代表データの最大評価値 $d'(o^*_{rep}, S)$ を比較し，推定値のほうが大きい場合，$\boldsymbol{C'}$ に追加する．ここで，以下の定理が示すとおり，$\boldsymbol{C'}$ に含まれないデータは最適なデータとはならないため，走査する必要はない．

定理. 最適なデータは走査対象クラスタ集合内に存在する．すなわち，$\arg\max_{o \in O} d'(o, S) = \arg\max_{o \in \boldsymbol{C'}} d'(o, S)$ である．

証明. 背理法により証明する．最適なデータは走査対象クラスタ集合外 $O \backslash \boldsymbol{C'}$ に存在すると仮定する．最適なデータを o^* とし，データ o^* を含むクラスタを C^* とする．データ o^* は正解集合に追加するデータとして最適であるため，その評価値は少なくとも代表データの最大評価値以上となる．ゆえに，$d'(o^*, S) \geq d'(o^*_{rep}, S)$ である．ここで，クラスタ C^* の評価値の上界について，明らかに $\overline{d'(C^*, S)} \geq d'(o^*, S)$ となる．これらの不等式および Algorithm 4 の 8 行目から 10 行目より，クラスタ C^* は走査対象クラスタ集合 $\boldsymbol{C'}$ に追加される．すなわち，$C^* \in \boldsymbol{C'}$ である．これは，最適なデータ o^* が走査対象クラスタ集合外 $O \backslash \boldsymbol{C'}$ に存在するという仮定に矛盾する． □

最後に，走査対象クラスタ集合 $\boldsymbol{C'}$ に含まれるすべてのデータを走査し，最大の評価値をとるデータを最適なデータとして，正解集合 S に追加する．追加されたデータがいずれかのクラスタの代表データであった場合，クラスタ内のデータからランダムに新たな代表データを選択する（15行目）．

4.2.1 クラスタの上界の推定

ここで，Algorithm 4 の 7 行目における，各クラスタ内データがとりうる評価値の上界の推定方法について説明する．クラスタが含むデータの分布の詳細は不明なため，クラスタ内に存在しうる仮想的なデータ v_i を考え，データ v_i がとりうる最大の評価値を，可能な限り正確に計算する．評価値は，データ間の空間距離と，環境属性値に基づくスコアの 2 つの指標から算出される．ここで，それぞれの最適化問題における評価値を，正解集合内のデータに非依存の項と依存する項に分解する．

$$d'_{min}(v_i, S) = \frac{1}{2} p(v_i) + \min_{u \in S} \left\{ \frac{1}{2} p(u) + \lambda d(v_i, u) \right\} \quad (12)$$

$$d'_{sum}(v_i, S) = |S| p(v_i) + \sum_{u \in S} \{ p(u) + 2\lambda d(v_i, u) \} \quad (13)$$

$$d'_{mmr}(v_i, S) = (1 - \lambda) p(v_i) + \min_{u \in S} \{ \lambda d(v_i, u) \} \quad (14)$$

まず，正解集合内のデータに非依存の項（第 1 項）がとりうる最大値を計算する．スコアは，重みベクトル $q.\boldsymbol{w}$ と環境属性値ベクトル $o.\boldsymbol{att}$ の内積としてとらえると，それぞれのベクトルのなす角を θ としたとき，以下の式で計算できる．

$$\begin{aligned} p(q, o) &= \sum_{i=1}^{m} q.w_i \cdot o.att_i \\ &= q.\boldsymbol{w} \cdot o.\boldsymbol{att} \\ &= |q.\boldsymbol{w}||o.\boldsymbol{att}| cos\theta. \end{aligned} \quad (15)$$

よって，クラスタ内に存在しうる仮想データ $v_i (v_i.\boldsymbol{att} = o_{i,cen}.\boldsymbol{att} + \boldsymbol{\epsilon})$ を考えたとき，第 1 項の最大値は以下の式で与えられる（**図 3** (a)）．

$$\begin{aligned} &\max_{v_i \in C_i} (p(v_i)) \\ &= \max_{|\boldsymbol{\epsilon}| \leq r_2, 0 \leq \theta \leq 2\pi} ((q.\boldsymbol{w} \cdot (o_{i,cen}.\boldsymbol{att} + \boldsymbol{\epsilon}))) \\ &= (p(o_{i,cen}) + \max_{|\boldsymbol{\epsilon}| \leq r_2, 0 \leq \theta \leq 2\pi} (|q.\boldsymbol{w}||\boldsymbol{\epsilon}| cos\theta)) \\ &= p(o_{i,cen}) + |q.\boldsymbol{w}| r_2. \end{aligned} \quad (16)$$

次に，正解集合内のデータに依存する項（第 2 項）がとりうる最大値を計算する．Maxmin 問題および MMR 問題

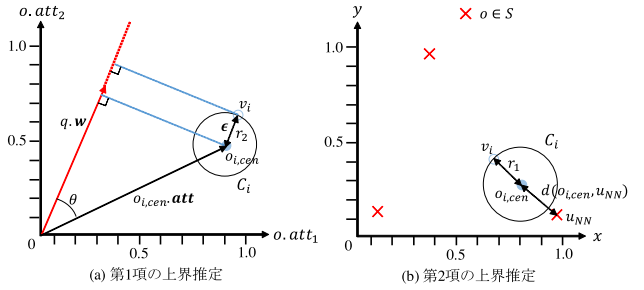


図 3 クラスタの上界の推定

Fig. 3 Estimation of upper bounds of clusters.

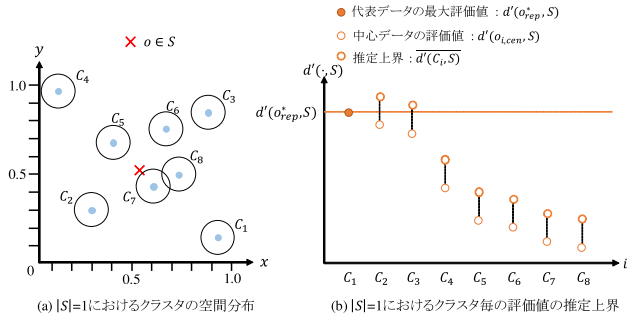


図 4 オンラインクエリ処理例

Fig. 4 An example of online query processing.

それぞれについて，クラスタの中心データと正解集合内のデータとの，各データのスコアを加味した距離を計算し，そのときの正解集合内のデータを $u_{NN} \in S$ とする．中心データとデータ u_{NN} を直線で結んだとき，2 つの交点が存在する（図 3 (b)）．ここで，データ u_{NN} から最も離れる位置は，2 つの交点のうち，データ u_{NN} から遠い方の点である．仮想データ v_i がこの点に位置するとき，正解集合 S からの距離も最大化される．そのため，Maxmin 問題および MMR 問題における第 2 項がとりうる最大値は，以下の式で与えられる．

$$\max_{v_i \in C_i} \left\{ \min_{u \in S} \left(\frac{1}{2} p(u) + \lambda d(v_i, u) \right) \right\}$$
$$= \frac{1}{2} p(u_{NN}) + \lambda \{ d(o_{i,cen}, u_{NN}) + r_1 \} \quad (17)$$

$$\max_{v_i \in C_i} \{ \min_{u \in S} (\lambda d(v_i, u)) \}$$
$$= \lambda \{ d(o_{i,cen}, u_{NN}) + r_1 \}. \quad (18)$$

また，Maxsum 問題において，仮想データ v_i の位置として，正解集合内のそれぞれのデータ $u \in S$ から最も離れた点を仮定する．もちろん，このような，仮想データが複数の位置情報を有する仮定は成り立ちえないが，この場合に計算される仮想データの評価値は，明らかに上界となる．よって，Maxsum 問題における第 2 項がとりうる最大値は，以下の式で与えられる．

$$\max_{v_i \in C_i} \left\{ \sum_{u \in S} (p(u) + 2\lambda d(v_i, u)) \right\}$$
$$= \sum_{u \in S} \{ p(u) + 2\lambda (d(o_{i,cen}, u) + r_1) \}. \quad (19)$$

これらの式から，それぞれの最適化問題におけるクラスタの上界を計算でき，以下の式で示される．

$$\overline{d'_{min}(C_i, S)}$$
$$= \frac{1}{2} \{ p(o_{i,cen}) + |q.\boldsymbol{w}| r_2 \}$$
$$+ \frac{1}{2} p(u_{NN}) + \lambda \{ d(o_{i,cen}, u_{NN}) + r_1 \}$$
$$= \left\{ \frac{1}{2} (p(o_{i,cen}) + p(u_{NN})) + \lambda d(o_{i,cen}, u_{NN}) \right\}$$
$$+ \frac{1}{2} |q.\boldsymbol{w}| r_2 + \lambda r_1$$
$$= d'(o_{i,cen}, S) + \frac{1}{2} |q.\boldsymbol{w}| r_2 + \lambda r_1 \quad (20)$$

$$\overline{d'_{sum}(C_i, S)}$$
$$= |S| \{ p(o_{i,cen}) + |q.\boldsymbol{w}| r_2 \}$$
$$+ \sum_{u \in S} \{ p(u) + 2\lambda (d(o_{i,cen}, u) + r_1) \}$$
$$= \left\{ |S| p(o_{i,cen}) + \sum_{u \in S} (p(u) + 2\lambda d(o_{i,cen}, u)) \right\}$$
$$+ |S| |q.\boldsymbol{w}| r_2 + 2|S| \lambda r_1$$
$$= d'(o_{i,cen}, S) + |S| (|q.\boldsymbol{w}| r_2 + 2\lambda r_1) \quad (21)$$

$$\overline{d'_{mmr}(C_i, S)}$$
$$= (1 - \lambda) \{ p(o_{i,cen}) + |q.\boldsymbol{w}| r_2 \}$$
$$+ \lambda \{ d(o_{i,cen}, u_{NN}) + r_1 \}$$
$$= \{ (1 - \lambda) p(o_{i,cen}) + \lambda d(o_{i,cen}, u_{NN}) \}$$
$$+ (1 - \lambda) |q.\boldsymbol{w}| r_2 + \lambda r_1$$
$$= d'(o_{i,cen}, S) + (1 - \lambda) |q.\boldsymbol{w}| r_2 + \lambda r_1. \quad (22)$$

これらの式より，上界の推定はクラスタの中心データの評価値と，クラスタ半径およびクエリ情報のみで簡単に計算できる．特に，中心データの評価値はアルゴリズムの上界推定の前に計算し記憶されている．1，2 行目の初期化処理についても，式 (16) からクラスタ内データのとりうるスコアの上界が推定できるため，同様の手順で走査データ数を削減できる．

4.2.2 オンラインクエリ処理例

図 4 を用いて，走査データ数を削減する方法を例示する．図 4 (a) は，初期化処理が完了し，$|S| = 1$ となっている状態である．評価値が最大の代表データ o^*_{rep} は探索済みで，かつ各クラスタの中心データの評価値は計算済みとする．ここでは簡単のために，評価値最大の代表データ o^*_{rep} を含むクラスタを C_1，それ以外のクラスタ ID を中心データの評価値の降順に割り当てている．次に，各クラスタの中心データの評価値から，クラスタの上界を推定する．

図 4 (b) に，計算されたそれぞれの値の分布を示す．このとき，クラスタ $C_4 \sim C_8$ は，推定された上界がデータ o^*_{rep}

の評価値を下回っているため，走査対象クラスタ集合 C' に追加されない．よって，クラスタ C_4〜C_8 内のデータは走査する必要はない．

5. 評価実験

Top-k 検索結果の多様化処理における，提案手法の性能を評価する．**表 2** は各パラメータの値を示し，太字はデフォルト値とする．

オフライン事前クラスタリング処理は，クエリ到着前に一度だけ実行されればよく，データ公開前に十分な時間が確保できると考えられる．一方，オンラインクエリ処理は複数のユーザからクエリを受け取るたびに繰り返し実行されるため，高速化が求められる．そこで本章における評価では，オンラインクエリ処理にともなう計算時間およびディスク IO コストを求めた．

5.1 データセット

データの位置情報を，各次元の値が区間 $[0, 1]$ 上の一様分布に従う，2 次元ベクトルで与えた．また，データの環境属性値は，**図 5** に示すような空間的自己相関の特徴を有する分布に従う値とし，1 次元から 4 次元まで設定した．図中の各矩形領域がクエリの検索範囲 $q.R$ であり，横軸および縦軸がそれぞれデータ位置の x 座標，y 座標を表す．具体的な環境属性値は，データの位置情報から決定される．また，センシング時の誤差を考慮して，位置情報から決定される環境属性値に対し，$N(0, 0.3)$ の正規分布に従う正規乱数を加算した．

5.2 比較手法

提案手法（以降のグラフ中では ‘P-cluster’ と表記）を，3 章で説明したベースライン手法（‘Naive’）および空間位置の近接性のみを考慮して作成されたクラスタを利用する手法（‘C-cluster’）と比較した．この比較手法は，クエリ処理のアルゴリズムは提案手法と同じだが，環境属性値空間における半径が $r_2 = \infty$ であり，式 (16) においてクラスタ内のデータのスコアについて上界を推定するための情報を持たない．そのため，最初にすべてのデータをディスクから読み込みすべてのデータのスコアを計算し，クラスタごとに最大のスコアとそのデータを記憶する．以降，評価値の上界推定にはこの最大のスコアを用いることで，他の手法と同一の正解集合が取得できる．

表 2 パラメータの値
Table 2 Parameter values.

パラメータ	値
データセットサイズ N	**1 M**, 5 M, 10 M, 50 M
要求データ数 k	5, 10, **15**, 20, 25, 30
クラスタの空間半径 r_1	0.01〜0.15
クラスタの環境属性値半径 r_2	0.1〜2.0
$\boldsymbol{w}$ の各要素	0.0〜1.0
λ（Maxmin, Maxsum）	0.0〜5.0
λ（MMR）	0.0〜1.0
環境属性の次元数 m	1, **2**, 3, 4

5.3 設定

すべてのアルゴリズムを Java7 で実装し，Intel(R) Core(TM) i7-4790K CPU @ 4.00 GHz with 24.0 GB RAM を搭載する Windows7 Enterprise で動作する計算機上で実験した．

実験においては，オンラインクエリ処理でセンサデータおよびクラスタデータを RAM に読み込んだ時点から，検索結果を取得するまでの計算時間を測定した．また，RAM に読み込んだデータ数としてディスク IO コストを示した．ここで，提案手法以外の手法は，データのスコアを計算するためにすべてのデータを RAM に読み込む必要がある．そのため，これらの手法（以降のディスク IO コストを示したグラフ中では ‘Others’ と表記）におけるディスク IO コストは，つねにデータセットサイズ N に等しくなる．さらに，比較手法と提案手法について，それぞれのオフライン事前クラスタリング処理によって生成されたクラスタ数を示した．

実験で用いたクエリは，$q.\boldsymbol{w}$ と $q.\lambda$ がそれぞれ表 2 に示す一定範囲内でランダムに設定されたものである．文献 [9], [18] に従い，$q.\boldsymbol{w}$ の各要素の値の範囲は 0.0〜1.0，かつ各要素の和が 1.0 となるように設定した．Maxmin 問題および Maxsum 問題における $q.\lambda$ については，文献 [8] に従い最小値を 0.0 とした．一方で最大値は，Maxmin 問題および Maxsum 問題における目的関数（式 (4), (5)）では，

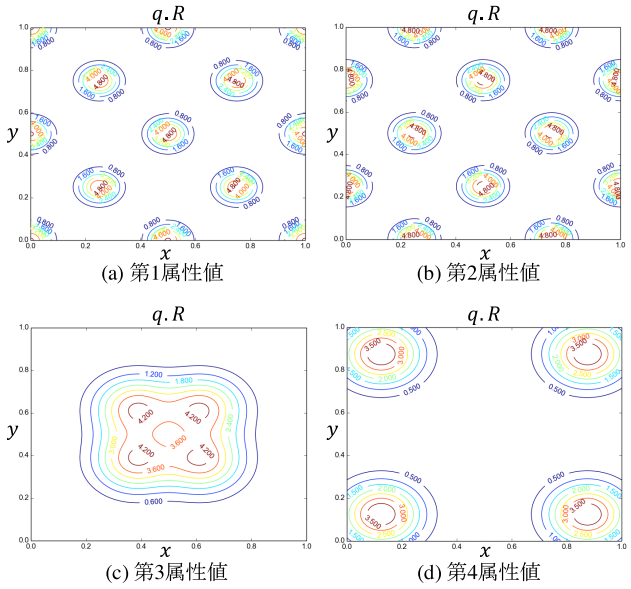
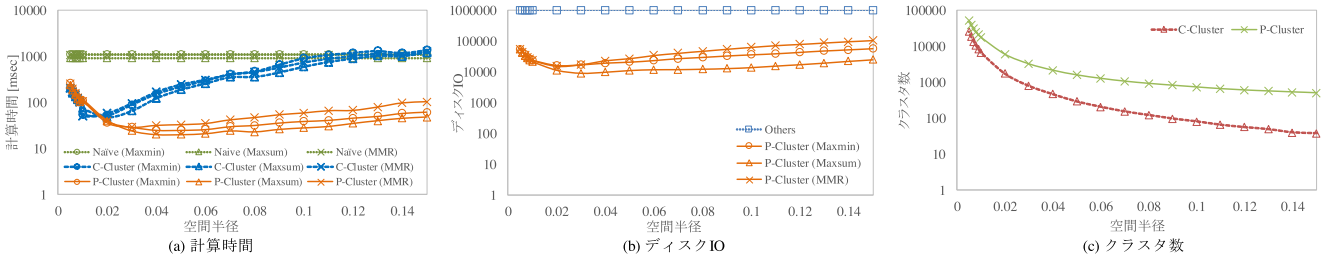


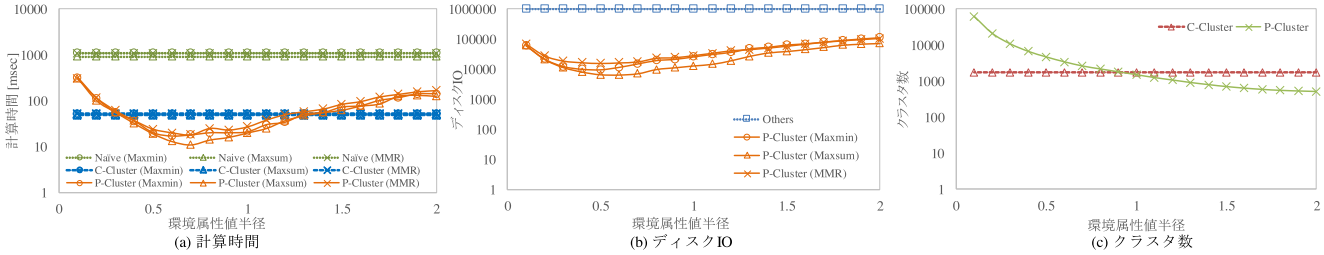
図 5 環境属性値分布
Fig. 5 Distributions of environmental attribute values.



(a) 計算時間 (b) ディスクIO (c) クラスタ数

図 6 空間半径 r_1 の影響

Fig. 6 Impact of spatial radius r_1.



(a) 計算時間 (b) ディスクIO (c) クラスタ数

図 7 環境属性値半径 r_2 の影響

Fig. 7 Impact of environmental attribute radius r_2.

データ間の空間距離の項に対してのみ λ が乗じられていることを考慮して，独自に設定した．上述した $q.\boldsymbol{w}$ の設定のもとで，データのスコアは図 5 に示す各ホットスポットの中心で最大 5.0 程度の値をとる．一方で，データ間の空間距離の最大値は検索範囲の対角線の距離である $\sqrt{2} \approx 1.4$ である．今回想定するクエリは，ホットスポット検出を目的とした Top-k 検索結果の多様化であり，評価においては目的関数でデータのスコアと地理的多様性が少なくとも同程度に重要視される必要がある．よって，Maxmin 問題および Maxsum 問題における $q.\lambda$ の最大値を 5.0 とした．一方，MMR 問題における $q.\lambda$ の範囲は，文献 [2], [7] に従い 0.0〜1.0 とした．

作成されたランダムな 100 個のクエリを処理した際の，計算時間とディスク IO コストの平均値を調べた．なお，比較手法と提案手法では，100 個のクエリの処理にそれぞれ同一のクラスタファイルセットを用いた．そのため，クラスタ数はクエリと無関係であり，用いたクラスタファイルセットのクラスタ数をそのまま示している．

5.4 空間半径 r_1 の影響

クラスタの空間半径 r_1 を変化させた場合の，計算時間を**図 6** (a) に，ディスク IO コストを図 6 (b) に，クラスタ数を図 6 (c) に示す．図 6 (a) から，空間半径が大きい場合，比較・提案手法ともに計算時間が長くなっていることが分かる．これは，クラスタ内データが広い地理範囲に存在しうるため，クラスタ内データの評価値の上界を過大に推定しており，走査対象から除外できたクラスタ数が少ないためである．このことは，属性値が急激に変化している領域におけるクラスタで顕著である．特に，$0.1 \leq r_1 \leq 0.15$ の場合の比較手法は，クラスタの中心データおよび代表データの評価値計算が余分に必要なため，ベースライン手法よりも計算時間がわずかに長くなっている．一方，空間半径が小さい場合も，比較・提案手法ともに計算時間が長くなっていることが分かる．これは，図 6 (c) に示されるように，空間半径が小さくなると生成されるクラスタ数が増加することによる．走査対象クラスタ数を削減できても，クラスタの上界を計算するためにすべてのクラスタの中心データを走査しなければならず，結果として全体の計算時間は長くなってしまう場合がある．

提案手法における最短の計算時間は，すべての最適化問題に関して，比較手法より短くなっている．また，さらに注目すべき点として，提案手法はディスク IO コストを最高 100 分の 1 に削減できている点があげられる．比較手法では，中心データから空間半径 r_1 の円内のデータはすべて同一のクラスタに割り当ててしまうため，センシング誤差により周辺のデータの環境属性値と大きく異なるデータが含まれる場合がある．その結果，クラスタ内の他のデータに比べ非常に高いスコアを有するデータがクラスタに含まれる場合，そのようなデータによりクラスタの上界が引き上げられてしまう．よって，最適なデータを探索する際，このようなクラスタの存在により多くの余分なデータの走査が必要となり，計算時間が長くなる．

以降の実験では，比較・提案手法それぞれで計算時間を最短にした r_1 を用いている．

5.5 環境属性値半径 r_2 の影響

次に，クラスタの環境属性値半径 r_2 を変化させた場合の，計算時間を**図 7** (a) に，ディスク IO コストを図 7 (b)

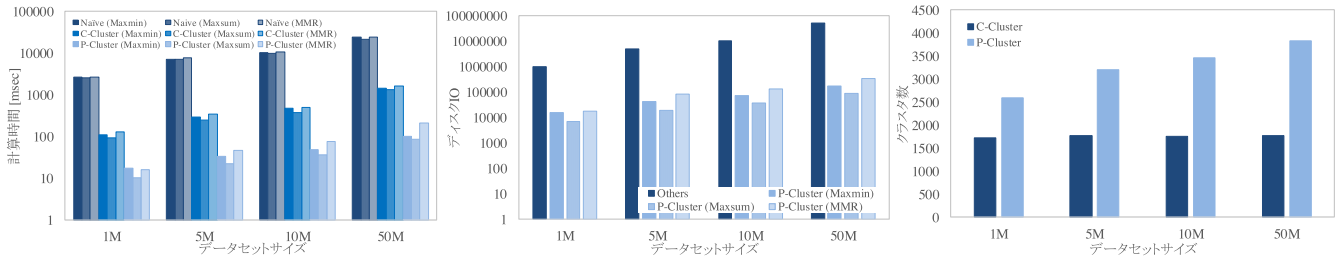


図 8 データセットサイズ N の影響
Fig. 8 Impact of dataset size N.

に，クラスタ数を図 7 (c) に示す．図 7 (a) から，環境属性値半径が大きい場合，提案手法における計算時間が長くなっていることが分かる．これは，空間半径を大きくした場合と同様に，クラスタ内に存在しうるデータのスコアを大きく見積もることで評価値の上界も過大に推定してしまい，走査対象から除外できたクラスタ数が少ないためである．一方，環境属性値半径が小さすぎる場合も，提案手法における計算時間が長くなっていることが分かる．これも空間半径を小さくした場合と同様，図 7 (c) に示されるように，環境属性値半径が小さくなると生成されるクラスタ数が増加し，走査が必要なデータの数も増加してしまうためである．

以降の実験では，提案手法において計算時間を最短にした r_2 を用いている．

5.6 その他のパラメタの影響

5.6.1 データセットサイズ N の影響

一般的に，ユーザはモバイルセンサデータの地理的分布を事前に知りえない．そのため，検索範囲内に存在するデータ数が大きい場合においても，検索結果を短時間で取得できることが重要である．そこで，データセットサイズ N を変化させた場合の，計算時間を**図 8** (a) に，ディスク IO コストを図 8 (b) に，クラスタ数を図 8 (c) に示す．いずれのデータ数の場合も，提案手法はすべての最適化問題に関して，計算時間およびディスク IO コストを大幅に削減している．図 8 (c) は，比較手法におけるクラスタ数が，データセットサイズ N が変化してもほぼ一定であることを示している．このことから比較手法では，データセットサイズ N が大きい場合，クラスタに含まれるデータの平均数が大きくなるため，アルゴリズムにおける走査データ数を削減する効果が小さくなることが分かる．一方，提案手法は，センシング誤差や環境属性値の変化に応じてクラスタを適切に分割できる．よって，データセットサイズ N の増加とともにクラスタ数も増加し追加処理のコストは大きくなるものの，全体として計算時間およびディスク IO コストは削減される．

5.6.2 要求データ数 k の影響

ユーザごとに，要求データ数は異なる．そこで，要求

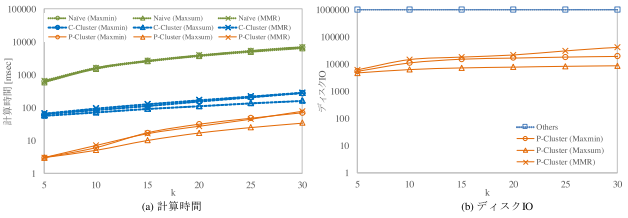


図 9 要求データ数 k の影響
Fig. 9 Impact of answer set size k.

データ数 k を変化させた場合の，計算時間を**図 9** (a) に，ディスク IO コストを図 9 (b) に示す．提案手法の効果は，検索範囲内に存在するホットスポットの数に依存する．すなわち，各ホットスポットにおけるデータがすべて探索されるまで，走査データ数削減の効果は大きく働く．ホットスポットにおけるデータの探索過程においては，提案手法はすべての最適化問題に関して，計算時間およびディスク IO コストともに効果的に削減できる．

5.6.3 属性の次元数 m の影響

ユーザごとに，注目する環境属性の次元数は異なる．また，次元の増大を考慮すると，最適な環境属性値半径 r_2 は大きく変化すると考えられる．そこで，まず最初に，それぞれの環境属性の次元数 m について，提案手法における環境属性値半径 r_2 を変化させた場合の，計算時間を**図 10** (a) に，ディスク IO コストを図 10 (b) に，クラスタ数を図 10 (c) に示す．ここではグラフの可読性のため，Maxmin 問題の結果のみを示している．図 10 (a) および (b) より，最適な環境属性値半径はそれぞれの次元数によって異なることが分かる．これは，高次元空間においては，データ間の距離の観点からデータどうしが相似しにくいためである．結果として，高次元空間で環境属性値半径 r_2 を小さくするとクラスタ数が急激に増加し，性能が低下してしまう．よって，以降の実験では，各次元数 m において最適な r_2 を用いている．

次に，環境属性の次元数 m を変化させた場合の，計算時間を**図 11** (a) に，ディスク IO コストを図 11 (b) に，クラスタ数を図 11 (c) に示す．いずれの次元数の場合も，すべての最適化問題に関して，提案手法が計算時間およびディスク IO コストを大幅に削減していることが分かる．

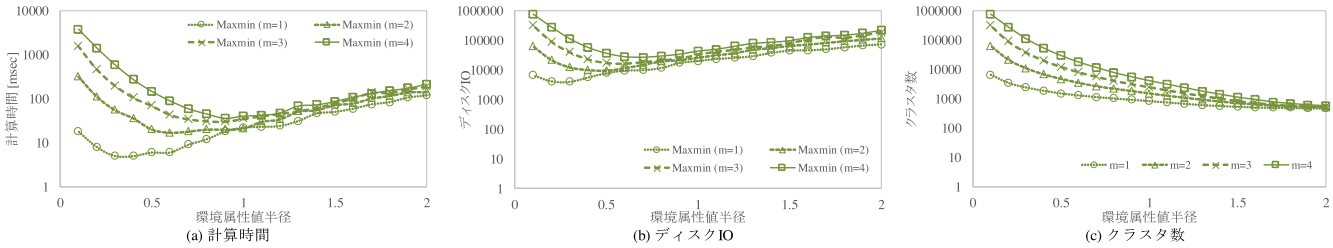


図 10 環境属性の次元数と環境属性値半径 r_2 の影響

Fig. 10 Impact of dimensionality of environmental attribute and environmental attribute radius r_2.

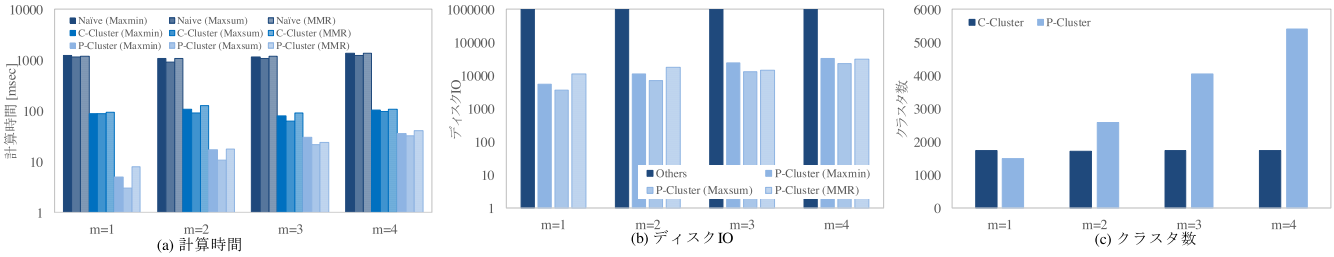


図 11 環境属性の次元数の影響

Fig. 11 Impact of dimensionality of environmental attribute.

6. 関連研究

Top-k クエリ処理は，全体のデータセットの中からスコアの最も高い（低い）k 個のデータを取得する検索であり，様々な研究分野において重要な役割を担っている．固定センサネットワークにおける Top-k 検索は，最も高い（低い）環境属性値を観測した k 個のノードを検索結果として返す．固定センサネットワークにおける Top-k 検索に関して，多数の研究が行われており [9], [12], [13]，通信コストと消費電力の削減を目的として様々なデータ集約手法が提案されている．一般的に，固定センサネットワークでは，設置されているセンサはセンシング間隔が同期して動作しており，設置位置も観測範囲内で分散している．一方，モバイルセンサネットワークでは，モバイルセンサは非同期かつ時空間上の任意の位置でセンサデータを生成する．このような環境では，従来の Top-k 検索を用いた場合，多くの冗長なセンサデータが検索結果に含まれてしまう．よって，検索範囲内に存在するホットスポットにおいて生成されたデータを取得するためには，別の手法が要求される．

また，検索範囲内における特定の物理現象に関する知識を得るために，集約クエリが有効であると考えられる．特定の領域の環境属性値を集約することで，その領域における物理現象に関する主要な情報は，平均や合計，分散といった代表的な統計量によって表現される．文献 [11], [15] では，インデックス構造を含む様々なフレームワークが提案され，このような集約処理を効率化している．しかし，一般的にユーザはセンサデータの時空間分布を事前に知りえない．また，集約によって検索範囲内の環境属性値は平均されてしまい，ホットスポットの検出は非常に難しくなる．その結果，ユーザは広大な地理空間を，繰り返し検索せざるをえない恐れがある．そこで本論文では，この問題に対処するため，著者らが知る限り初めて，Top-k 検索結果の多様化を環境モニタリングを目的としたモバイルセンサデータベースに適用した．

従来の Top-k 検索では，検索結果はデータの関連度にのみ基づいて定まるが，ユーザの要求をより満たせるように検索結果の多様性を考慮した研究がさかんに行われている [4], [7], [16]．研究対象は，文書に対するキーワード検索，EC サイトにおける商品推薦など多岐にわたり，多くの既存手法では，検索範囲内の対象データに関するデータのスコアはすべて既知とした状態を仮定している．しかし，この仮定を満たすためには，検索範囲内のデータをすべてファイルから読み込み，さらに初期処理としてすべてのデータのスコアを計算しなければならない．一方，本論文における提案手法では，モバイルセンサデータはオフライン事前クラスタリング処理によって構造化して管理されている．これにより，たとえ事前にユーザのクエリパラメタが入力されなくても，限られたデータのみにアクセスすることで検索結果を短時間で取得できる．

7. おわりに

本論文では，モバイルセンサデータベースにおける Top-k 検索結果の多様化について取り組み，効率的なクエリ処理手法を提案した．単純なグリーディアルゴリズムでは，すべてのデータについて評価値を計算し，正解集合のサイズが k になるまで最適なデータを正解集合に追加する必要が

あるため，計算コストが非常に大きい．一方，提案手法では，環境属性値の空間的自己相関と呼ばれる特徴を利用し，グリーディアルゴリズムの評価値が互いに近いデータどうしを事前にクラスタリングする．評価値計算をクラスタ単位で行い，クラスタの半径情報を利用することで，最適なデータを含み得ないクラスタを走査対象から除外する．その結果，走査するデータ数を大幅に削減できる．

クラスタ半径，データセットサイズ，要求データ数 k，環境属性数といった様々な要因を変化させたシミュレーション実験から，提案手法は単純手法に比べ，計算時間およびディスク IO コストを大幅に削減できることを確認した．

近年，Maxmin 問題や Maxsum 問題に基づいた，継続的な検索結果の多様化問題がさかんに取り組まれている [5], [14]．今後の課題の 1 つとして，モバイルセンサデータベースにおける，効率的かつ高精度な多様化結果のモニタリングの検討があげられる．

謝辞 本研究の一部は，文部科学省科学研究費補助金・基盤研究（A）（26240013）および JST 国際科学技術共同研究推進事業（戦略的国際共同研究プログラム）の研究助成によるものである．ここに記して謝意を表す．

参考文献

[1] Beckmann, N., Kriegel, H.-P., Schneider, R. and Seeger, B.: The R*-tree: An Efficient and Robust Access Method for Points and Rectangles, *ACM SIGMOD*, pp.322–331, ACM (1990).
[2] Carbonell, J. and Goldstein, J.: The Use of MMR, Diversity-based Reranking for Reordering Documents and Producing Summaries, *ACM SIGIR*, pp.335–336, ACM (1998).
[3] Drosou, M. and Pitoura, E.: Diversity over Continuous Data., *IEEE Data Eng. Bull.*, Vol.32, No.4, pp.49–56 (2009).
[4] Drosou, M. and Pitoura, E.: Disc diversity: result diversification based on dissimilarity and coverage, *VLDB*, Vol.6, No.1, pp.13–24 (2012).
[5] Drosou, M. and Pitoura, E.: Dynamic Diversification of Continuous Data, *EDBT*, pp.216–227, ACM (2012).
[6] D'Hondt, E., Stevens, M. and Jacobs, A.: Participatory noise mapping works! An evaluation of participatory sensing as an alternative to standard techniques for environmental monitoring, *Pervasive and Mobile Computing*, Vol.9, No.5, pp.681–694 (2013).
[7] Fraternali, P., Martinenghi, D. and Tagliasacchi, M.: Top-k Bounded Diversification, *ACM SIGMOD*, pp.421–432, ACM (2012).
[8] Gollapudi, S. and Sharma, A.: An Axiomatic Approach for Result Diversification, *World Wide Web*, pp.381–390, ACM (2009).
[9] Jiang, H., Cheng, J., Wang, D., Wang, C. and Tan, G.: A General Framework for Efficient Continuous Multidimensional Top-k Query Processing in Sensor Networks, *IEEE Trans. Parallel and Distributed Systems*, Vol.23, No.9, pp.1668–1680 (2012).
[10] Khan, H.A., Sharaf, M.A. and Albarrak, A.: DivIDE: Efficient Diversification for Interactive Data Exploration, *SSDBM '14*, ACM (2014).
[11] Lazaridis, I. and Mehrotra, S.: Progressive Approximate Aggregate Queries with a Multi-resolution Tree Structure, *ACM SIGMOD*, pp.401–412, ACM (2001).
[12] Liu, X., Xu, J. and Lee, W.-C.: A Cross Pruning Framework for Top-k Data Collection in Wireless Sensor Networks, *International Conference on Mobile Data Management* (*MDM*), pp.157–166 (2010).
[13] Malhotra, B., Nascimento, M. and Nikolaidis, I.: Exact Top-K Queries in Wireless Sensor Networks, *IEEE Trans. Knowledge and Data Engineering*, Vol.23, No.10, pp.1513–1525 (2011).
[14] Minack, E., Siberski, W. and Nejdl, W.: Incremental Diversification for Very Large Sets: A Streaming-based Approach, *ACM SIGIR*, SIGIR, ACM, pp.585–594 (2011).
[15] Papadias, D., Kalnis, P., Zhang, J. and Tao, Y.: Efficient OLAP Operations in Spatial Data Warehouses, *Advances in Spatial and Temporal Databases*, Vol.2121, pp.443–459, Springer (2001).
[16] Qin, L., Yu, J.X. and Chang, L.: Diversifying top-k results, *VLDB*, Vol.5, No.11, pp.1124–1135 (2012).
[17] Sazaki, H., Kanzaki, A., Hara, T. and Nishio, S.: SeRAVi: A Spatio-Temporal Data Distribution Visualization System for Mobile Sensor Data Retrieval, *Reliable Distributed Systems Workshops* (*SRDSW*), pp.88–93, IEEE (2014).
[18] Tao, Y., Hristidis, V., Papadias, D. and Papakonstantinou, Y.: Branch-and-bound processing of ranked queries, *Information Systems*, Vol.32, No.3, pp.424–445 (2007).
[19] Vuran, M.C., Akan, Ö.B. and Akyildiz, I.F.: Spatio-temporal correlation: Theory and applications for wireless sensor networks, *Computer Networks*, Vol.45, No.3, pp.245–259 (2004).

横山 正浩 （学生会員）

2012 年大阪大学工学部情報システム工学科卒業．2014 年同大学大学院情報科学研究科博士前期課程修了後，同博士後期課程に在学中．モバイルセンサデータのデータ管理・問合せ機構に関する研究に従事．日本データベース学会学生会員．

原 隆浩 （正会員）

1995 年大阪大学工学部情報システム工学科卒業．1997 年同大学大学院工学研究科博士前期課程修了．同年同大学院工学研究科博士後期課程中退後，同大学院工学研究科助手，2002 年同大学院情報科学研究科助手，2004 年同大学院情報科学研究科准教授．2015 年より同大学院情報科学研究科教授となり，現在に至る．工学博士．1996 年本学会山下記念研究賞受賞．2000 年電気通信普及財団テレコムシステム技術賞受賞．2003 年本学会研究開発奨励賞受賞．2008 年，2009 年本学会論文賞，2015 年日本学術振興会賞受賞．モバイルコンピューティング，ネットワーク環境におけるデータ管理技術に関する研究に従事．IEEE，ACM，電子情報通信学会，日本データベース学会の各会員．