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関する研究

提出先　　大阪大学大学院情報科学研究科

提出年月　2018 年 1 月

横 山　正 浩

関連発表論文

1. 学会論文誌発表論文

1. 横山正浩, 義久智樹, 原 隆浩, 西尾章治郎：P2P ストリーミング環境におけるモデル・動作分離型コンテンツの再生途切れ時間短縮方式, 情報処理学会論文誌, Vol. 54, No. 11, pp. 2360–2370 (2013 年 11 月).

2. 横山正浩, 原 隆浩：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, 情報処理学会論文誌, Vol. 58, No. 2, pp. 481–494 (2017 年 2 月).

2. 国際会議発表論文

1. Yokoyama, M., Yoshihisa, T., Hara, T., and Nishio, S.: A data reception method to reduce interruption time for model and motion separated data in P2P streaming environments, Proc. Int. Conf. on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC 2012), pp. 79–86 (Nov. 2012).

2. Yokoyama, M., Yoshihisa, T., Hara, T., and Nishio, S.: A method to reduce interruption time for the multiple bit rate video delivery in P2P streaming environments, Proc. Int. Conf. on Advances in Mobile Computing and Multimedia (MoMM 2014), pp. 253–262 (Dec. 2014).

3. Yokoyama, M. and Hara, T.: Efficient top-k result diversification for mobile sensor data, Proc. Int. Conf. on Distributed Computing Systems (ICDCS 2016), pp. 477-486 (June 2016).

4. Nakashima, K., Yokoyama, M., Taniyama, Y., Yoshihisa, T., and Hara, T.: S^3 system: a system for sharing social sensor data and analytical programs, Adjunct

Proc. Int. Conf. on Mobile and Ubiquitous Systems: Computing, Networking and Services (MOBIQUITOUS 2016), pp. 147–152 (Nov. 2016).

5. Yokoyama, M., Hara, T., and Madria, S. K.: Efficient diversified set monitoring for mobile sensor stream environments, Proc. Int. Conf. on Big Data (Big Data 2017), pp. 500–507 (Dec. 2017).

3. 国内会議発表論文（査読有）

1. 横山正浩, 義久智樹, 原 隆浩, 西尾章治郎：P2P ストリーミング環境におけるモデル・動作分離型コンテンツの再生途切れ時間短縮のための分割データ受信方式, 情報処理学会マルチメディア，分散，協調とモバイル (DICOMO 2012) シンポジウム論文集, Vol. 2012, No. 1, pp. 1215–1222 (2012 年 7 月).

2. 横山正浩, 義久智樹, 原 隆浩, 西尾章治郎：P2P ストリーミング環境における課金型マルチビットレート映像配信サービスのための再生途切れ時間短縮方式, 情報処理学会マルチメディア通信と分散処理ワークショップ (DPSWS 2013) 論文集, Vol. 2013, No. 6, pp. 246–253 (2013 年 12 月).

3. 加藤 諒, 原 隆浩, 白川真澄, 駒井友香, 真嶋温佳, 天方大地, 大澤 純, 松尾和哉, 横山正浩, 佐崎 悠, 中村達哉, 水野聖也, 西尾章治郎：位置プライバシ問題の啓蒙のためのダミーによる位置曖昧化手法の体験システムの実装, 情報処理学会マルチメディア通信と分散処理ワークショップ (DPSWS 2014) 論文集, Vol. 2014, No. 5, pp.50–52 (2014 年 12 月).

4. 横山正浩, 原 隆浩, 西尾章治郎：モバイルセンサデータベースにおける効率的な Top-k 検索結果の多様化について, Web とデータベースに関するフォーラム (WebDB Forum 2015) 論文集, Vol. 2015, pp. 118–125 (2015 年 11 月).

5. 中嶋奎介, 谷山雄基, 横山正浩, 義久智樹, 原 隆浩, 西尾章治郎：社会センサデータ生成・共有基盤システムの設計と実装, 情報処理学会マルチメディア,

分散，協調とモバイル (DICOMO 2016) シンポジウム論文集, Vol. 2016, pp. 1215–1222 (2016 年 7 月).

6. 愛甲善之助, 谷山雄基, 横山正浩, 中嶋奎介, 義久智樹, 原 隆浩：社会センサデータ生成・共有基盤におけるデータフロー制御機構, 情報処理学会マルチメディア，分散，協調とモバイル (DICOMO 2017) シンポジウム論文集, Vol. 2017, pp. 1026–1033 (2017 年 7 月).

4. その他の研究会等発表論文

1. 横山正浩, 原 隆浩：モバイルセンサデータベースにおける階層的クラスタリングを用いた Top-k 検索結果の多様化について, データ工学と情報マネジメントに関するフォーラム (DEIM 2017), online (2017 年 3 月).

2. 横山正浩, 原 隆浩, Madria, S. K.：モバイルセンサデータストリーム環境における継続的な Top-k 検索結果の多様化について, 電子情報通信学会技術研究報告, Vol. 117, No. 212, pp. 7–12 (2017 年 9 月).

3. 中嶋奎介, 横山正浩, 義久智樹, 原 隆浩：社会センサデータ生成・共有生成基盤における社会センサ可視化機構, 電子情報通信学会技術研究報告, Vol. 117, No. 212, pp. 13–18 (2017 年 9 月).

以上

内容梗概

近年，半導体技術の発展により，スマートフォンやウェアラブルデバイスなどの人々が身につけて持ち歩く携帯端末に，音や温度，湿度などの環境情報を生成できるセンサが搭載されつつある．これらのセンサを搭載した携帯端末の普及に伴い，一般ユーザがもつ端末が生成したモバイルセンサデータ（環境情報および位置情報）を提供してもらい，都市部における環境モニタリングなどのサービスに利活用するユーザ参加型センシングが注目されている．端末保持者がセンシング領域を移動しながらデータを生成することで，従来の固定センサネットワークに比べて，より細かい地理的粒度でのデータ収集が可能となる．

収集されたモバイルセンサデータを参照し分析することで，センシング領域の環境情報を調査できる．一般的に，検索を行うユーザは自身の興味に適合する少数のデータにのみ関心がある．そこで，モバイルセンサデータの環境属性値から，ユーザの興味に基づいて算出されるスコアを定義し，よりユーザが関心を示す高いスコアをとる少数のデータを取得することが重要である．さらに，データ間の空間距離を併せて考慮し，地理空間上で広く分散し，かつ高いスコアをとるデータの集合を考える．このようなデータの集合を取得することで，その時々のユーザの興味に基づいた注目すべき領域を，広大なセンシング領域から効果的に割り出せる．高いスコアをとり，かつ地理空間上で分散するデータからなるデータ集合を，本論文では**多様集合**と呼び，また多様集合を取得することを **Top-k 検索結果の多様化**と呼ぶ．モバイルセンサデータに対する Top-k 検索結果の多様化は，広大なセンシング領域の環境情報を調査する方法として有用であるが，大量のモバイルセンサデータに対して多様集合を計算するための単純な手法を用いると，データセット全体を複数回走査する必要があり，計算コストが極めて大きくなってしまう．

そこで本論文では，モバイルセンサデータベースにおいて，ユーザの興味に適合し地理空間上で分散した多様集合を短時間で計算する手法について議論する．本論文ではまず，過去に収集されて蓄積された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．次に，モバイルセンサデー

タの環境情報が高次元である場合の，計算効率の低下の影響を緩和するための手法を提案する．また，モバイルセンサデータが時々刻々と到着するモバイルセンサストリーム環境における，継続的な Top-k 検索結果の多様化手法を提案する．

本論文は，5 章から構成され，各章の内容は次の通りである．まず，第 1 章において，序論として研究の背景と目的について述べる．

第 2 章において，環境属性値の空間的相関性を考慮した，クラスタリングを用いた効率的な Top-k 検索結果の多様化手法を提案する．この手法では，クエリが到着する前にオフラインでモバイルセンサデータのクラスタリング処理を行い，生成されたクラスタの性質を利用した効率的なオンラインクエリ処理を行う．提案するオンラインクエリ処理手法では，作成したクラスタ構造の性質を利用し，データセット全体と比較して少数のモバイルセンサデータのみを走査することで，計算時間を短縮する．また，提案手法の性能評価のために行ったシミュレーション実験の結果を示し，その有効性について検証する．その結果より，提案手法では，すべてのデータを複数回走査する必要のある既存手法で得られる多様集合と同一の検索結果を，短時間で取得できることを確認した．

次に第 3 章において，第 2 章の手法における環境情報が高次元である場合の問題点に言及し，その問題点を緩和する手法を提案する．第 2 章におけるクラスタリングでは，モバイルセンサデータの環境属性値ベクトルの類似度に基づいてクラスタリングを行っている．そのため，モバイルセンサデータが高次元である場合，クラスタの数が大幅に増加することでオンラインクエリ処理に要する時間が増加してしまう．そこで，第 3 章の手法では，階層的クラスタリング処理によりモバイルセンサデータを 2 階層のクラスタに構造化する．上位クラスタはモバイルセンサデータの空間位置の近接性のみを考慮して構成されるため，その数は環境属性の次元数に依存しない．これにより，クラスタの性質を利用したオンラインクエリ処理における，走査対象のデータの絞り込みのための計算コストを削減でき，第 2 章の手法よりも計算時間を短縮できる．また，提案手法の性能評価のために行ったシミュレーション実験の結果を示し，その有効性について検証する．その結果より，モバイルセンサデータの環境情報が高次元の場合でも，提案手法は第 2 章で提案した手法と比較してより短時間で多様集合を取得できることを確認した．

第 4 章では，ユーザの興味に基づいた注目すべき領域のリアルタイムの追跡を目的とし，モバイルセンサストリーム環境における継続的な Top-k 検索結果の多様化手法を提案する．この手法では，新たに追加，削除されるモバイルセンサデータを短時間で構造化して管理するために，格子グリッドベースのデータ構造を用いる．モバイルセンサデータの位置情報から割当て先の空間セルを決定し，さらに空間セルの中で環境属性値に基づいて格納先のデータリストを決定する．格子グリッドベースのデータ構造の空間セルの性質を利用し，モニタリングの対象のデータセット全体と比較して少数のモバイルセンサデータのみを走査することで，計算時間を短縮する．また，提案手法の性能評価のために行ったシミュレーション実験の結果を示し，その有効性について検証する．その結果より，提案手法は第 2 章で提案した手法と比較して，より短時間で多様集合を更新できることを確認した．

最後に，第 5 章では，本論文の成果を要約した後，今後の研究課題について述べる．

目 次

第1章　序章

1.1　研究背景

近年，スマートフォンやウェアラブルデバイスなど，人々が身につけて持ち歩く携帯端末に様々なセンサが搭載されつつある．これらのセンサを搭載した携帯端末の普及に伴い，センサが生成する位置情報をはじめとした情報を定期的に取得し，これを活用するライフログ[1][2]や位置情報連動型サービス[3][4]などの普及が進んでいる．また，上記のような端末に搭載されるセンサの多様化が進み，音や温度，湿度などの環境情報を生成できる端末も普及しつつある．これらのセンサから取得した環境情報は，ライフログなどのサービスがユーザに提供できる情報を豊富にするだけでなく，多数の端末から収集することで，都市部における環境モニタリングをはじめとしたサービスにも利活用できるものと考えられる．このような背景のもと，一般ユーザが持つ端末が生成した環境情報を含むセンサデータを提供してもらい，環境モニタリングをはじめとする様々なサービスへ利活用するユーザ参加型センシングが注目されている [30, 41, 45, 50, 68, 69]．ユーザ参加型センシングでは，センサデータを提供したユーザに，収集したセンサデータから得られる有用な情報を提供するといったインセンティブを与えるなど，センサデータを提供しやすい環境を構築することで，多数のユーザからのセンサデータの収集を実現している．

図 1.1 に示すように，ユーザ参加型センシングでは，センサデータは町中を移動するユーザによって生成され，LTE や WiMAX など，インターネットに接続可

[1]Lifelog, http://www.d-healthcare.co.jp/products/moveband3/index.html
[2]MOVEBAND3, http://www.d-healthcare.co.jp/products/moveband3/index.html
[3]Growth Push, https://growthpush.com/
[4]popinfo, http://www.d-healthcare.co.jp/products/moveband3/index.html

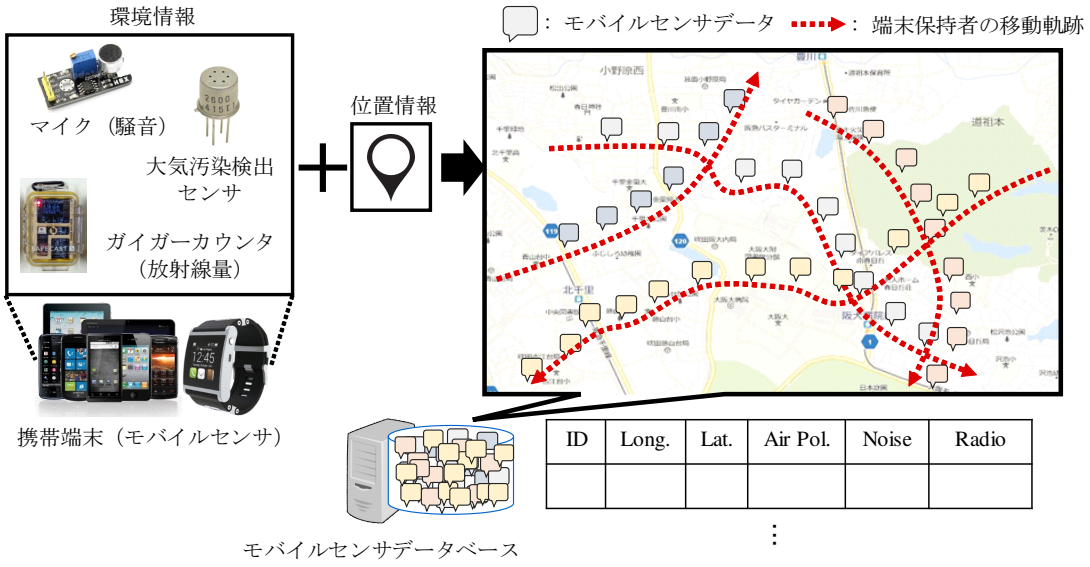
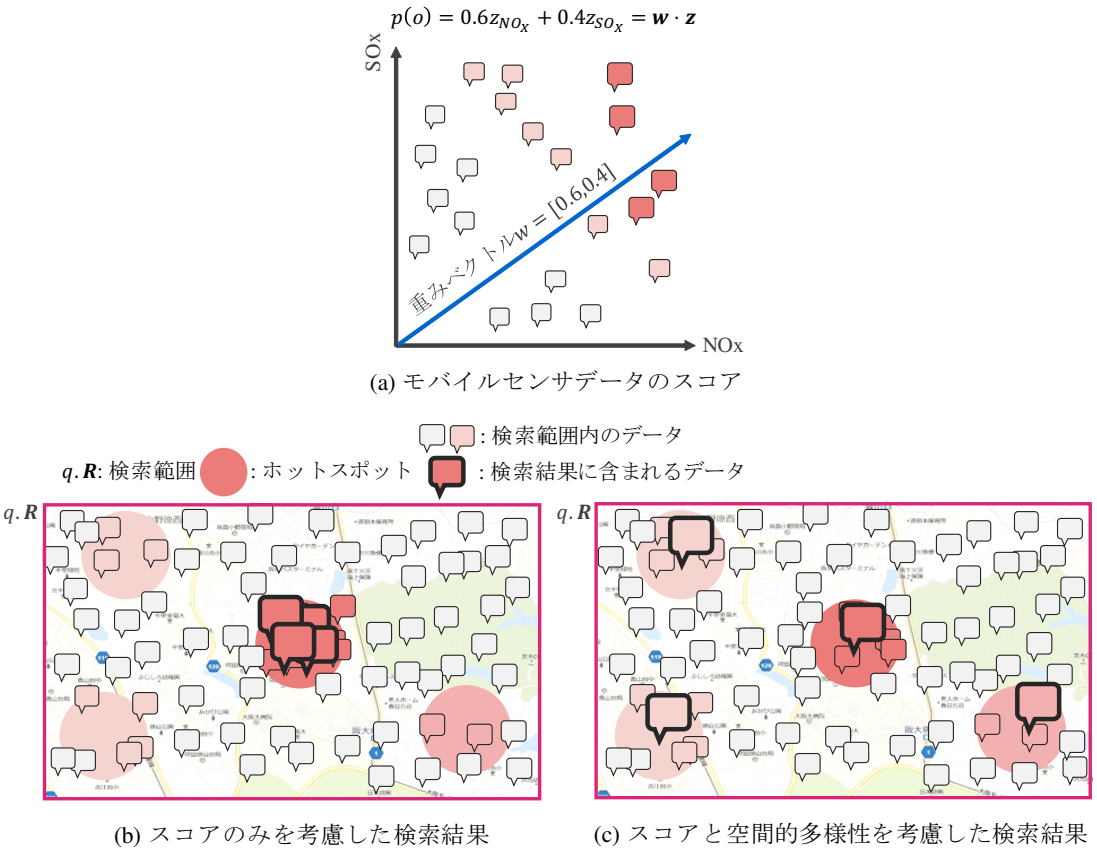


図 1.1: ユーザ参加型センシングとモバイルセンサデータベース

能な通信インフラを介して，集中管理のデータベースに格納される．このような，端末保持者が移動しながら生成するデータを本論文では**モバイルセンサデータ**と呼び，モバイルセンサデータが格納されるデータベースを**モバイルセンサデータベース**と呼ぶ．モバイルセンサデータは，観測時の位置情報，観測時刻，気温・湿度・大気汚染指数・騒音指数といった複数の環境属性値からなる多次元データとしてモバイルセンサデータベースに蓄積される．ユーザ参加型センシングでは，従来の固定センサネットワークとは異なりセンサデバイスを新たに設置することなく，多数の一般ユーザが持つ携帯端末を利用してセンシングすることで，より細かい地理的粒度でのデータ収集が可能となる [64]．

収集されたモバイルセンサデータから注目するセンシング領域の環境情報を調べるにあたって，まずは時空間範囲を指定し範囲内に存在するモバイルセンサデータのみを取得する，時空間範囲検索 [55, 60, 61, 66, 74, 75] が想定される．しかし，センシング領域が広大であり収集されるモバイルセンサデータの量が膨大である場合，有用な情報を得るために，検索のたびに範囲内のすべてのデータを調べることは困難である．一方で，検索を行うユーザが注目する環境属性値が極端な値は，特定の注目すべき物理現象の発生を示唆するものである．このようなデータ



(a) モバイルセンサデータのスコア

(b) スコアのみを考慮した検索結果

(c) スコアと空間的多様性を考慮した検索結果

図 1.2: データのスコアとホットスポット

が得られる領域を，本論文では**ホットスポット**と呼ぶ．ホットスポットはユーザが注目すべき領域であるため，広大なセンシング領域から効果的に検出することが重要である．

極端な値を示す個々のデータを取得するためには，Top-k 検索 [14, 40] が有効である．Top-k 検索では，分析の目的に応じて，注目する環境属性値（気温，湿度，騒音指数，大気汚染指数など）に重み付けをし（図 1.2(a)），データにスコアを割り当てる．このスコアによってデータを順位付けし，上位 k 個のデータを取得する．これまでに，固定センサネットワークにおける様々な Top-k 検索に関する研究が行われている [6, 42, 54, 72, 81, 82]．固定センサネットワークにおける Top-k 検索では，各センサが観測したデータはシンクへと集められ，シンクに対し検索

要求を送信することで上位 k 個のセンサが返される．一方，モバイルセンサデータベースにおける Top-k 検索では，指定した時空間範囲内に存在するデータから上位 k 個のモバイルセンサデータが返される．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 [17, 24, 38, 79] と呼ばれる特徴を有する．そのため，Top-k 検索結果に含まれる大部分のデータは，図 1.2(b) 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．図中の 4 つのホットスポットは，周辺と比較して十分に高いスコアをとる，ユーザにとって注目すべき領域である．しかし，中央のホットスポットから生成されたデータのスコアが，他の 3 つのホットスポットから生成されたデータのスコアに比べて高いため，$k = 4$ とした場合の上位 4 個のデータは，中央のホットスポットからしか取得できない．このように，Top-k 検索では，しばしば得られる解の冗長性が問題となることが指摘されている [15, 94].

1.2　多様集合の利用

図 1.2(c) 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果的に取得するためには，**Top-k 検索結果の多様化** [33, 76, 92] が有効である．この処理は，ユーザのクエリとの関連度が高いデータで，かつ結果に含まれるデータ同士ができる限り類似しないようなデータの組合せ，すなわち**多様集合**を取得することを目的としている．Top-k 検索結果の多様化は，Top-k 検索結果の冗長性を解消し，よりユーザに対して満足度の高い結果を返すことを目的として，Web 文書に対するキーワード検索 [15]，オンラインショッピングにおける商品検索 [78]，推薦システム [12] など，多岐にわたって研究が行われている．文献 [18, 28, 31, 36] では，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をモデル化し，目的関数の最大化を目指す組合せ最適化問題に帰着して，検索結果の多様化を行っている．この問題は NP 困難であることが示されており，データセットサイズが大きいとき，すべての部分集合を総当りで探索するのは，計算

時間の観点から現実的ではない．そのため，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多様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 [26, 31, 34].

ここで，既存研究におけるデータのクエリに対する関連度，およびデータ間の非類似度を，それぞれユーザの検索基準に基づいたデータのスコア，およびデータの位置情報から計算されるデータ間の空間距離と置き換えると，データのスコアが高くかつ検索範囲内でより空間的に分散するデータの組合せを取得できる．従って，Top-k 検索結果の多様化をモバイルセンサデータベースに適用することで，上述したホットスポットの検出に応用できる．

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をモデル化し，目的関数を最大化することによって解かれる．図 1.2(c) に示すような地理空間上のホットスポットの検出を目的として，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，データの関連度は環境情報から算出されるデータのスコアに，多様性は位置情報から算出されるデータ間の空間距離に相当する．よって，最適化問題を解くためのグリーディアルゴリズムでは，データのスコアとデータ間の空間距離から算出されるデータの評価値を計算し，最大の評価値をとるデータを探索する必要がある．Top-k 検索とは異なり，グリーディアルゴリズムにおけるデータの評価値の計算には，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，個々の単独のデータからは計算出来ない．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，最大の評価値をとる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．一方で，本研究で想定している環境では，クエリごとにユーザの環境属性値に対する関心は異なるため，クエリごとにデータのスコアは異なる．そのため，クエリが到着する前にデータのスコアは計算できず，検索範囲内のデータを取得した段

階では，データのスコアは未知であると仮定している．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，評価値が最大のデータを取得するために，クエリごとにすべてのデータのスコアを計算する必要がある．そのため，データセット全体を走査することは避けられず，データセットサイズが極めて大きい場合，計算コストも比例して大きくなる．

1.3　研究内容

本論文では，すべてのデータを複数回走査する必要のある既存手法で得られる多様集合と同一の結果を，短時間で取得することを目的として，モバイルセンサデータベースにおける Top-k 検索結果の多様化手法について議論する．具体的には，以下の 3 つの研究課題に取り組む．

1.3.1　過去に収集され蓄積された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（第 2 章）

1.2 節の後半で説明したように，既存の多様化手法をモバイルセンサデータの検索に適用した場合，定義された目的関数のもとで評価値が最大のデータを取得するために，クエリごとにすべてのデータのスコアを計算する必要があり，データセット全体を走査する必要がある．また，従来の Top-k 検索では，データのスコアの計算は一度だけであるのに対し，Top-k 検索結果の多様化では，すでに検索結果として選択されているデータとの間の空間距離を繰り返し計算し直す必要がある．このため，データセットサイズが極めて大きい場合，計算コストも比例して大きくなる．

そこで，過去に収集され蓄積されたモバイルセンサデータに対し，事前にデータをクラスタ構造化することによって，短時間で多様集合を取得する手法を提案する．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，かつ既存手法のようにすべてのデータを複数回走査することで得られる多様集合と同一の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．データの空間位置が近く，かつデータの環境属性値が似ているデータは，互いに評価値が近い値となる．このようなデータを 1 つのクラスタにまとめることで，クラスタ内のデータの評価値の計算を避けられる場合がある．このため，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同士をクラスタ化し，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．環境属性値の類似度を併せて考慮する理由は，センシング誤差により，空間的に互いに近接するデータであっても，環境属性値が大きく異なる場合があるためである．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取りうる評価値の上界を計算する．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取りうる評価値の上界を出来る限り小さく，短時間で計算できる．これにより，計算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，多様集合に追加すべき最適なデータを短時間で探索できる．結果として，走査するデータの数を大幅に削減しつつ，既存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，提案手法のアプローチは，多くの既存研究で対象とされている，複数の種類の最適化問題に対して同様に適用可能である．

1.3.2 高次元なモバイルセンサデータへの対応（第 3 章）

近年，ユーザ参加型センシングの取り組みが多数行われており [13, 37, 52, 67]，収集される環境情報も多岐にわたっている．また，様々なセンサデバイスやセンサネットワークを仮想化し，クラウド的にセンサデータを管理するセンサクラウド [2, 53, 59] と呼ばれるフレームワークも注目を集めている．このような背景から，モバイルセンサデータがより豊かな環境情報を持つようになり，モバイルセンサデータの環境属性値ベクトルは，高次元化が進んでいる．このような状況で，1.3.1 項で説明したクラスタベースの手法を用いる場合，計算時間が増加する問題

が生じる．これは，環境属性値ベクトルが高次元になるほどデータ間の類似度が小さくなるため，球形クラスタの半径が大きく，かつクラスタの数も大幅に増加してしまい，走査するデータの数を削減するための計算コストが大きくなるためである．

そこで，事前クラスタリング処理において階層的クラスタリングを行うことで，環境属性値ベクトルの次元数の増加にともなう計算効率の低下の影響を緩和する手法を提案する．提案手法の階層的クラスタリングでは，空間的に近接するすべてのデータはある1つの上位クラスタのメンバとし，上位クラスタの配下でメンバデータの環境属性値の類似度に基いてさらに下位クラスタに分割する．空間的に近接するデータからなる上位クラスタの数は，環境属性の次元数の増加に非依存であり，小さく保たれる．提案手法のオンラインクエリ処理では，階層クラスタ構造を利用し，上位クラスタ内のデータが取りうる評価値の上界を計算することで，下位クラスタをまとめて走査対象から除外できる．そのため提案手法では，第2章におけるクラスタベースの手法と比べて，走査対象のデータをより短時間で絞り込める．

1.3.3 モバイルセンサストリーム環境における継続的なTop-k検索結果の多様化（第4章）

モバイルセンサデータが時々刻々と到着するモバイルセンサストリーム環境において，地理的多様性を考慮した多様集合をモニタリングすることで，ホットスポットの地理的分布の変化をリアルタイムに追跡できる．モバイルセンサストリーム環境において多様集合をモニタリングする場合，多様集合内のデータが一定時間の経過により削除される場合や，よりスコアが大きい，もしくはより地理的に分散するデータが生成された場合に，多様集合を更新する必要がある．過去に収集され蓄積されたモバイルセンサデータベースにおける多様集合の計算と同様に，単純な手法では多様集合の更新に要する計算コストは大きい．

一方で，モバイルセンサストリーム環境において，第2章や第3章の事前クラスタリング処理手法を用いる場合，多様集合の更新の前に，到着するデータや削除

されるデータに対するクラスタ構造の更新を行う必要がある．クラスタ構造を利用することで多様集合の更新は短時間で可能となるものの，クラスタ構造の更新にかかる時間が非常に大きくなるため，結果的に多様集合の更新時間が長くなってしまう．

そこで，データの更新コストが小さい格子グリッドベースのデータ構造を用いた，モバイルセンサストリーム環境における効率的な多様集合のモニタリング手法を提案する．格子グリッドの空間セルは，環境属性ごとにデータのリストを保持しており，それぞれのリストは環境属性値が一定の範囲内であるデータを管理する．それぞれのデータを被覆する空間セルおよび管理するリストは定数時間で決定できるため，提案データ構造はストリーム環境に適している．提案アルゴリズムでは，最適なデータを探索する際，空間セル内のデータがとりうるスコアの上界を計算できる．また，この上界は空間セル内のデータの走査が進むにつれて減少し，これにより空間セル内のデータの走査を途中で打ち切り，次の空間セルの走査に進める．結果として，走査するデータの数を大幅に削減しつつ，既存手法のようにすべてのモニタリングの対象のデータを走査することで得られる多様集合と同一の結果を取得できる．

1.4 本論文の構成

本論文は，5 章から構成され，本章以降の内容は次の通りである．

第 2 章では，モバイルセンサデータを空間的な近接性，および環境属性値の類似度に基づいてクラスタリングし，効率的に Top-k 検索結果の多様化を行う手法を提案する．この手法では，1.3.1 項で述べた通り，クエリが到着する前にオフラインでモバイルセンサデータのクラスタリング処理を行い，生成されたクラスタの性質を利用し，オンラインクエリ処理を行う．提案手法のオンラインクエリ処理では，クラスタ内のデータの取りうる評価値の上界を計算し，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ことで，計算時間を短縮する．また，提案手法の性能評価のために行ったシミュレーション実験の結果を示し，その有効性について検証する．

第 3 章では，モバイルセンサデータの環境情報が高次元である場合を想定し，階層的クラスタリングを用いた，クラスタの数の増加に伴う計算効率の低下を緩和する手法を提案する．この手法では，1.3.2 項で述べた通り，空間的に近接するデータから上位クラスタを構築し，その配下で環境属性値の類似度に基づいて下位クラスタに分割する．提案手法のオンラインクエリ処理では，階層クラスタ構造を利用し，上位クラスタ内のデータが取りうる評価値の上界を計算することで，下位クラスタをまとめて走査対象から除外することで，走査対象のデータの絞り込みに要する計算コストを削減できる．また，提案手法の性能評価のために行ったシミュレーション実験の結果を示し，その有効性について検証する．

第 4 章では，モバイルセンサデータが時々刻々と到着するモバイルセンサストリーム環境における，継続的な Top-k 検索結果の多様化手法を提案する．この手法では，1.3.3 項で述べた通り，新たに追加，削除されるモバイルセンサデータを短時間で構造化して管理するために，格子グリッドベースのデータ構造を用いる．提案手法のクエリ処理では，格子グリッドベースのデータ構造の空間セルの性質を利用し，モニタリングの対象のデータセット全体と比較して少数のモバイルセンサデータのみを走査するように，途中でデータの走査を打ち切ることで，計算時間を短縮する．また，提案手法の性能評価のために行ったシミュレーション実験の結果を示し，その有効性について検証する．

最後に第 5 章では，本論文の成果を要約したのち，今後の研究課題について述べる．

第 2 章は文献 [85, 87, 90] で公表した結果に，第 3 章は文献 [86] で公表した結果に，第 4 章は文献 [88, 89] で公表した結果に基づき論述する．

第2章　空間的相関性を考慮したクラスタリングを用いたTop-k検索結果の多様化手法

2.1　まえがき

図 1.2(a) 中の式 $p(\cdot)$ のように，注目する環境属性値に重み付けをして計算されるデータのスコアが高く，かつ空間的に分散した多様集合を取得するためには，データのスコアおよび空間的多様性の両者を用いて目的関数をモデル化し，目的関数を最大化することによって解かれる，組合せ最適化問題を考えれば良い．

組合せ最適化問題を解くことによって多様集合を取得する既存研究は，文献 [18, 28, 31, 36] で提案されているものをはじめとして，これまでにもいくつか行われている．1.3.1 項で述べた通り，既存の多様化手法 [26, 34] 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との距離を計算する．データのスコアとデータ間の空間距離で定義される目的関数のもと，個々のデータに対する評価値も同様に，データのスコアとデータ間の空間距離の重み付き和で表される．計算したデータ間の距離とデータのスコアから，すべてのデータについて評価値を計算し，最大の評価値をとる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極めて大きい場合，計算コストも比例して大きくなる．

本章では，この問題を解決するために，事前にデータを構造化することによっ

て，短時間で多様集合を取得する手法を提案する．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，かつすべてのデータを複数回走査することで得られる多様集合と同一の検索結果を保証することを目的とする．提案手法では，類似するデータは評価値も類似することに着目し，クエリ処理の前に類似するデータをクラスタリングする．クエリ処理の際は，まず最初に各クラスタを代表するデータについてのみ評価値を計算する．クラスタの中心座標からクラスタ内のほかのデータの評価値が取りうる値の上界を計算できるため，上界がほかのクラスタの代表データの評価値より小さく，走査する必要のないクラスタを走査の対象から除外できる．

具体的に，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同士をクラスタ化し，その際にクラスタの中心として用いたデータを，そのクラスタの中心データおよび代表データとする．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の際に利用される．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取りうる評価値の上界を計算する．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取りうる評価値の上界を出来る限り小さく，短時間で計算できる．これにより，計算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，多様集合に追加すべき最適なデータを短時間で探索できる．結果として，走査するデータの数を大幅に削減しつつ，すべてのデータを複数回走査することで得られる多様集合と同一の検索結果を取得できる．また，提案手法のアプローチは，本研究で対象とする地理的な多様性だけではなく，検索結果の多様化に関する任意の最適化問題に対して同様に適用可能である．

以下では，まず2.2節で想定環境を紹介し，本章の問題を定義する．2.3節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，2.4節で提案手法について説明する．その後，2.5節でシミュレーション実験の結果を示す．その後，2.6節で本章の考察を行い，2.7節で関連研究について述べる．最後に，2.8節で本

章のまとめと課題について述べる．

2.2 想定環境と問題定義

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 検索結果の多様化を行う時空間範囲であり，図 1.2(b) および (c) における矩形領域が該当する．モバイルセンサデータベースがセンシング領域内で長期間にわたって収集されたデータを管理する場合を想定し，空間だけでなく時間を考慮して検索範囲を定義することで，ある限られた期間内に観測されたデータのみを取得できるようにする．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 $o.\boldsymbol{z}$ を保持している．$o.\boldsymbol{loc}$ は，経度 $o.x$ と緯度 $o.y$ によって表される 2 次元平面内の点とし，$o.\boldsymbol{z}$ は d 次元のベクトル $o.z_i(i = 1, ..., d)$ で表される．例えば，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 $\boldsymbol{z}$ をセンシングしデータ ID が id のデータとして生成した後，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 $\boldsymbol{z}'$ をセンシングしデータ ID が id' の別のデータとして生成したとする．このとき，データベースには 2 つのタプル $(id, t_n, \boldsymbol{loc}, \boldsymbol{z})$ および $(id', t_{n+1}, \boldsymbol{loc}', \boldsymbol{z}')$ が格納され，各カラムの値は不変で更新は行われない．ただし，データの観測時刻 $o.t_n$ は検索範囲内のデータを取得する際にのみ参照される．そのため，検索範囲内のデータを取得した後，多様集合を計算する際には，データの観測時刻を無視する．

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．

$$p(q, o) = \sum_{i=1}^{d} q.w_i \cdot o.z_i \tag{2.1}$$

式 (2.1) 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 $dist : O \times O \to R^+$ 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 $dist$ を空間距離とし，$dist(u,v)$ 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．

$$dist(u,v) = \sqrt{(u.x - v.x)^2 + (u.y - v.y)^2} \tag{2.2}$$

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検索結果の多様化問題を以下のように定義する．

定義 (Top-k 検索結果の多様化問題)**.** 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，多様集合 S^* が得られる．

$$S^* = \underset{S \subseteq O, |S|=k}{\arg\max}\, f(S, q, p(\cdot), dist(\cdot,\cdot)) \tag{2.3}$$

ここで，$f(S, q, p(\cdot), dist(\cdot,\cdot))$ は目的関数である．以降では文脈上明らかな場合は，$f(S, q, p(\cdot), dist(\cdot,\cdot))$ を $f(S)$ 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．例えば，MaxMin[34]，MaxSum[26]，Maximal Marginal Relevance（MMR）[16, 31] の目的関数は，それぞれ以下の各式で示される．

$$f_{min}(S) = \min_{u \in S} p(u) + \lambda \min_{u,v \in S} dist(u,v) \tag{2.4}$$

$$f_{sum}(S) = (k-1) \sum_{u \in S} p(u) + 2\lambda \sum_{u,v \in S} dist(u,v) \tag{2.5}$$

$$f_{mmr}(S) = (1-\lambda) \sum_{u \in S} p(u) + \lambda \min_{u,v \in S} dist(u,v) \tag{2.6}$$

いずれの目的関数についても，多様集合 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，多様集合 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度を表しており，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，最終的な多様集合は純粋な 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 [26, 31, 34]，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を総当りで探索するのは，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多様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 [26]．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．

2.3 ベースライン手法

本節では，ベースラインとなる単純なグリーディアルゴリズムについて説明する．グリーディアルゴリズムでは，すべてのデータの評価値を計算し，最大のデータを選択するという動作を繰り返すことで，多様集合を計算する．

具体的なアルゴリズムを，Algorithm 1 に示す．文献 [26, 31] において，多様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 [31] に従い，データセット内で最大のスコアをとるデータを多様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，多様集合の大きさが k となるまで，繰り返しデータを多様集合に追加する．4 行目の $d_r(\cdot, S^*)$ 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

Algorithm 1 Algorithm for the Optimization Problem

Input: Data set O, diversified set size k, importance of spatial diversity λ, weight vector $\boldsymbol{w}$

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

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

定義される．

$$d_r^{min}(o, S^*) = \min_{u \in S^*} \{\frac{1}{2}(p(o) + p(u)) + \lambda dist(o, u)\} \tag{2.7}$$

$$d_r^{sum}(o, S^*) = \sum_{u \in S^*} \{p(o) + p(u) + 2\lambda dist(o, u)\} \tag{2.8}$$

$$d_r^{mmr}(o, S^*) = \min_{u \in S^*} \{(1 - \lambda)p(o) + \lambda dist(o, u)\} \tag{2.9}$$

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

$$o^* = \arg\max_{o \in O \backslash S^*} d_r(o, S^*) = \arg\max_{o \in O \backslash S^*} f(S^* \cup o) \tag{2.10}$$

最大の評価値をとるデータ o^* は，グリーディアルゴリズムのもとで目的関数を最大化するデータであるため，以降ではデータ o^* を**最適なデータ**と呼ぶ．

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．そこで本章では，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の数を削減し，かつベースライン手法と同

一の多様集合を取得する手法を提案する．

2.4 提案手法

本節では，本章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，多様集合に含まれないすべてのデータ $O\backslash S^*$ 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 [17, 24, 38, 79] 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，多様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，クラスタの中心のデータを，そのクラスタの中心データおよび初期の代表データとする．オンラインクエリ処理では，最初にすべての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計算できる．この時，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多様集合に追加され得ないと判断できるため，走査するデータの数を削減できる．

まず，2.4.1 項において，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，2.4.2 項において，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．

2.4.1　オフライン事前クラスタリング処理

オフライン事前クラスタリングでは，最初に選んだクラスタの中心から，一定の空間距離内に位置し，かつ環境属性値ベクトルが類似するデータをクラスタのメンバとする．クエリパラメータであるλや$\boldsymbol{w}$に非依存なクラスタを生成するため，オンラインクエリ処理の際に任意のクエリパラメータに対して適用できる．このようなクラスタリングは，canopyクラスタリング[56]を空間位置と環境属性値の2つの空間に同時に適用することで実現できる．ただし，クラスタ内のデータを走査する際，同じデータの評価値を重複して計算することを避けるため，クラスタ間データを共有しないよう，canopyクラスタリングを拡張する．

クラスタ半径r_1, r_2は，中心データからの存在範囲を示すものであり，オンラインクエリ処理の際にクラスタ内のデータが取りうる評価値の上界を計算するために用いられる．小さいクラスタ半径を用いてクラスタリングすると，評価値の上界は小さくなるため，オンラインクエリ処理の際に多くのデータを走査の対象から除外できるが，クラスタの数は大きくなる．オンラインクエリ処理ではクラスタの代表データは必ず評価値を計算するため，クラスタの数の増大は計算効率を低下させる．そのため，環境属性値の半径r_2は環境属性値の誤差を考慮し，小さな誤差でクラスタが分割されないよう設定する必要がある．また，空間位置の半径r_1は空間的多様性の重要度λによって最適な値が変わると考えられるため，クエリパラメータとして用いられる値の頻度に応じて設定する必要がある．

クラスタリングのためのアルゴリズムを，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$o_i.\boldsymbol{z}$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．d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

Algorithm 2 Algorithm for Clustering

Input: Data set O, spatial radius r_1, environmental attribute radius 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
```

れる．

$$O_i = \{o \mid \sqrt{\sum_{j=1}^{d}(o_i.z_j - o.z_j)^2} \leq r_2\} \tag{2.11}$$

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，環境属性値が空間的自己相関の性質を有していたとしても，以下のような場合を想定し，環境属性値についてもデータの近接性を考慮する必要がある．検索範囲の時間幅が大きい場合，同じ位置でも物理現象の出現，あるいは消滅によって，観測時刻間で環境属性値が大きく変わることが考えられる．例えば，ある特定の位置の騒音指数について，朝の騒音指数に比べてものや人の動きが活発になる昼の騒音指数のほうが大きくなる場合が考えられる．また，観測した環境属性値に誤差が含まれる場合，互いに空間位置が近くても，環境属性値が大きく異なる可能性がある．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

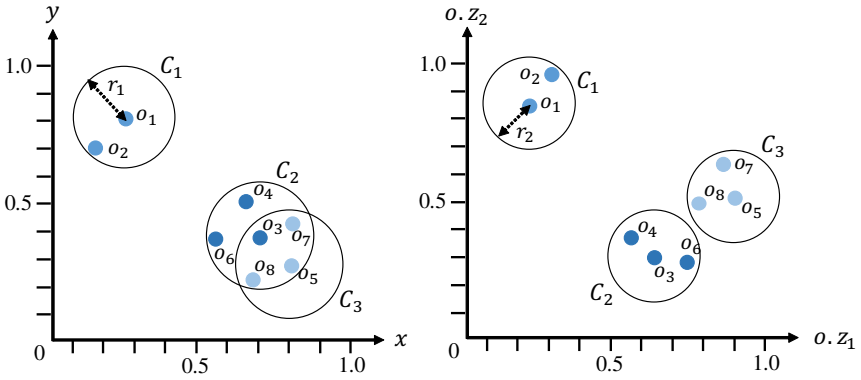
生じることでスコアが大きく異なり，互いに評価値も大きく異なる可能性がある．このような場合に，空間位置の近接性のみを考慮してクラスタリングすると，クラスタに含まれる評価値の大きなデータがたとえ少数であっても，そのようなデータによってクラスタ内のデータがすべて走査の対象として選ばれてしまう．そこで，環境属性値に関して近接性を考慮することで，このようなデータを互いに別々のクラスタに分割でき，オンラインクエリ処理の計算コストをより削減できる．

このアルゴリズムで最も計算量の大きい操作は，クラスタメンバの候補を取得する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，データが空間位置や環境属性値による多次元インデックスなどによって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$O(N^2)$となる．クエリが到着する前にオフラインで実行されることから，予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に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の候補の取得は，R*木[7, 71]やk-d木[9, 32]のような多次元インデックス構造を用いた範囲検索や，cover木[10, 39]を利用したk最近傍検索により効率化できる．

クラスタリング例

図2.1(a)および(b)に示す具体例を用いて，環境属性の次元数$d = 2$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．残りのクラスタに割り当てられていないすべてのデータは，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，

クラスタ	代表データ（中心データ）	クラスタメンバ
C_1	o_1	o_1, o_2
C_2	o_3	o_3, o_4, o_6
C_3	o_5	o_5, o_7, o_8



(a) 空間位置　　(b) 環境属性値

図 2.1: クラスタリングの例

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，クラスタに割り当てられていないデータの内，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．このように，すべてのデータはそれぞれ 3 つのクラスタの内のいずれかに割り当てられる．

2.4.2 クラスタを利用したオンラインクエリ処理

本項では，クラスタを利用したオンラインクエリ処理アルゴリズムについて説明する．提案するアルゴリズムでは，走査対象のクラスタと走査する必要のないクラスタを正しく振り分けることで，評価値を計算するデータの数を削減しつつ，ベースライン手法と同一の多様集合を計算できる．

具体的なアルゴリズムを Algorithm 3 に示し，表 2.1 に本項で用いる記号をまとめる．ここでは特に，走査するデータの数を削減する手続きを含む，Algorithm 3

表 2.1: 2.4.2 項で用いる記号のリスト

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
$\boldsymbol{C'}$	走査対象のクラスタ集合
$\overline{d_r(C_i, S^*)}$	クラスタ C_i の上界
v_i	クラスタ C_i 内の仮想データ

Algorithm 3 Algorithm for Optimization Problem Leveraging Clusters

Input: Set of clusters $\boldsymbol{C}$, diversified set size k, importance of spatial diversity λ, weight vector $\boldsymbol{w}$, spatial radius r_1, environmental attribute radius r_2

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

Initialize the set $S^* = \emptyset$
Find $o^* = \arg\max_{o \in O} p(x)$ and set $S^* = \{o^*\}$
while $|S^*| < k$ **do**
  Find o^*_{rep} such that $o^*_{rep} = \arg\max_{o_{i,rep} \in C_i} d_r(o_{i,rep}, S^*)$
  Initialize the set $\boldsymbol{C'} = \{C \mid o^*_{rep} \in C\}$
  for all $i = 1$ to $|\boldsymbol{C}|$ **do**
    Estimate upper bound of each cluster $\overline{d_r(C_i, S^*)} = \max_{v_i \in C_i} d_r(v_i, S^*)$
    if $d_r(o^*_{rep}, S^*) \leq \overline{d_r(C_i, S^*)}$ **then**
      $\boldsymbol{C'} = \boldsymbol{C'} \cup \{C_i\}$
    end if
  end for
  Find $o^* \in \boldsymbol{C'} \backslash S^*$ such that $o^* = \arg\max_{o \in \boldsymbol{C'} \backslash S^*} d_r(o, S^*)$
  Set $S^* = S^* \cup \{o^*\}$
  if o is representative data of C_i **then**
    Select new representative data for C_i
  end if
end while

の反復部分について説明する．まず，すべての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 行目）．以降，この代表データ o^*_{rep} を基準データと呼ぶ．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，本項の後半で説明するように，クラスタ内のデータが取りうる評価値の上界を計算する際に必要となるためである．基準データ o^*_{rep} を含むクラスタは，多様集合に追加すべき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のクラスタ集合 $\boldsymbol{C'}$ に追加する．次に，すべてのクラスタについて，各クラスタ内のデータの評価値が取りうる値の上界（以降では，単に**クラスタの上界**と略記）$\overline{d_r(C_i, S^*)}$ を計算する（7 行目）．各クラスタの上界と，最初に計算した基準データの評価値 $d_r(o^*_{rep}, S^*)$ を比較し，上界のほうが大きい場合，$\boldsymbol{C'}$ に追加する．ここで，以下の定理が示す通り，$\boldsymbol{C'}$ 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のクラスタ集合 $\boldsymbol{C'}$ 内に存在する．すなわち，$\arg\max_{o \in O} d_r(o, S^*) = \arg\max_{o \in \boldsymbol{C'}} d_r(o, S^*)$ 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のクラスタ集合外 $O \backslash \boldsymbol{C'}$ に存在すると仮定する．最適なデータ o^* を含むクラスタを C^* とする．データ o^* は，候補となるデータの集合の中で最大の評価値をとるため，その評価値は少なくとも基準データの評価値以上となる．ゆえに，$d_r(o^*, S^*) \geq d_r(o^*_{rep}, S^*)$ 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 $\overline{d_r(C^*, S^*)} \geq d_r(o^*, S^*)$ となる．これらの不等式および Algorithm 3 の 8 行目から 10 行目より，クラスタ C^* は走査対象のクラスタ集合 $\boldsymbol{C'}$ 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^* が走査対象のクラスタ集合外 $O \backslash \boldsymbol{C'}$ に存在するという仮定に矛盾する． □

最後に，走査対象のクラスタ集合 $\boldsymbol{C'}$ 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，多様集合 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）．空間的相関性が弱く，クラスタが細かく分割される場合でも，評価値が最大のデータを含んでいる可能性

のあるクラスタは必ず走査されるため，提案手法によりベースライン手法と同じ多様集合を取得できる．

クラスタの上界の計算

ここで，Algorithm 3 の 7 行目における，各クラスタ内のデータが取りうる評価値の上界の計算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 が取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，多様集合内のデータに非依存の項と依存する項に分解する．

$$d_r^{min}(v_i, S^*) = \frac{1}{2}p(v_i) + \min_{u \in S^*}\{\frac{1}{2}p(u) + \lambda dist(v_i, u)\} \tag{2.12}$$

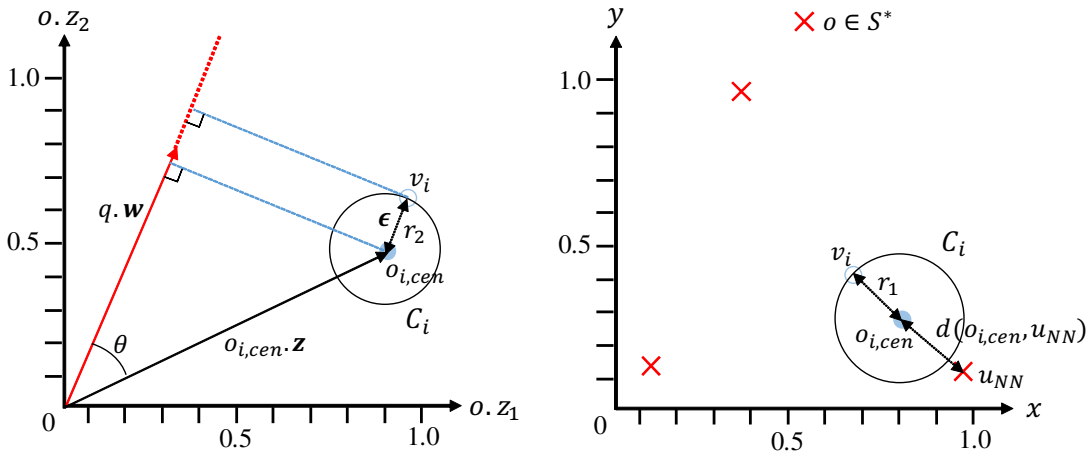
$$d_r^{sum}(v_i, S^*) = |S|p(v_i) + \sum_{u \in S^*}\{p(u) + 2\lambda dist(v_i, u)\} \tag{2.13}$$

$$d_r^{mmr}(v_i, S^*) = (1-\lambda)p(v_i) + \min_{u \in S^*}\{\lambda dist(v_i, u)\} \tag{2.14}$$

まず，多様集合内のデータに非依存の項（第 1 項）が取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 $o.\boldsymbol{z}$ の内積として捉えると，それぞれのベクトルのなす角を θ とした時，以下の式で計算できる．

$$\begin{aligned} p(q, o) &= \sum_{i=1}^{d} q.w_i \cdot o.z_i \\ &= q.\boldsymbol{w} \cdot o.\boldsymbol{z} \\ &= |q.\boldsymbol{w}||o.\boldsymbol{z}|cos\theta \end{aligned} \tag{2.15}$$

よって，クラスタ内に存在しうる仮想データ $v_i(v_i.\boldsymbol{z} = o_{i,cen}.\boldsymbol{z} + \boldsymbol{\epsilon})$ を考えた時，第



(a) 第1項の上界の計算　　(b) 第2項の上界の計算

図 2.2: クラスタの上界の計算

1 項の最大値は以下の式で与えられる（図 2.2(a)）.

$$\begin{aligned}\max_{v_i \in C_i}\{p(v_i)\} &= \max_{|\boldsymbol{\epsilon}| \leq r_2, 0 \leq \theta \leq 2\pi}\{(q.\boldsymbol{w} \cdot (o_{i,cen}.\boldsymbol{z} + \boldsymbol{\epsilon}))\} \\ &= p(o_{i,cen}) + \max_{|\boldsymbol{\epsilon}| \leq r_2, 0 \leq \theta \leq 2\pi}(|q.\boldsymbol{w}||\boldsymbol{\epsilon}|cos\theta) \\ &= p(o_{i,cen}) + |q.\boldsymbol{w}|r_2 \end{aligned} \tag{2.16}$$

次に，多様集合内のデータに依存する項（第 2 項）が取りうる最大値を計算する．MaxMin 問題および MMR 問題それぞれについて，クラスタの中心データと多様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，その時の最小の値を取る多様集合内のデータを $u_{NN} \in S^*$ とする．中心データとデータ u_{NN} を直線で結んだ時，2 つの交点が存在する（図 2.2(b)）．ここで，データ u_{NN} から最も離れる位置は，2 つの交点の内，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，多様集合 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 項が取りうる最大値は，以下の

式で与えられる．

$$\max_{v_i \in C_i}\{\min_{u \in S^*}\{\frac{1}{2}p(u) + \lambda dist(v_i, u)\}\} = \frac{1}{2}p(u_{NN}) + \lambda\{dist(o_{i,cen}, u_{NN}) + r_1\} \quad (2.17)$$

$$\max_{v_i \in C_i}\{\min_{u \in S^*}\{\lambda dist(v_i, u)\}\} = \lambda\{dist(o_{i,cen}, u_{NN}) + r_1\} \quad (2.18)$$

また，MAXSUM問題において，仮想データv_iの位置として，多様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．勿論このような，仮想データが複数の位置情報を有する仮定は成り立ち得ないが，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項が取りうる最大値は，以下の式で与えられる．

$$\max_{v_i \in C_i}\{\sum_{u \in S^*}\{p(u) + 2\lambda dist(v_i, u)\}\} = \sum_{u \in S^*}\{p(u) + 2\lambda(dist(o_{i,cen}, u) + r_1)\} \quad (2.19)$$

これらの式から，MAXMIN問題，MAXSUM問題，MMR問題におけるクラスタの上界を，それぞれ以下のように計算できる．

$$\begin{aligned}\overline{d_r^{min}(C_i, S^*)} &= \frac{1}{2}\{p(o_{i,cen}) + |q.\boldsymbol{w}|r_2\} + \frac{1}{2}p(u_{NN}) + \lambda\{dist(o_{i,cen}, u_{NN}) + r_1\} \\ &= \{\frac{1}{2}(p(o_{i,cen}) + p(u_{NN})) + \lambda dist(o_{i,cen}, u_{NN})\} + \frac{1}{2}|q.\boldsymbol{w}|r_2 + \lambda r_1 \\ &= d_r^{min}(o_{i,cen}, S^*) + \frac{1}{2}|q.\boldsymbol{w}|r_2 + \lambda r_1 \end{aligned} \quad (2.20)$$

$$\begin{aligned}\overline{d_r^{sum}(C_i, S^*)} &= |S^*|\{p(o_{i,cen}) + |q.\boldsymbol{w}|r_2\} + \sum_{u \in S^*}\{p(u) + 2\lambda(dist(o_{i,cen}, u) + r_1)\} \\ &= \{|S^*|p(o_{i,cen}) + \sum_{u \in S^*}(p(u) + 2\lambda dist(o_{i,cen}, u))\} + |S^*||q.\boldsymbol{w}|r_2 + 2|S^*|\lambda r_1 \\ &= d_r^{sum}(o_{i,cen}, S^*) + |S^*|(|q.\boldsymbol{w}|r_2 + 2\lambda r_1) \end{aligned} \quad (2.21)$$

$$\begin{aligned}\overline{d_r^{mmr}(C_i, S^*)} &= (1-\lambda)\{p(o_{i,cen}) + |q.\boldsymbol{w}|r_2\} + \lambda\{dist(o_{i,cen}, u_{NN}) + r_1\} \\ &= \{(1-\lambda)p(o_{i,cen}) + \lambda dist(o_{i,cen}, u_{NN})\} + (1-\lambda)|q.\boldsymbol{w}|r_2 + \lambda r_1 \\ &= d_r^{mmr}(o_{i,cen}, S^*) + (1-\lambda)|q.\boldsymbol{w}|r_2 + \lambda r_1 \end{aligned} \tag{2.22}$$

これらの式より，上界の計算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パラメータ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の計算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 (2.16) からクラスタ内のデータの取りうるスコアの上界が計算できるため，同様の手順で走査するデータの数を削減できる．

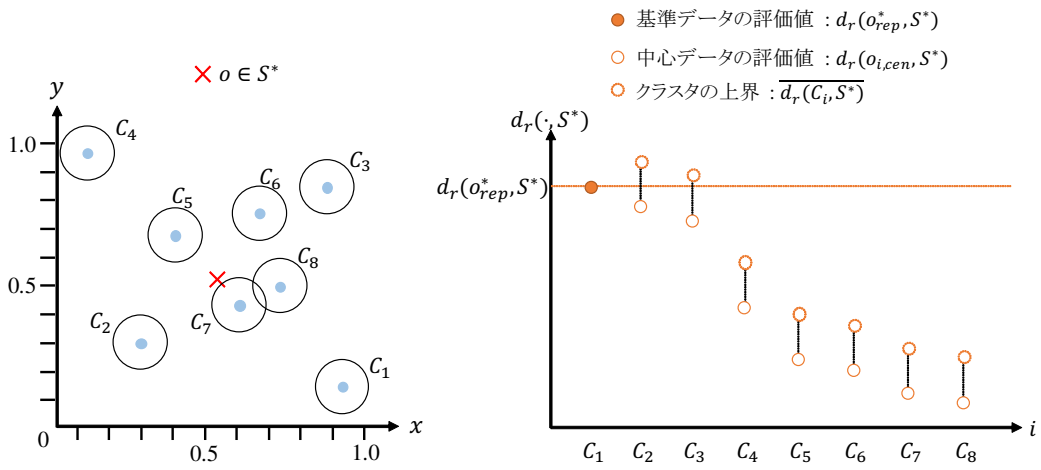
オンラインクエリ処理例

図 2.3 を用いて，走査するデータの数を削減する方法を例示する．図 2.3(a) は，初期化処理が完了し，$|S^*| = 1$ となっている状態である．基準データ 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，基準データ o^*_{rep} を含むクラスタを C_1，それ以外のクラスタの 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，クラスタの上界を計算する．

図 2.3(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，計算された上界が基準データ o^*_{rep} の評価値を下回っているため，走査対象のクラスタ集合 $\boldsymbol{C'}$ に追加されない．よって，クラスタ $C_4 \sim C_8$ 内のデータは走査する必要はない．

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．ベースライン手法は，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，以下で説明するファイル管理方法を用いることで，大部分のクラスタ内のデータをディスクから読み



(a) $|S^*|$=1におけるクラスタの空間分布　　(b) $|S^*|$=1におけるクラスタ毎の評価値の上界

図 2.3: オンラインクエリ処理の例

込むことなく多様集合が得られるため，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が走査される．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得ない．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2.4に示されるようなインデックスファイルと，関連するクラスタの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．

検索の対象の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常に効

クラスタ	代表データ（中心データ）	クラスタメンバ
C_1	o_1	o_1, o_2
C_2	o_3	o_3, o_4, o_6
C_3	o_5	o_5, o_7, o_8

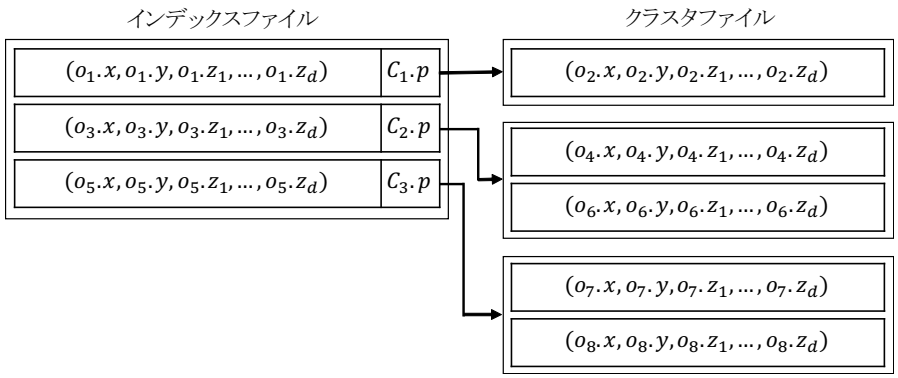


図 2.4: クラスタファイル集合

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，どのようなクエリにも提案するオンラインクエリ手法が適用できる．

計算量分析

クラスタを利用したオンラインクエリ処理の時間計算量について分析する．多様集合の計算には，はじめは空集合の S^* に最適なデータをサイズが k となるまで繰り返し追加するために，k 回のデータおよびクラスタの走査を繰り返す．この際のループ変数を i（$i = |S^*| + 1$）とする．また，クラスタの数を c とする．ここで，1 回あたりの反復における計算量を考える．まず，走査対象のクラスタ集合 $\boldsymbol{C'}$ を決定するために，すべてのクラスタの代表データの評価値の計算，およびクラスタの上界の計算が必要となる．データの評価値の計算には，1 回のデータのスコアの計算に加え，$|S^*| = i - 1$ 個のデータとの距離の計算が必要となる．よって，c 個の代表データの評価値の計算の計算量は，$c \cdot \{1 + (i-1)\} = ci$ となる．さらに，基準データの評価値を上界が下回るクラスタを除外するために，c 個のクラスタの走査が必要となる．よって，走査対象のクラスタ集合 $\boldsymbol{C'}$ の決定の計算量はこれら

の合計であり，$ci + c = c(i + 1)$ となる．最後に走査対象のクラスタ集合 $\boldsymbol{C'}$ 内のすべてのデータの評価値の計算が必要となる．走査対象のデータの数を n_i^* とすると，計算量は $n_i^* \cdot \{1 + (i - 1)\} = n_i^* i$ となる．また，各反復における走査対象のデータの数の最大値を，$\max_{1 \leq i \leq k}\{n_i^*\} = \overline{n^*}$ とする．オンラインクエリ処理の全体の計算量は，k 回の反復の合計であるため，次式で表される．

$$\begin{aligned}\sum_{i=1}^{k}\{c(i+1) + n_i^* i\} &\leq (c + \overline{n^*})\sum_{i=1}^{k} i + \sum_{i=1}^{k} c \\ &= \frac{1}{2}(c + \overline{n^*})k(k+1) + ck \\ &= \frac{1}{2}k^2(c + \overline{n^*}) + \frac{1}{2}k(c + \overline{n^*}) + kc \end{aligned} \tag{2.23}$$

よって，オンラインクエリ処理の計算量は，$O(k^2(c + \overline{n^*}))$ である．また，2.3 節の末尾で述べた通り，データセットサイズを N としたとき，ベースライン手法の計算量は $O(k^2 N)$ である．多くの場合，クラスタの数 c や各反復において走査されるデータの数の最大値 $\overline{n^*}$ は，N に比べ大幅に小さい．このため，提案手法の計算量は，ベースライン手法の計算量に比べて小さくなる．

2.5　性能評価

Top-k 検索結果の多様化における，提案手法の性能を評価する．表 2.2 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，クエリが到着する前に一度だけ実行されればよく，データを公開する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．そこで本節における評価では，オンラインクエリ処理に伴う計算時間およびディスク IO コストを求めた．

表 2.2: パラメータの値

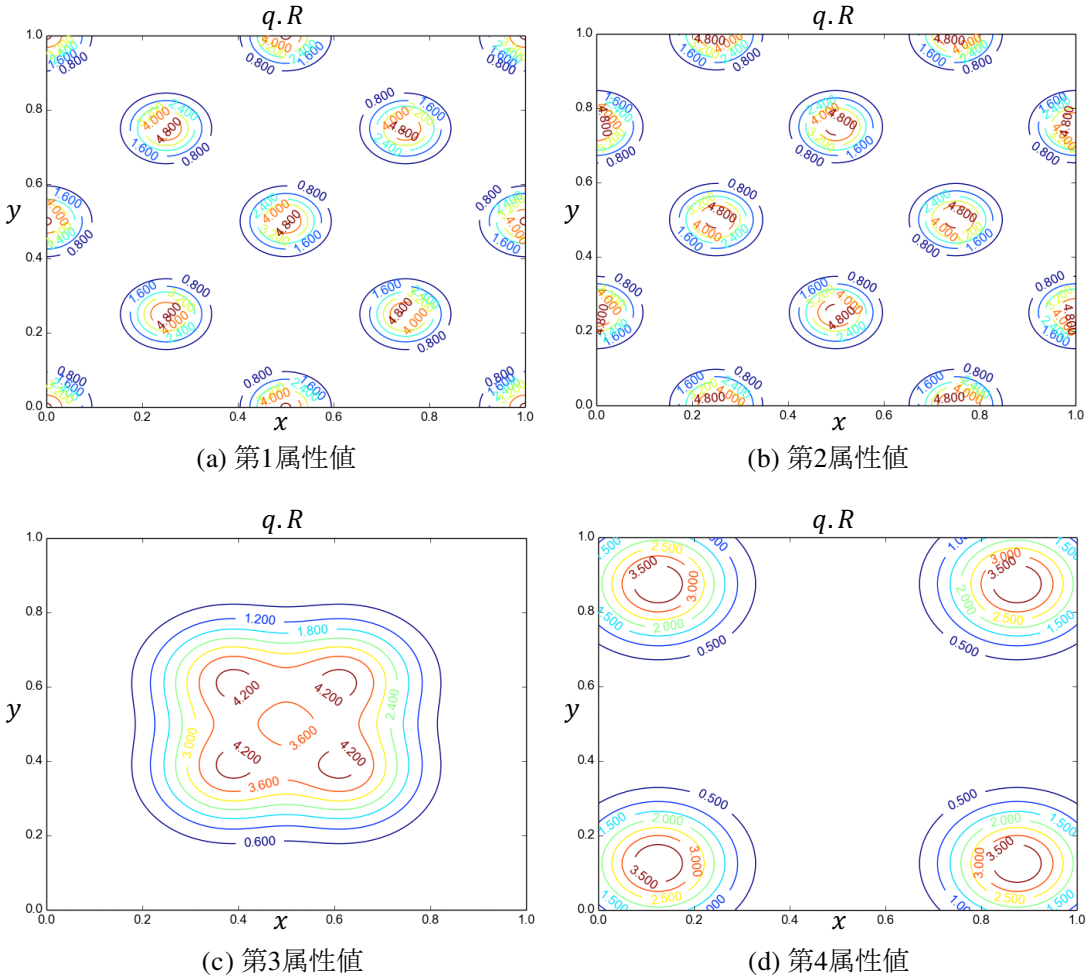
パラメータ	値
データセットサイズ N	**1M**, 5M, 10M, 50M
要求するデータの数 k	5, 10, **15**, 20, 25, 30
クラスタの空間半径 r_1	0.01∼0.15
クラスタの環境属性値半径 r_2	0.1∼2.0
$\boldsymbol{w}$ の各要素	0.0∼1.0
λ (MAXMIN, MAXSUM)	0.0∼5.0
λ (MMR)	0.0∼1.0
環境属性の次元数 d	1, **2**, 3, 4

2.5.1 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 2.5 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．

2.5.2 比較手法

提案手法（以降のグラフ中では ‘P-cluster’ と表記）を，2.3 節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 (2.16) においてクラスタ内のデータのスコアについて上界を計算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，クラスタ毎に最大のスコアとそのデータを記憶する．



(a) 第1属性値　(b) 第2属性値

(c) 第3属性値　(d) 第4属性値

図 2.5: 環境属性値の分布

以降，評価値の上界の計算にはこの最大のスコアを用いることで，他の手法と同一の多様集合が取得できる．

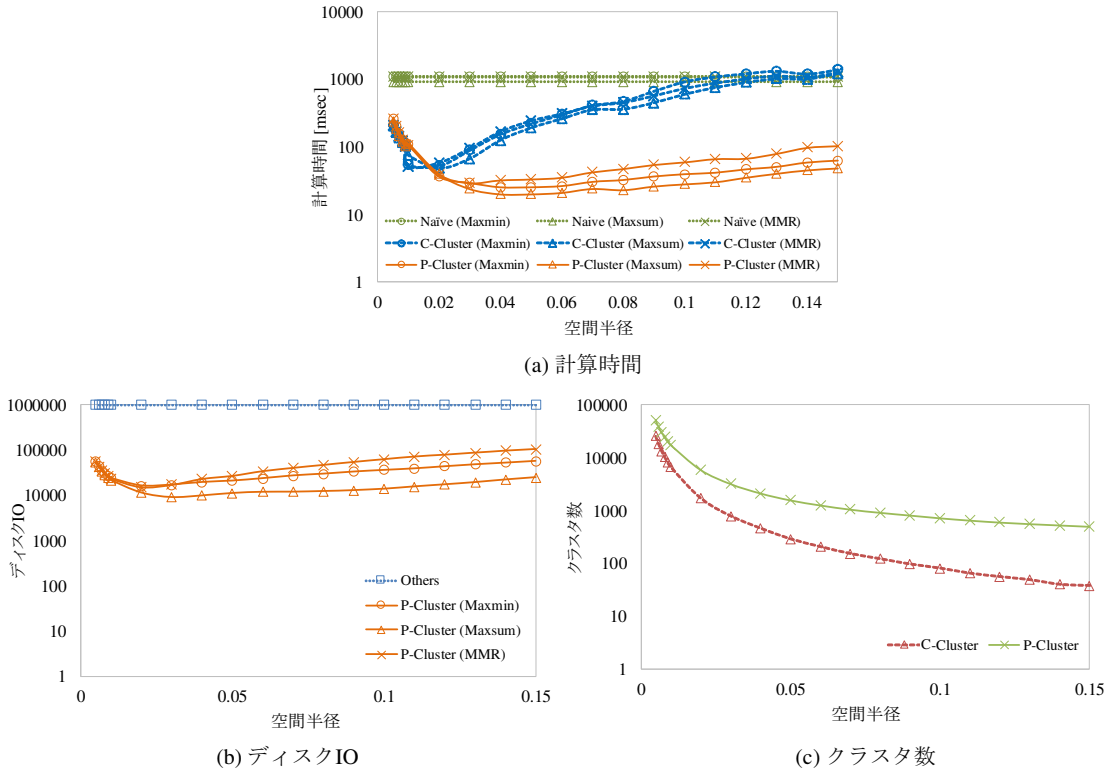
2.5.3　設定

すべてのアルゴリズムを Java7 で実装し，Intel(R) Core(TM) i7-4790K CPU @ 4.00GHz 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の数としてディスク IO コストを示す．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，常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の数を示す．

実験で用いたクエリは，$q.\boldsymbol{w}$ と $q.\lambda$ がそれぞれ表 2.2 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 [43, 73] 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 [34] 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 (2.4),(2.5)）では，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 2.5 に示す各ホットスポットの中心で最大 5.0 程度の値を取る．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，ホットスポットの検出を目的とした 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 [16, 31] 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，クラスタの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の数をそのまま示している．



(a) 計算時間

(b) ディスクIO

(c) クラスタ数

図 2.6: 空間半径 r_1 の影響

2.5.4　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 2.6(a) に，ディスクIO コストを図 2.6(b) に，クラスタの数を図 2.6(c) に示す．図 2.6(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わかる．これは，クラスタ内のデータが広い地理範囲に存在しうるため，クラスタ内のデータの評価値の上界を過大に計算しており，走査対象から除外できたクラスタの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の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わかる．これは，図 2.6(c) に

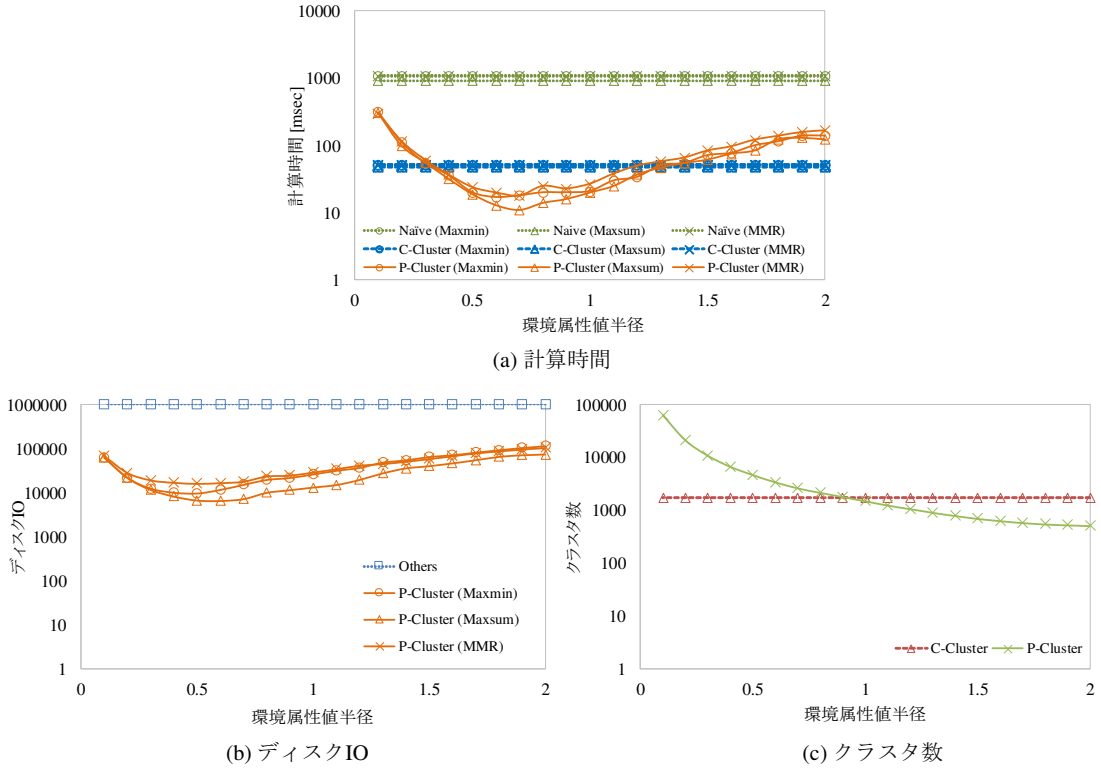
示されるように，空間半径が小さくなると生成されるクラスタの数が増加することによる．走査対象のクラスタの数を削減できても，クラスタの上界を計算するためにすべてのクラスタの中心データを走査しなければならず，結果として全体の計算時間は長くなってしまう場合がある．

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 コストを最高で 100 分の 1 に削減できている点が挙げられる．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．

2.5.5 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 2.7(a) に，ディスク IO コストを図 2.7(b) に，クラスタ数を図 2.7(c) に示す．図 2.7(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わかる．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計算してしまい，走査対象から除外できたクラスタの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わかる．これも空間半径を小さくした場合と同様，図 2.7(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の数が増加し，走査が必要なデータの数も増加してしまうためである．



(a) 計算時間

(b) ディスクIO

(c) クラスタ数

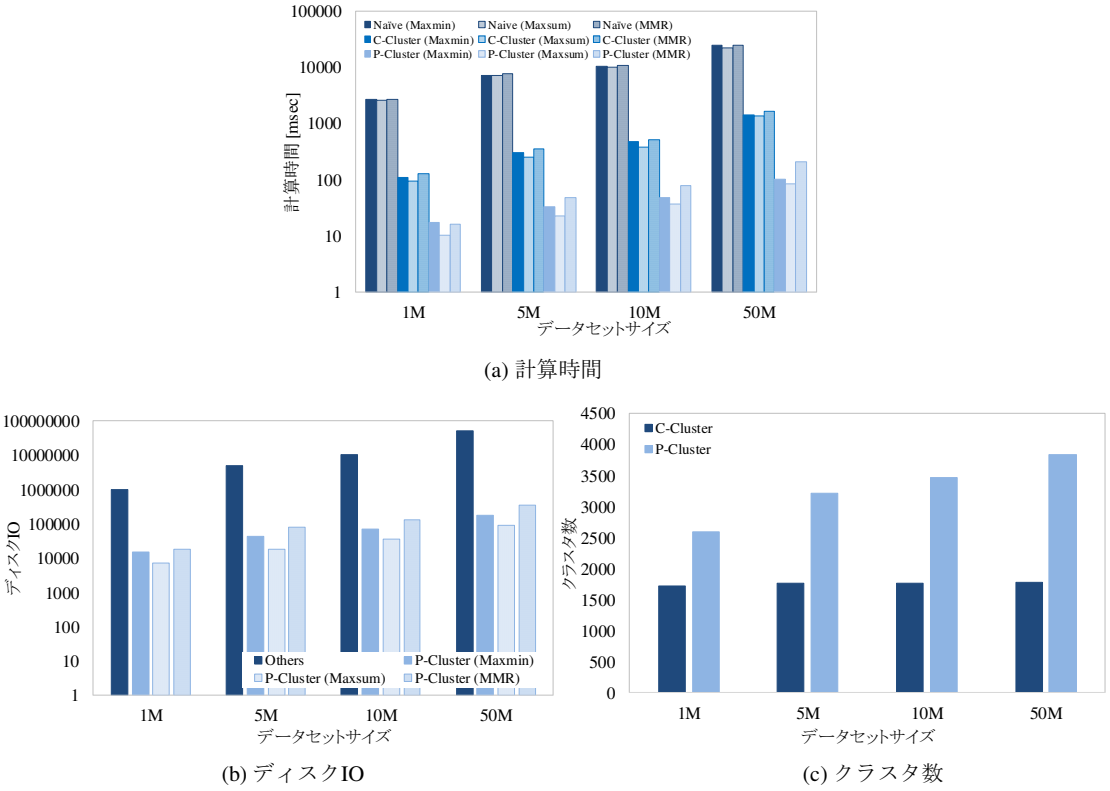
図 2.7: 環境属性値半径 r_2 の影響

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．

2.5.6　その他のパラメータの影響

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得ない．そのため，検索範囲内に存在するデータの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 2.8(a) に，ディスク IO コストを図 2.8(b) に，クラスタの数を 2.8(c) に示す．いずれのデータセットサイズの場合も，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 2.8(c) は，比較手法におけるクラスタの数が，データセットサイズ N が変化し



(a) 計算時間

(b) ディスクIO

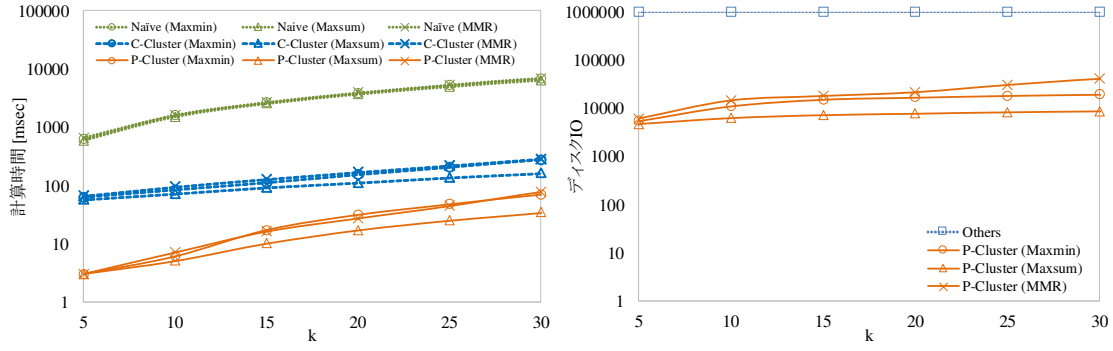
(c) クラスタ数

図 2.8: データセットサイズ N の影響

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数の平均値が大きくなるため，アルゴリズムにおける走査対象のデータの数を削減する効果が小さくなることがわかる．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の数も増加し，走査対象のデータの数を削減するための計算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．

要求するデータの数 k の影響

ユーザごとに，要求するデータの数は異なる．そこで，要求するデータの数 k を変化させた場合の，計算時間を図 2.9(a) に，ディスク IO コストを図 2.9(b) に示



(a) 計算時間　　(b) ディスクIO

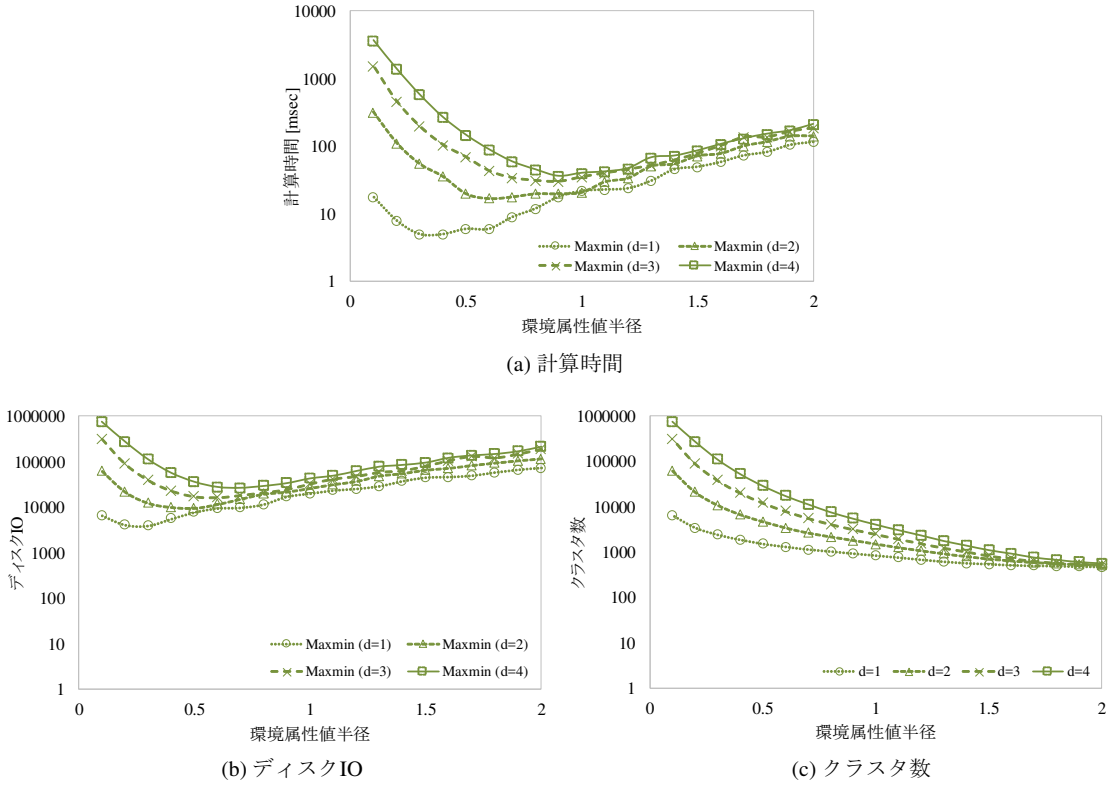
図 2.9: 要求するデータの数 k の影響

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，走査対象のデータの数を削減する効果は大きく働く．ホットスポットにおけるデータを探索する過程においては，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．

環境属性の次元数 d 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 d 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 2.10(a) に，ディスク IO コストを図 2.10(b) に，クラスタの数を図 2.10(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 2.10(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同士が類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の数が急激に増加し，性能が低下してしまう．よって，以降の実験では，各次元数 d 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 d を変化させた場合の，計算時間を図 2.11(a) に，ディ



(a) 計算時間

(b) ディスクIO

(c) クラスタ数

図 2.10: 環境属性の次元数 d と環境属性値半径 r_2 の影響

スク IO コストを図 2.11(b) に，クラスタの数を図 2.11(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わかる．

2.6 考察

2.6.1 オフライン事前クラスタリング処理の計算時間について

本項では，オフライン事前クラスタリング処理に要する計算時間について考察する．本研究では，2.4.1 項で説明した最悪計算量が $O(N^2)$ の Algorithm 2 を，空間位置や環境属性値による多次元インデックスを用いずに実行している．図 2.12 に，表 2.2 のデフォルトのパラメータのもとで環境属性値半径 r_2 を変化させた場

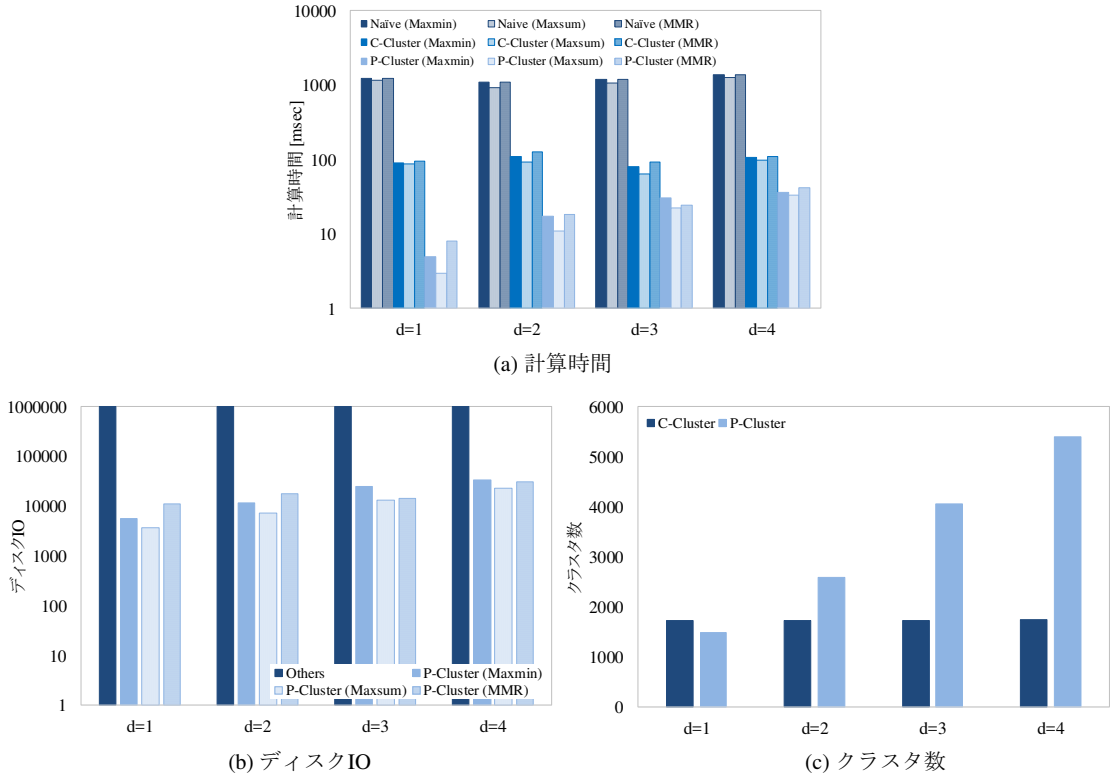


図 2.11: 環境属性の次元数 d の影響

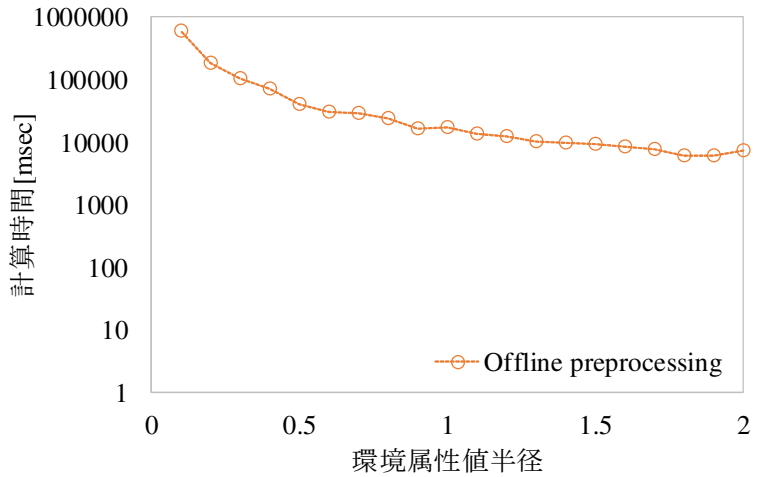


図 2.12: クラスタリングの計算時間

合の，クラスタリングに要した計算時間を示す．図 2.12 より，計算時間はクラスタリング半径が小さくなるほど大きくなっていることが分かる．これは，クラスタ半径が小さい場合 1 つのクラスタに含まれるデータの数が小さく，多くのデータがクラスタに割り当てられない状態で残るため，以降のクラスタの生成のために必要なデータ間の距離を計算する回数が大きいためである．図 2.7(a) より，提案手法のオンラインクエリ処理時間は r_2 が 0.7 のとき約 0.03 秒であり，このときのクラスタリング処理時間は約 28 秒である．一方，ベースライン手法ではオンラインクエリ処理時間は約 1 秒であるため，29 回以上クエリを処理することで，全体の計算時間は提案手法のほうが小さくなる．

本研究では，データを公開する前にオフライン事前クラスタリング処理のための十分な時間が確保できると想定している．また，クエリパラメータを逐一変化させながら広大なセンシング領域を調査する場合を想定し，ユーザビリティ向上のためには，ユーザに即座に検索結果を返すことが重要であると考え，オンラインクエリ処理の高速化に主眼を置いている．しかし，データのクラスタリングを完了させユーザに提供できるまでの時間は短いほうが望ましい．オフライン事前クラスタリング処理の計算コストを小さくするためには，2.4.1 項で述べたように，空間位置や環境属性値による多次元インデックスを用いる方法が考えられる．また，データ公開時にはクラスタリングに用いる半径を大きく設定し，その後システムの裏側でよりクエリ処理に適した小さいクラスタ半径を用いてクラスタリングし，完了した段階で新しいクラスタに差し替える方法なども考えられる．

2.6.2 検索エンジンにおける Top-k 検索結果の多様化との相違点について

検索エンジンにおける多様化とモバイルセンサデータベースにおける多様化では，幾つか前提が異なっている点があるため，分野間の相違点について説明する．

文献 [16, 26, 34] は，検索エンジンにおける Top-k 検索結果の多様化を対象としている．これらの既存研究では，まず最初にユーザがクエリキーワードを指定し，キーワードに関連する文書の集合を取得する．その後，取得した文書のうち関連度

が上位の文書集合内で多様化を行い，文書間の類似度が小さくなるような多様集合を計算する．これらの既存研究における多様化の対象となる文書の集合は，ユーザのクエリキーワードに対して一定の関連度を示す文書である．そのため，多様集合の計算の際に文書のスコア，すなわち関連度は考慮せず，しばしば目的関数を文書間の非類似度の最小値（MaxMin問題）や和（MaxSum問題）のみで定義している．ただし，文献 [5, 31] のように，文書間の非類似度に加えて文書の関連度も目的関数に含まれ，また実際の検索のように関連度順にソートされたリストが利用可能であると仮定している研究もある．

次に，本研究で対象とするモバイルセンサデータベースにおけるTop-k検索結果の多様化では，まず最初にユーザが指定する時空間の検索範囲内に存在するセンサデータの集合を取得する．その後，取得したセンサデータの集合内で，環境属性値から算出されるデータのスコアと，位置情報から算出されるデータ間の空間距離を考慮し，高スコアを示し，かつ地理的に分散するデータからなる多様集合を計算する．目的関数がデータのスコアとデータ間の空間距離で定義される点において，先述した文献 [5, 31] における多様化問題と類似しているが，時空間範囲検索の結果を取得した段階では，データのスコアは計算されていない点が異なっている．そこで，時空間範囲検索でデータ集合を取得した後，データのスコアを計算することで，文献 [5, 31] における既存研究と，入力となるデータセットについての前提が類似する．一方で，検索範囲内のデータのスコアを計算し，スコアが上位のデータのみを対象として，データ間の空間距離のみを考慮した多様化問題を考えることで，文献 [16, 26, 34] における既存研究の前提に近づけられる．本章における提案手法では，クエリパラメータを $\lambda = 0$ と設定することで，データ間の空間距離を無視し，データのスコアのみを考慮したTop-k検索についても効率的に計算できるため，文献 [16, 26, 34] の多様化問題についても踏襲できる．しかし，このような多様化問題では，スコアが上位のデータを取得するために，ユーザは新たに閾値を設定する必要があり，また試行錯誤的に閾値を変化させながら検索を繰り返す必要があると考えられる．本研究では，ユーザの指定する検索範囲内で，ユーザの興味に基づいたデータのスコアが高く，かつ地理的に分散するデータ集合を取得するクエリを直感的に定義できることを目指し，文献 [5, 31] に

おける既存研究を参考とし，多様化問題を定義している．

2.6.3 クラスタリング手法について

本研究では，グリーディアルゴリズムで得られる多様集合を効率的に計算することを目的として，センサデータをクラスタリングする．提案手法におけるオフライン事前クラスタリング処理では，クラスタ間でデータを共有しないように拡張した canopy クラスタリング [56] を，空間位置と環境属性値の 2 つの空間に同時に適用することで，クラスタを生成している．

一方，k-means 法 [44] や凝集型クラスタリング [8] をはじめとして，データマイニングを目的としたクラスタリング手法は，これまでに数多く提案されている．しかし，提案アルゴリズムでは任意のクエリパラメータのもとでクラスタの上界を計算する必要があることを考慮すると，このようなデータマイニングを目的としたクラスタリング手法は，提案アルゴリズムには適さないと考えられる．例えば，k-means 法や凝集型クラスタリングでは，データの分布に依存してクラスタが生成されるため，生成されるクラスタの形状を制御することは困難である．このようなクラスタリングのもとで任意の λ や $\boldsymbol{w}$ に対応するためには，生成されたクラスタに対し中心座標，および中心座標から最も遠いデータの座標との間の距離を事前に計算する必要がある．しかし，クラスタの形状によってはこのような中心から最遠点までの距離が大きく，クラスタの上界が大きくなる場合が考えられる．クラスタの上界が大きい場合，走査対象から除外される可能性が低くなるため，計算コストが削減されない可能性が高くなる．

一方，提案手法のようにクラスタ半径を事前に設定した上でクラスタリングすることで，クラスタの上界は制御できる．本研究では，システム管理者の観点から，クラスタ半径によって提案アルゴリズムの計算コストを制御できる手法が望ましいと考え，オフライン事前クラスタリング手法を提案している．

2.6.4　実環境への適用可能性について

本章の性能評価では，大規模なデータセットに対して提案手法が有効に機能するかを調べるため，人工データを用いたシミュレーション実験を行った．実験では環境属性値の空間的自己相関の性質を仮定し，ホットスポットの中心で最も環境属性値が大きく，中心から離れるほど環境属性値が小さく減衰するような分布のもとで，人工データは生成されている．使用した人工データのように空間的自己相関の性質が仮定される場合，クラスタの数が過剰に大きくなることはなく，走査対象のクラスタを絞り込むための計算コストは抑えられる．また，ホットスポットが存在する場合は，ホットスポット付近のクラスタのみが走査対象となるため，短時間で多様集合を計算できる．

しかし，実環境では地理空間上に建物などの遮蔽物が存在するため，環境属性値の分布はより複雑になると考えられる．多くの環境情報は空間的自己相関の性質を有すると考えられるが，遮蔽物の影響などで相関性の性質が弱くなる場合，クラスタの数の増加による性能の低下を防ぐためにクラスタ半径を大きくする必要がある．また，検索範囲内のホットスポットの減衰の度合いが小さく，中心から距離が離れるにつれてなだらかに環境属性値が小さくなるような分布では，走査対象のデータの数が大きくなり，性能が低下する場合が考えられる．

このように，シミュレーションでは考慮されていない事項が，提案手法の性能に影響を与える可能性がある．このため，より実環境に近い環境を再現したシミュレーション実験や，実データを用いた実験については，今後検討すべき課題である．

2.7　関連研究

センシング領域の中からホットスポットを検出するための手段として，集約クエリの利用が有効であると考えられる．集約クエリによって検索範囲の環境属性値を集約することで，その範囲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．このため，例えば検索範囲の平均値から，環境属性値が極端な値を示す領域を発見できる可能性がある．文献 [19, 49, 51, 65, 70, 77] では，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，検索範囲が大きい場合，集約によって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が難しくなることが予想される．また，検索範囲を小さくすることで，平均値はホットスポットを検出する上でより信頼できる値となるが，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得ない．このため，検索範囲に対してセンシング領域が広大である場合，ホットスポットを捉えるために繰り返し検索せざるを得ない恐れがある．そこで本論文では，この問題に対処するため，Top-k 検索結果の多様化を環境モニタリングを目的としてモバイルセンサデータベースに適用する．

組合せ最適化問題を解くことによって多様集合を取得する方法について，いくつか研究が行われている [5, 18, 28, 31, 36]．多くの既存手法では，キーワード検索によって得られるデータセットの多様化を対象としているため，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である状態を仮定している．さらに，文献 [5, 31] では，データのスコアによってソートされたデータリスト，および任意の空間位置からの距離によってソートされたデータリストが利用可能であると仮定している．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，ソートしなければならない．モバイルセンサデータベースのように大規模なデータセットに対するこのような前処理のコストは，多様集合の計算以上に計算負荷が大きくなってしまう．そこで，本章の提案手法では，オンラインクエリ処理に要する計算コストを削減するために，クエリが到着する前に任意のクエリパラメータに対応可能なデータの構造化をオフライン処理として実行する．

2.8 むすび

2.8.1 発展：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，クラスタの中心データの観測時刻付近で生成されたデータが，新たにサーバにアップロードされる場合がある．例えば，データの送受信は電力消費が激しいため，携帯端末がセンサデータを取得し

Algorithm 4 Algorithm for Maintenance of Clusters

Input: Set of clusters $\boldsymbol{C}$, spatial radius r_1, environmental attribute radius r_2, new data item 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**

たとしても直ちにサーバにアップロードせず，電源が確保できた後アップロードする場合などが考えられる．このような遅延を伴って到着した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，新たにアップロードされたデータを o_{new} とする．クラスタのメンテナンスのアルゴリズムを，Algorithm 4 に示す．2.4.1 項冒頭で述べた通り，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良い．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の集合を対象として，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）．

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，新たにアップロードされたデータの多く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，以降にアップロードされるデータの多くはそれらの新しいクラスタに組み込まれる．

2.8.2 まとめ

本章では，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，多様集合のサイズが k になるまで最適なデータを多様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同士を事前にクラスタリングする．評価値の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，走査するデータの数を大幅に削減できる．

シミュレーション実験により，提案手法はすべてのデータを走査する単純なグリーディアルゴリズムと比較して，同一の多様集合をより短時間で取得できることを確認した．さらに，要求するデータの数 k やデータセットサイズ N を変化させた場合も，提案手法は比較手法に比べ，計算時間およびディスク IO コストを削減できることを確認した．特に，データセットサイズが大きくなっても短時間で多様集合を計算できるため，センシングに参加する端末保持者が増加した場合も，効率的に Top-k 検索結果の多様化を行える．

一方，2.5.6 項における評価結果より，環境属性の次元数 d が大きい場合には，クラスタの数が大幅に増加することで提案手法の性能が低下してしまうという問題が生じることが分かった．このような，クラスタの増加による計算効率の低下の影響を緩和する手法については，第 3 章で取り組む．また，本章では過去に収集されて蓄積されたモバイルセンサデータベースにおけるスナップショットクエリを想定しており，継続的な Top-k 検索結果の多様化を行う場合は考慮していない．時々刻々とモバイルセンサデータが到着するモバイルセンサストリーム環境において，継続的な Top-k 検索結果の多様化を行うために提案手法を用いた場合，クラスタ構造をオンラインで更新する必要があり，クラスタの数が大きい場合に計算時間が増加してしまうという問題がある．この問題を解決するためには，到着するデータによる更新が短時間で可能な別のデータ構造によってモバイルセンサデータを管理する必要がある．第 4 章では，この問題について取り組む．

また，2.5.4項および2.5.5項における評価結果より，提案手法はクラスタ半径によって性能が大きく左右される．重みベクトルや空間的多様性の重要度といったクエリパラメータや，環境属性値の分布によって最適値は変化すると考えられる．そのため今後は，クエリ処理の統計情報からクラスタ半径を自動的に設定する方法などについて検討する予定である．

第3章　階層的クラスタリングを用いたTop-k検索結果の多様化手法

3.1　まえがき

2.5.6 項における環境属性の次元数 d を変化させた場合の評価結果より，第 2 章におけるクラスタベースの手法では，モバイルセンサデータの環境情報の次元数が大きくなると，計算時間が増大することがわかった．これは，環境属性値ベクトルが高次元になるほどデータ間の類似度が小さくなり，球形クラスタの半径が大きく，かつクラスタの数が大幅に増加することで，走査対象のデータの数を削減するための計算コストが大きくなってしまうためである．近年，様々な環境情報が参加型センシングによって観測，収集されているため，環境属性値ベクトルの次元数が大きい場合でも検索結果をより短時間で取得できることが望ましい．

そこで本章では，階層的クラスタリングを行うことで，環境属性値ベクトルの次元数の増加にともなう計算効率の低下の影響を緩和する手法を提案する．提案手法の階層的クラスタリングでは，空間的に近接するすべてのデータはある 1 つの上位クラスタのメンバとし，上位クラスタの配下でメンバデータの環境属性値の類似度に基いてさらに下位クラスタに分割する．提案手法のオンラインクエリ処理では，階層クラスタ構造を利用し，上位クラスタ内のデータが取りうる評価値の上界を計算することで，下位クラスタをまとめて走査対象から除外できる．空間的に近接するデータからなる上位クラスタの数は，環境属性数の増加に非依存であり，第 2 章におけるクラスタベースの手法におけるクラスタの数よりも少なくなる．そのため，提案手法ではクラスタベースの手法と比べて，走査対象のデータを短時間で絞り込める．

以下では，まず 3.2 節で関連研究について述べる．3.3 節で提案手法について説

明した後，3.4 節でシミュレーション実験の結果を示す．最後に，3.5 節で本章のまとめと課題について述べる．

3.2　関連研究

データの次元の増加に伴うデータ間の類似度の低下は，クラスタリングを始めとして多くのデータマイニング手法に対して，性能の低下を引き起こす．このため，主成分分析 [25] やランダムプロジェクション [11] といった手法によって，情報損失量をできる限り小さくしつつ特徴の数を削減する，次元削減がしばしば行われる．また，文献 [91] では，高次元空間における Top-k 検索を効率的に行うために，次元削減を行った低次元空間におけるデータセット（サブスペース）を複数保持し，クエリごとに適切なサブスペースの組み合わせを用いて上位 k 個のデータを計算する手法が提案されている．さらに文献 [91] では，次元削減によってデータセットが変化することから，得られる結果は近似解であるものの，誤差範囲の制御が可能であることが示されている．一方，本研究で対象とする多様化問題は NP 困難であることからグリーディアルゴリズムによって多様集合を計算するため，誤差範囲は計算できない．また，主成分分析のような次元削減を施した後のデータセットに対して，直接グリーディアルゴリズムを適用し多様集合を計算した場合，ユーザの興味や意図に反した予期せぬ結果が返される場合がある．そのため，次元削減を行わずに計算効率の低下の影響を緩和する，別の手法が必要となる．

3.3　提案手法

本節では，本章の提案手法について説明する．モバイルセンサデータのデータモデルやシステムモデル，および Top-k 検索結果の多様化についての問題定義は，2.2 節で述べたものと同じである．モバイルセンサデータの環境属性値ベクトルは最大で 20 次元程度を想定するが，本節の提案手法の説明では，簡単のために 2 次元の場合を想定している．

第 2 章のクラスタベースの手法における，クラスタの数の増加による計算効率の低下は，以下の 2 つの処理の計算コストの増加に起因する．

- クラスタの代表データを走査し，基準データを決定する．
- クラスタ毎に評価値の上界を計算し，走査対象のクラスタ集合を決定する．

この 2 つの処理は，走査するデータの数を削減するために行う追加の処理であるが，クラスタの増加とともに計算コストが大きくなる．

そこで，提案手法ではこれらの追加の処理の計算コストを削減しつつ，最終的に走査するデータの数を削減することを目指す．提案手法におけるオフライン事前階層的クラスタリング処理では，まず最初にデータの位置情報のみを考慮し，空間的に近接するデータをある 1 つの上位クラスタとしてまとめる．第 2 章におけるクラスタベースの手法では，位置情報に加えて環境属性値ベクトルの類似度も考慮してクラスタが生成されるため，環境属性値ベクトルの次元数が大きくなるほどクラスタの数も大きくなる．一方で，提案手法における上位クラスタは位置情報のみを考慮して生成されるため，上位クラスタの数は第 2 章のクラスタベースの手法におけるクラスタの数と比べて小さくなる．また，位置情報の次元数は環境属性値ベクトルの次元数によらず一定であるため，環境属性値ベクトルの次元数が大きくなっても上位クラスタの数は一定である．上位クラスタを形成した後，各上位クラスタの配下で，環境属性値ベクトルの類似度に基づいたクラスタリングを行い，下位クラスタに分割する．そして，提案手法におけるオンラインクエリ処理では，上位クラスタの代表データの中から基準データを決定し，上位クラスタ毎に走査対象のデータの絞り込みを行う．次に，走査対象として残った上位クラスタについて，その配下の下位クラスタに対し，さらに走査対象の絞り込みを行う．このように，上位クラスタを走査対象から除外できると，除外された上位クラスタの配下の下位クラスタを走査する必要はないため，短時間で走査対象のデータを絞り込める可能性がある．中心データのスコアが小さく，多様集合内のデータに近接する上位クラスタは，走査対象から除外される可能性が高く，短時間での走査対象のデータの絞り込みが見込める．また，上位クラスタの配下の下位クラスタの数が大きい場合ほど，このような走査対象のデータを 2 段階で

絞り込むことの効果は大きくなる．

まず，3.3.1 項において，オンラインクエリ処理で利用する階層クラスタを作成するためのオフライン事前階層的クラスタリング処理について説明する．次に，3.3.2 項において，作成された階層クラスタ構造を利用したオンラインクエリ処理について説明する．

3.3.1　オフライン事前階層的クラスタリング処理

階層的なクラスタを構築するために，まず最初にいずれのクラスタにも属していないデータについて空間的に近接するデータ集合を取得し，これらをすべて上位クラスタのメンバとする．次に，上位クラスタ内のあるメンバに着目したとき，環境属性値ベクトルが類似しているメンバを下位クラスタのメンバとする．上位クラスタのメンバがすべて，いずれかの下位クラスタに割り当てられるまで，上述した処理を繰り返すことで，階層クラスタが構築される．また，クラスタ半径 r_1, r_2 は第 2 章と同様な方針で設定する．

具体的な階層的クラスリングアルゴリズムを，Algorithm 5 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 o_i を見つけた場合，o_i の空間位置ベクトルを中心とした上位クラスタを生成する．5 行目で，全体のデータ集合 O から，上位クラスタの中心データ o_i の空間位置ベクトルを中心とした半径 r_1 の円内に存在するデータを取得する．ここで，retrieveNeighbors(o_i, r_1, ∞, O) は，データ集合 O から，データ o_i に空間位置が互いに近いデータを返す操作である．取得したデータのうち，ラベルが付与されていないデータはすべてこの上位クラスタのメンバであり，これらのデータをさらに環境属性値に基づきクラスタリングする．8，9 行目で，上位クラスタのメンバのうち，いずれのクラスタにも属していないデータ o_{ij} を見つけた場合，そのデータを新たな下位クラスタの中心データかつ代表データとする．10 行目で retrieveNeighbors(o_{ij}, ∞, r_2, X) により，上位クラスタのメンバから，データ o_{ij} 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取得する．取得したデータのうち，ラベルが付与されていないデータに対し，現在作成中の上位クラスタおよび下位クラスタのラベルを付与する．こ

Algorithm 5 Algorithm for Hierarchical Clustering

Input: Data set O, spatial radius r_1, environmental attribute radius r_2
Output: Set of upper clusters $\boldsymbol{UC} = UC_1, UC_2, ..., UC_l$

```
upperClusterLabel = 1
for i = 1 to |O| do
  if o_i is not in any clusters then
    Mark o_i as the center of the current upper cluster
    X = retrieveNeighbors(o_i, r_1, ∞, O)
    for j = 1 to |X| do
      lowerClusterLabel = 1
      if o_ij is not in any clusters then
        Mark o_ij as the center and initial representative of the current lower-level cluster
        X' = retrieveNeighbors(o_ij, ∞, r_2, X)
        for all o ∈ X' do
          if o is not in any clusters then
            Mark o with current lowerClusterLabel and upperClusterLabel
          end if
        end for
      end if
      lowerClusterLabel++
    end for
    upperClusterLabel++
  end if
end for
```

こで，上位クラスタ UC_i が m 個の下位クラスタを配下に含むとき，上位クラスタおよび下位クラスタの関係を以下のように表現する．

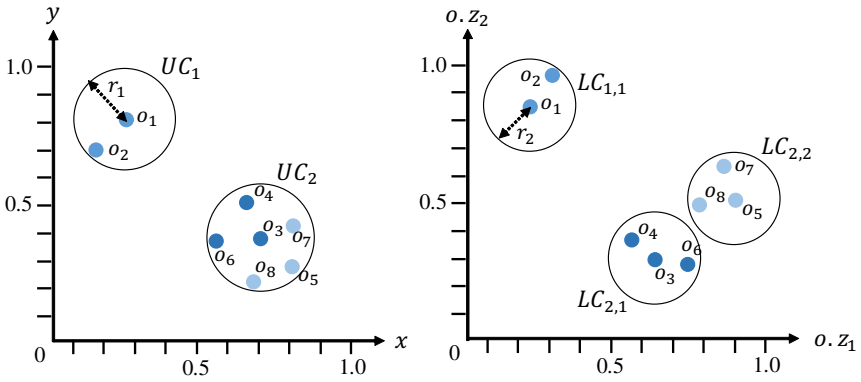
$$UC_i = \boldsymbol{LC}_i = \{LC_{i1}, LC_{i2}, ..., LC_{im}\} \tag{3.1}$$

第 2 章のクラスタベースの手法と同様，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．

2.4.1 項における議論と同様に，このアルゴリズムの最悪計算量は，すべてのデー

上位クラスタ	中心データ	メンバ
UC_1	o_1	LC_1
UC_2	o_3	$LC_{2,1}, LC_{2,2}$

クラスタ	代表データ（中心データ）	メンバ
$LC_{1,1}$	o_1	o_1, o_2
$LC_{2,1}$	o_3	o_3, o_4, o_6
$LC_{2,2}$	o_5	o_5, o_7, o_8



(a) 空間位置　　(b) 環境属性値

図 3.1: 階層的クラスタリングの例

タが別々の上位クラスタに分離されてしまう場合であり，$O(N^2)$ となる．

階層的クラスタリング例

図 3.1(a) および (b) に示す具体例を用いて，環境属性の次元数 $d = 2$ の場合の，オフライン事前階層的クラスタリング処理について説明する．まず，データ o_1 を中心として上位クラスタ UC_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 の円内に存在するため，上位クラスタ UC_1 のメンバとする．次に，上位クラスタ UC_1 の配下で下位クラスタの分割を行うため，データ o_1 を中心として下位クラスタ $LC_{1,1}$ を生成する．データ o_2 はデータ o_1 を中心とした環境属性値空間で半径 r_2 の円内に存在するため，下位クラスタ $LC_{1,1}$ のメンバとする．続いて，データ o_3 を中心として上位クラスタ UC_2 を生成する．残りのクラスタに割り当てられていないすべてのデータは，データ o_3 を中心とした空間距離 r_1 の円内に存在するため，これらのデータをすべて上位クラスタ UC_2 のメンバとする．同様に，上位クラスタ UC_2 の配下で下位クラスタの分割を行うと，データ o_3 を中心とした

下位クラスタ $LC_{2,1}$ およびデータ o_5 を中心とした下位クラスタ $LC_{2,2}$ が生成される．このように，すべてのデータは 2 つの上位クラスタに割り当てられ，それぞれの上位クラスタの配下で合計 3 つの下位クラスタに割り当てられる．

3.3.2 階層クラスタを利用したオンラインクエリ処理

第 2 章のクラスタベースの手法では，すべてのクラスタを探索し基準データを選択していたが，本章の提案手法では上位クラスタのみを探索し，上位クラスタの代表データの中から基準データを選択する．上位クラスタの配下の下位クラスタのうち，評価値の大きいデータを含む下位クラスタを記憶しておくことで，評価値の大きい基準データをより短時間で選択できる．また，上位クラスタの評価値の上界を計算することで，まず最初に走査対象のデータの絞り込みを上位クラスタ単位で行う．上位クラスタ内のデータの評価値が取りうる値の上界が基準データの評価値を上回る場合は，下位クラスタ内のデータの評価値が取りうる値の上界を計算し基準データの評価値と比較することで，より細かい粒度で走査対象のデータの絞り込みを行う．本項では，この提案アルゴリズムを実現する上で重要な，上位クラスタの評価値の上界および下位クラスタの上界を計算する方法についても新たに示す．

具体的なアルゴリズムを Algorithm 6 に示し，反復部分（3〜25 行目）について説明する．まず，すべての上位クラスタを走査し，基準データを選択する（5 行目）．基準データの評価値は高いほど望ましいため，上位クラスタ内の下位クラスタの代表データをすべて走査し，スコアが最大のものを選択する．各上位クラスタごとに，スコアが最大の下位クラスタの代表データを記憶しておき，以降の反復ではこれら $|\boldsymbol{UC}|$ 個のデータの中から基準データを選択する．次に，すべての上位クラスタについて，各上位クラスタ内のデータの評価値が取りうる値の上界（以降では，単に**上位クラスタの上界**と略記）$\overline{d_r(UC_i, S^*)}$ を計算する（7 行目）．各上位クラスタの上界と，基準データの評価値 $d_r(o^*_{rep}, S^*)$ を比較し，上界のほうが大きい場合，走査対象の上位クラスタ集合 $\boldsymbol{UC'}$ に追加する．一方，上界のほうが小さい場合，少なくともこの上位クラスタ内のすべてのデータよりも，基準デー

Algorithm 6 Algorithm for Optimization Problem Leveraging Hierarchical Clusters

Input: Set of upper-level clusters $\boldsymbol{UC}$, diversified set size k, importance of spatial diversity λ, weight vector $\boldsymbol{w}$, spatial radius r_1, environmental attribute radius r_2
Output: Set $S^*(|S^*| = k)$ that maximizes $f(S)$

1: Initialize the set $S^* = \emptyset$
2: Find $o^* = \arg\max_{o \in O} p(o)$ and set $S^* = \{o^*\}$
3: **while** $|S^*| < k$ **do**
4:　Initialize the sets $\boldsymbol{UC'} = \boldsymbol{LC'} = \emptyset$
5:　Find o^*_{rep} such that $o^*_{rep} = \arg\max_{o_{i,rep} \in UC_i} d_r(o_{i,rep}, S^*)$
6:　**for all** $i = 1$ to $|\boldsymbol{UC}|$ **do**
7:　　Estimate upper bound of each upper cluster $\overline{d_r(UC_i, S^*)}$
8:　　**if** $d_r(o^*_{rep}, S^*) \leq \overline{d_r(UC_i, S^*)}$ **then**
9:　　$\boldsymbol{UC'} = \boldsymbol{UC'} \cup \{UC_i\}$
10:　　**end if**
11:　**end for**
12:　**for all** $i = 1$ to $|\boldsymbol{UC'}|$ **do**
13:　　**for all** $j = 1$ to $|UC_i|$ **do**
14:　　Estimate upper bound of each lower-level cluster $\overline{d_r(LC_{ij}, S^*)}$
15:　　**if** $d_r(o^*_{rep}, S^*) \leq \overline{d_r(LC_{ij}, S^*)}$ **then**
16:　　$\boldsymbol{LC'} = \boldsymbol{LC'} \cup \{LC_{ij}\}$
17:　　**end if**
18:　　**end for**
19:　**end for**
20:　Find $o^* \in \boldsymbol{LC'} \backslash S^*$ such that $o^* = \arg\max_{o \in \boldsymbol{LC'} \backslash S^*} d_r(o, S^*)$
21:　**if** o^* is representative data of LC_{ij} **then**
22:　　Select new representative data for LC_{ij}
23:　**end if**
24:　Set $S^* = S^* \cup \{o^*\}$
25: **end while**

タ o^*_{rep} のほうが多様集合に追加するデータとして適しているため，以降の走査からは除外する．すべての上位クラスタをチェックした後，走査対象の上位クラスタ集合 $\boldsymbol{UC'}$ に含まれる下位クラスタについて，走査対象をさらに絞り込む（12〜19 行目）．ここでは，上位クラスタの絞り込みと同様，各下位クラスタの上界と基準

データの評価値を比較し，走査対象の下位クラスタ集合 $\boldsymbol{LC'}$ に該当する下位クラスタを追加する．最後に，$\boldsymbol{LC'}$ 内のすべてのデータの評価値を計算し，最大の評価値を取るデータを多様集合 S^* に追加する．

上位クラスタおよび下位クラスタの上界の計算

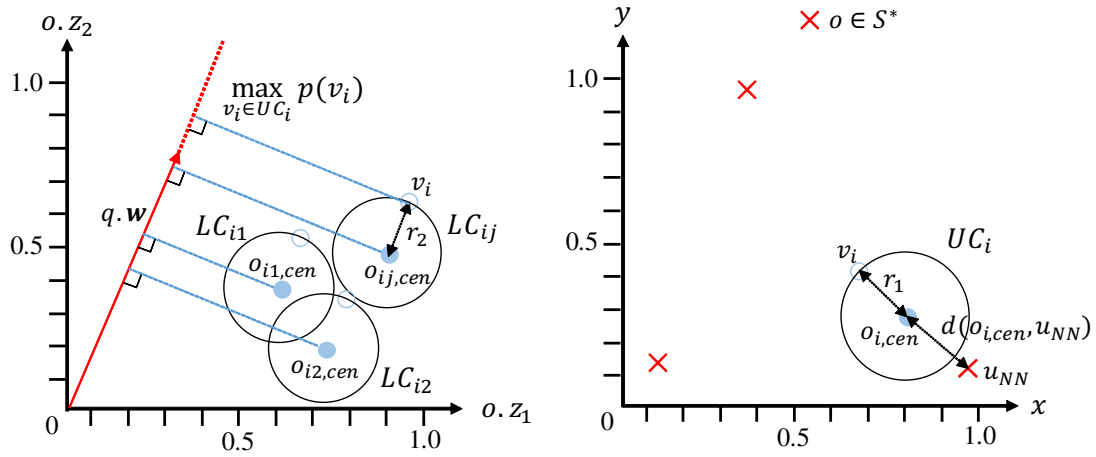
ここで，Algorithm 6 の 7 行目における，上位クラスタ内のデータが取りうる評価値の上界 $\overline{d_r(UC_i, S^*)}$ および 14 行目における下位クラスタ内のデータが取りうる評価値の上界（以降では，単に**下位クラスタの上界**と略記）$\overline{d_r(LC_{ij}, S^*)}$ の計算方法について説明する．まず，上位クラスタの上界について，上位クラスタが含むデータの分布の詳細は不明なため，上位クラスタ内に存在しうる仮想的なデータ v_i を考え，データ v_i が取りうる最大の評価値を計算する．上位クラスタ内のデータは，環境属性値の類似度を考慮してさらに下位クラスタに分割される．そのため，上位クラスタの上界は，配下の下位クラスタの上界のうち最大のものとして，以下の式で与えられる．

$$\overline{d_r(UC_i, S^*)} = \max_{LC_{ij} \in UC_i, 1 \leq j \leq m} \{\overline{d_r(LC_{ij}, S^*)}\} \tag{3.2}$$

次に，下位クラスタ内のデータが取りうる評価値の上界 $\overline{d_r(LC_{ij}, S^*)}$ の計算方法について説明する．下位クラスタ内に存在しうる仮想的なデータ v_{ij} を考え，データ v_{ij} が取りうる最大の評価値を，可能な限り正確に計算する．第 2 章のクラスタベースの手法のクラスタの上界と同様，評価値を多様集合内のデータに非依存の項と依存する項に分解し，別々に上界を計算することで求める．多様集合内のデータに非依存の項（第 1 項）の上界は，下位クラスタの中心データ $o_{ij,cen}$ を用いて，以下の式で計算できる（図 3.2(a)）．

$$\max_{v_{ij} \in LC_{ij}} \{p(v_{ij})\} = p(o_{ij,cen}) + |q.\boldsymbol{w}|r_2 \tag{3.3}$$

次に，多様集合内のデータに依存する項（第 2 項）の上界を計算する．MaxMin 問題および MMR 問題それぞれについて，上位クラスタの中心データと多様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，その時の距離が最



(a) 第1項の上界の計算　　(b) 第2項の上界の計算

図 3.2: 階層クラスタの上界の計算

小となる多様集合内のデータを $u_{NN} \in S^*$ とする．中心データとデータ u_{NN} を直線で結んだ時，2 つの交点が存在する（図 3.2(b)）．ここで，データ u_{NN} から最も離れる位置は，2 つの交点の内，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，多様集合 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 項が取りうる最大値は，以下の式で与えられる．

$$\max_{v_{ij} \in LC_{ij}} \{\min_{u \in S^*}(\frac{1}{2}p(u) + \lambda dist(v_{ij}, u))\} = \frac{1}{2}p(u_{NN}) + \lambda\{dist(o_{i,cen}, u_{NN}) + r_1\} \quad (3.4)$$

$$\max_{v_{ij} \in LC_{ij}} \{\min_{u \in S^*}(\lambda dist(v_{ij}, u))\} = \lambda\{dist(o_{i,cen}, u_{NN}) + r_1\} \quad (3.5)$$

MaxSum 問題において，仮想データ v_{ij} の位置として，多様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．勿論このような，仮想データが複数の位置情報を有する仮定は成り立ち得ないが，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 項が取り

うる最大値は，以下の式で与えられる．

$$\max_{v_{ij}\in LC_{ij}}\{\sum_{u\in S^*}(p(u)+2\lambda dist(v_{ij},u))\}=\sum_{u\in S^*}\{p(u)+2\lambda(dist(o_{i,cen},u)+r_1)\} \quad (3.6)$$

これらの式から，MAXMIN 問題，MAXSUM 問題，MMR 問題における下位クラスタの上界を，それぞれ以下のように計算できる．

$$\begin{aligned}\overline{d_r^{min}(LC_{ij},S^*)}&=\frac{1}{2}\{p(o_{ij,cen})+|q.\boldsymbol{w}|r_2\}+\frac{1}{2}p(u_{NN})+\lambda\{dist(o_{i,cen},u_{NN})+r_1\}\\&=\{\frac{1}{2}(p(o_{ij,cen})+p(u_{NN}))+\lambda dist(o_{i,cen},u_{NN})\}+\frac{1}{2}|q.\boldsymbol{w}|r_2+\lambda r_1\\&=d_r^{min}(o_{ij,axis},S^*)+\frac{1}{2}|q.\boldsymbol{w}|r_2+\lambda r_1 \quad (3.7)\end{aligned}$$

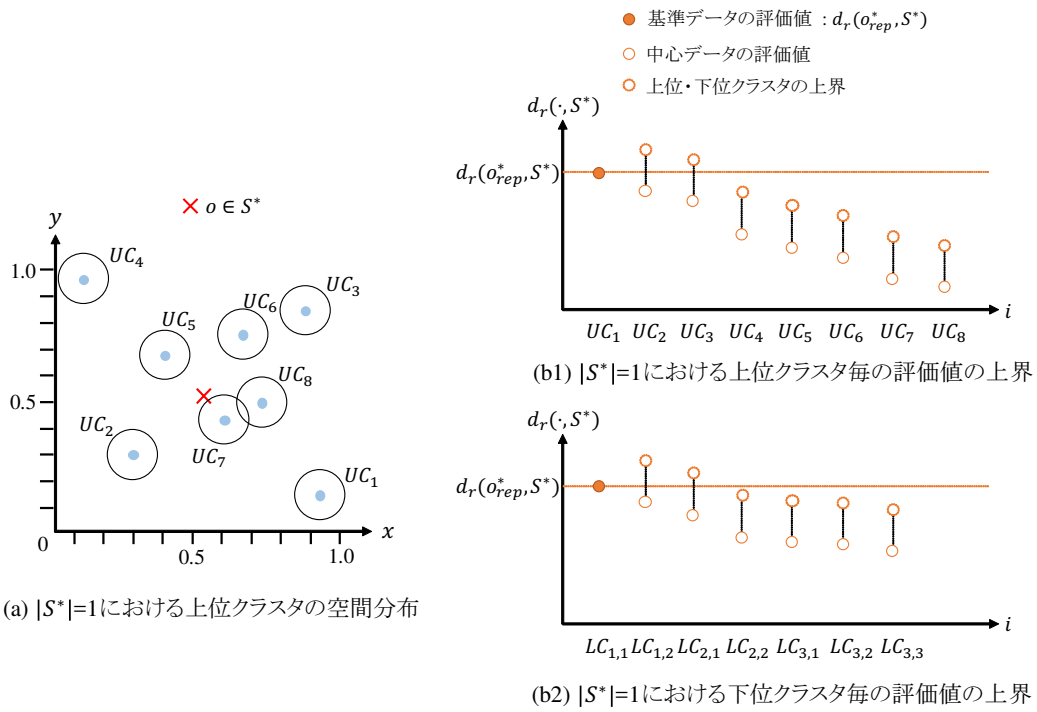
$$\begin{aligned}\overline{d_r^{sum}(LC_{ij},S^*)}&=|S|\{p(o_{ij,cen})+|q.\boldsymbol{w}|r_2\}+\sum_{u\in S^*}\{p(u)+2\lambda(dist(o_{i,cen},u)+r_1)\}\\&=\{|S^*|p(o_{ij,cen})+\sum_{u\in S^*}(p(u)+2\lambda dist(o_{i,cen},u))\}+|S^*||q.\boldsymbol{w}|r_2+2|S^*|\lambda r_1\\&=d_r^{sum}(o_{ij,axis},S^*)+|S^*|(|q.\boldsymbol{w}|r_2+2\lambda r_1) \quad (3.8)\end{aligned}$$

$$\begin{aligned}\overline{d_r^{mmr}(LC_{ij},S^*)}&=(1-\lambda)\{p(o_{ij,cen})+|q.\boldsymbol{w}|r_2\}+\lambda\{dist(o_{i,cen},u_{NN})+r_1\}\\&=\{(1-\lambda)p(o_{ij,cen})+\lambda dist(o_{i,cen},u_{NN})\}+(1-\lambda)|q.\boldsymbol{w}|r_2+\lambda r_1\\&=d_r^{mmr}(o_{ij,axis},S^*)+(1-\lambda)|q.\boldsymbol{w}|r_2+\lambda r_1 \quad (3.9)\end{aligned}$$

ここで $o_{ij,axis}$ は，上位クラスタ UC_i の中心データ $o_{i,cen}$ の空間位置ベクトルと，下位クラスタ LC_{ij} の中心データ $o_{ij,cen}$ の環境属性値ベクトルからなり，以下の式で与えられる．

$$o_{ij,axis}=(o_{i,cen}.x,o_{i,cen}.y,o_{ij,cen}.z_1,...,o_{ij,cen}.z_d) \quad (3.10)$$

このデータの評価値 $d_r(o_{ij,axis},S^*)$ は，Algorithm 6 の 5 行目の基準データを探索する際に計算でき，この値を下位クラスタごとに記憶しておくことで，以降の反復で継続して用いることができる．また，式 (3.7)，(3.8)，(3.9) から下位クラスタの



(a) $|S^*|$=1における上位クラスタの空間分布

(b1) $|S^*|$=1における上位クラスタ毎の評価値の上界

(b2) $|S^*|$=1における下位クラスタ毎の評価値の上界

図 3.3: オンラインクエリ処理の例

上界は，下位クラスタ内のデータが取りうるスコア（第 1 項）の上界にのみ依存する．よって，上位クラスタの上界は，スコアの上界が最大の下位クラスタの上界となる（図 3.2(a)）．スコアの上界は，多様集合内のデータに非依存であるため一度だけ計算されればよく，以降は上位クラスタごとに記憶した値を用いる．

オンラインクエリ処理例

図 3.3 を用いて，走査するデータの数を削減する方法を例示する．図 3.3(a) は，初期化処理が完了し，$|S^*| = 1$ となっている状態である．基準データ 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．次に，各クラスタの中心データの評価値から，クラスタの上界を計算する．

図 3.3(b1) に，計算されたそれぞれの値の分布を示す．このとき，上位クラスタ $UC_4 \sim UC_8$ は，計算された上界が基準データ o^*_{rep} の評価値を下回っているため，走査対象の上位クラスタ集合 $\boldsymbol{UC'}$ に追加されない．よって，上位クラスタ

$UC_4 \sim UC_8$ 内のデータは走査する必要はない．次に，上位クラスタ $UC_1 \sim UC_3$ それぞれの配下の下位クラスタの上界を調べる．図 3.3(b2) に，計算されたそれぞれの値の分布を示す．このとき，下位クラスタ $LC_{2,2}, LC_{3,1}, LC_{3,2}, LC_{3,3}$ は，計算された上界が基準データ o^*_{rep} の評価値を下回っているため，走査対象の下位クラスタ集合 $\boldsymbol{LC'}$ に追加されない．よって，上位クラスタ $LC_{2,2}, LC_{3,1}, LC_{3,2}, LC_{3,3}$ 内のデータは走査する必要はない．

階層クラスタのファイル管理方法

作成された階層クラスタのファイル管理方法について説明する．第 2 章と同様，提案手法では，以下で説明するファイル管理方法を用いることで，大部分のクラスタ内のデータをディスクから読み込むことなく多様集合が得られるため，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，まず最初に上位クラスタ，下位クラスタの中心データおよび代表データのみが走査される．詳細は後述するように，上位クラスタや下位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得ない．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．

構造化された階層クラスタは，図 3.4 に示されるような上位・下位インデックスファイルと，関連する下位クラスタの数分の下位クラスタファイルとして保存される．上位インデックスファイルは，各上位クラスタの中心データの位置情報および下位インデックスファイルへのポインタからなる．また，下位インデックスファイルは，各下位クラスタの中心データ（代表データ）の位置情報，環境属性値，そして当該下位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下位クラスタの下位クラスタファイルを読み込み，走査できる．

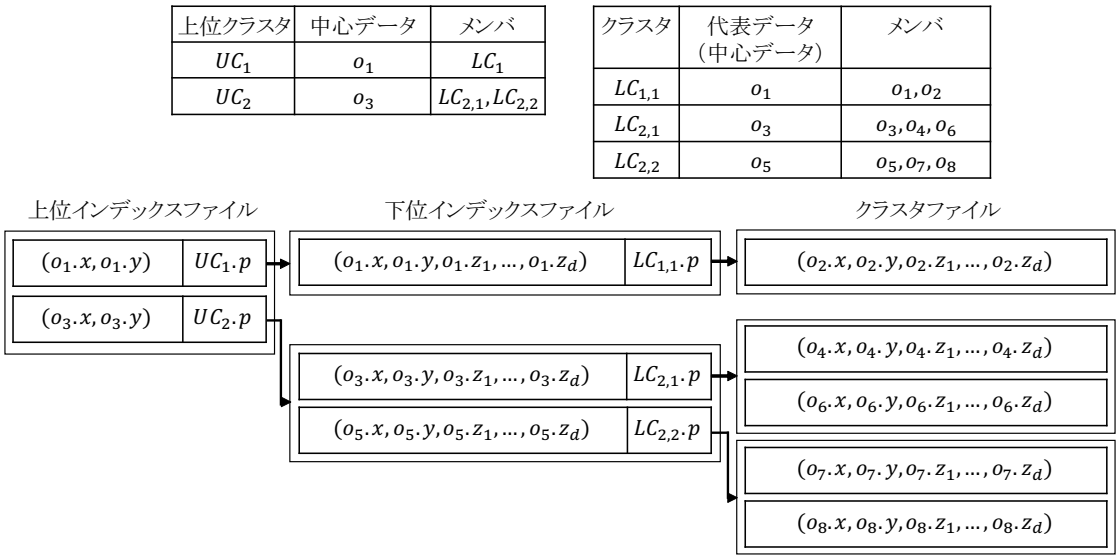


図 3.4: クラスタファイル集合

計算量分析

階層クラスタを利用したオンラインクエリ処理の時間計算量について分析する．多様集合の計算には，はじめは空集合の S^* に最適なデータを追加するために k 回のデータおよびクラスタの走査を繰り返す．この際のループ変数を i $(i = |S^*| + 1)$ とする．また，上位クラスタの数を u，各上位クラスタの配下に存在する下位クラスタの総数を l とする．ここで，1 回あたりの反復における計算量を考える．

まず，1 回目の反復においては，上位クラスタ単位でのデータの絞り込みは出来ない．このために，すべての下位クラスタの代表データのスコアの計算，および下位クラスタの上界の計算，走査対象の下位クラスタ集合の決定のための下位クラスタの走査が必要となる．これらの一連の処理の計算量は，$2l$ である．次に，走査対象の下位クラスタ集合中のすべてのデータのスコアの計算が必要となる．このデータの数を n_i^* とすると，計算量は n_1^* となる．よって，1 回目の反復の計算量は，$2l + n_1^*$ である．

次に，2 回目以降の反復について考える．まず，走査対象の上位クラスタ集合を決定するために，すべての上位クラスタの代表データの評価値の計算，および上位クラスタの上界の計算が必要となる．データの評価値の計算には，1 回のデータの

スコアの計算に加え，$|S^*| = i-1$ 個のデータとの距離の計算が必要となる．よって，u 個の上位クラスタの代表データの評価値の計算の計算量は，$u \cdot \{1+(i-1)\} = ui$ となる．さらに，基準データの評価値を上界が下回る上位クラスタを除外するために，u 個のクラスタの走査が必要となる．よって，走査対象の上位クラスタ集合を決定するための計算量はこれらの合計であり，$ui + u = u(i+1)$ となる．走査対象の上位クラスタ集合を決定した後，次に走査対象の下位クラスタ集合を決定する．このための計算量は，走査対象の上位クラスタの配下に存在する下位クラスタの総数を l_i^* とすると，走査対象の上位クラスタ集合を決定するための計算量と同様に求められ，$l_i^*(i+1)$ となる．最後に，走査対象の下位クラスタ集合中のすべてのデータの評価値計算が必要となり，この計算量は $n_i^* i$ である．ここで，各反復における走査対象の下位クラスタの総数 l_i^* の最大値を $\max_{2\leq i\leq k}\{l_i^*\} = \overline{l^*}$，走査対象の下位クラスタ集合内のデータの総数 n_i^* の最大値を $\max_{1\leq i\leq k}\{n_i^*\} = \overline{n^*}$ とする．

オンラインクエリ処理の全体の計算量は，k 回の反復の合計であるため，次式で表される．

$$
\begin{aligned}
&(2l + n_1^*) + \sum_{i=2}^{k}\{u(i+1) + l_i^*(i+1) + n_i^* i\} \\
&\leq (2l + \overline{n^*}) + (u + \overline{l^*} + \overline{n^*}) \sum_{i=2}^{k} i + \sum_{i=2}^{k}(u + \overline{l^*}) \\
&= 2l + \overline{n^*} + (u + \overline{l^*} + \overline{n^*})\{\frac{1}{2}k(k+1) - 1\} + (u + \overline{l^*})(k-1) \\
&\leq \frac{1}{2}k(k+1)(u + \overline{l^*} + \overline{n^*}) + 2l + \overline{n^*} + (u + \overline{l^*})k
\end{aligned}
\tag{3.11}
$$

よって，オンラインクエリ処理の計算量は，$O(k^2(u + \overline{l^*} + \overline{n^*}) + l)$ である．

2.4.2 項の末尾で述べたとおり，第 2 章のクラスタベースの手法におけるクラスタの数を c としたとき，クラスタベースの手法の計算量は $O(k^2(c + \overline{n^*}))$ である．ここで，提案手法とクラスタベースの手法で空間半径 r_1 が同じ値の場合，上位クラスタの数 u は，下位クラスタの数 l や c に比べ小さい値となる．また，提案手法とクラスタベースの手法で環境属性値半径 r_2 が同じ値の場合，l および c は近い値

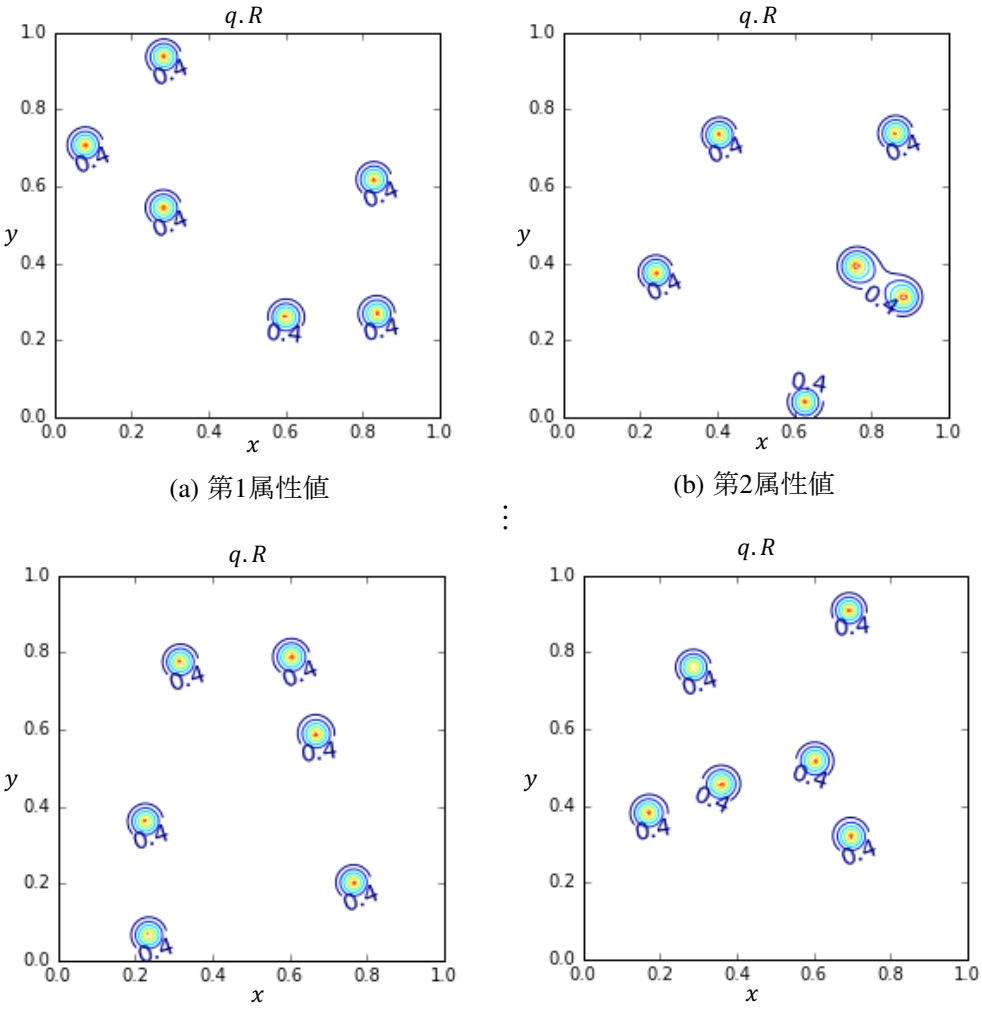
表 3.1: パラメータの値

パラメータ	値
データセットサイズ N	**1M**, 5M, 10M
要求するデータの数 k	5∼50 (**25**)
多様性の重要度 λ	0.0∼1.0 (**0.5**)
$\boldsymbol{w}$ の各要素	0.0∼1.0
環境属性の次元数 d	4, 10, **20**

となる．さらに，最終的な走査対象のデータの総数 $\overline{n^*}$ もまた，それぞれの手法において近い値となる．ここで，環境属性の次元数 d が大きくなると，データ同士が類似しにくいため，l および c はともに大きな値となる．一方で，提案手法では多くの場合，大部分の上位クラスタが走査対象から除かれるため，走査対象の下位クラスタの総数 $\overline{l^*}$ は，c に比べ小さい値となる．以上のことから，空間半径 r_1 および環境属性値半径 r_2 が同じ値の場合，提案手法の計算量はクラスタベースの手法の計算量に比べて小さくなる．また，クラスタベースの手法において，環境属性値半径 r_2 を大きくするとクラスタの数 c は小さくなるが，クラスタの上界が大きくなることから，走査対象から除外されるクラスタの数は小さくなる．このため，クラスタベースの手法で環境属性値半径 r_2 を大きくすると，最終的な走査対象のデータの数 $\overline{n^*}$ は大きくなってしまう．よって，それぞれの手法において適切な環境属性値半径 r_2 を用いた場合においても，提案手法の計算量はクラスタベースの手法の計算量に比べて小さくなる．

3.4　性能評価

本節では，本章における提案手法の性能を評価する．表 3.1 は各パラメータの値を示し，太字はデフォルト値とする．



(a) 第1属性値　(b) 第2属性値

(c) 第19属性値　(d) 第20属性値

図 3.5: 環境属性値の分布

3.4.1 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 3.5 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 20 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，$N(0, 0.05)$ の正規分布に従う正規乱数を加算した．

3.4.2　設定

提案手法（以降のグラフ中では ‘Proposed’ と表記）を，第 2 章で説明したクラスタベースの手法（‘Compared’）と比較した．すべてのアルゴリズムを Java8 で実装し，Intel Xeon E5-2643 v2 @ 3.50GHz with 48.0GB RAM を搭載する Ubuntu 14.04.5 LTS 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の数としてディスク IO コストを示し，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の数，下位クラスタの数，および上位クラスタの数を示す．さらに，各手法の走査対象のデータの絞り込みの効果を調べるために，多様集合を計算するまでに走査したデータの数を示す．

実験で用いたクエリは，$q.\boldsymbol{w}$ が表 3.1 に示す一定の範囲内でランダムに設定されたものである．作成されたランダムな 100 個のクエリを処理した際の，各評価指標の平均値を調べた．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，クラスタの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の数をそのまま示している．

3.4.3　空間半径 r_1 および環境属性値半径 r_2 の影響

環境属性の次元数 d が 20 の場合における，クラスタの空間半径 r_1 を変化させたときの，計算時間，ディスク IO コスト，走査したデータの数，クラスタの数を図 3.6 に示す．ここでは，MMR 問題のみの評価指標を計測している．図 3.6(a) より，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ることが分かる．これは，2.5.4 項の議論と同様，クラスタ内のデータの評価値の上界を過大に計算

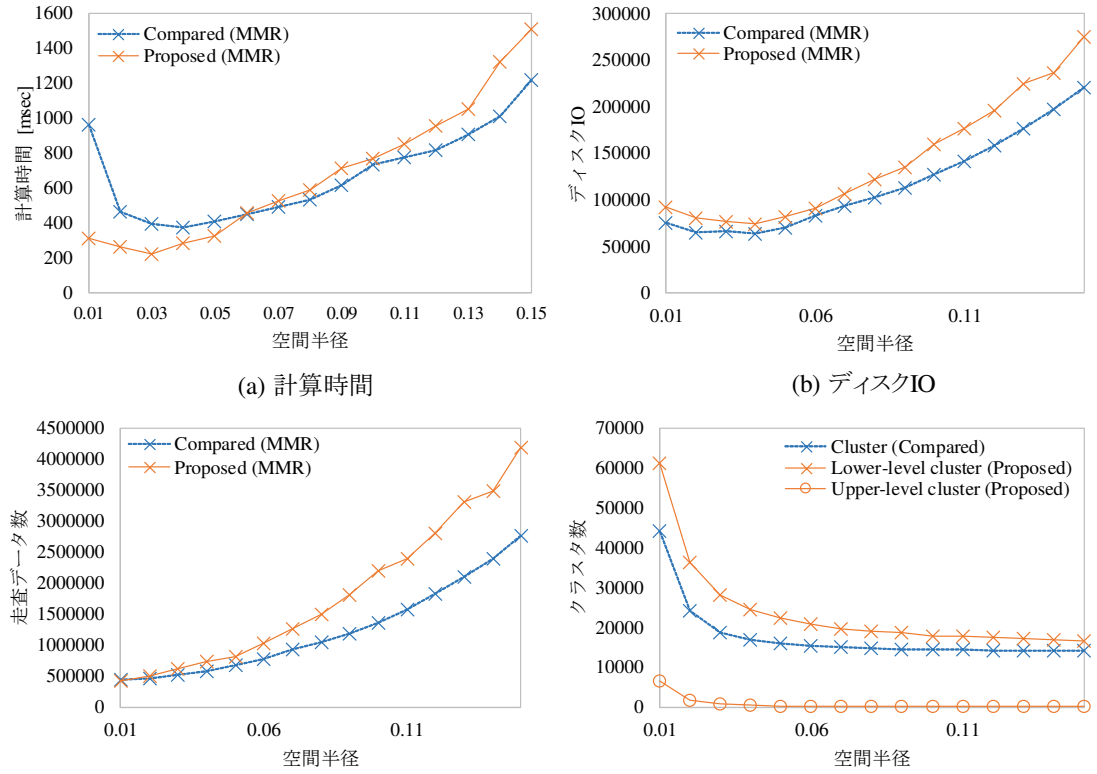


図 3.6: 空間半径 r_1 の影響

することで，走査対象から除外できたクラスタの数が小さいためである．この事実は，図 3.6(c) より，空間半径が大きいほど，走査したデータの数が大きいことからも確認できる．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ることが分かる．これは，2.5.4 項の議論と同様，図 3.6(d) より空間半径が小さくなると生成されるクラスタの数が増加することによる．クラスタ半径が小さい場合，図 3.6(c) より，走査対象のクラスタの数が削減されることで，走査対象のデータの数は削減される．一方で，走査対象のクラスタ集合を絞り込むための計算コストが大きくなってしまうため，計算時間は増加してしまう．また，図 3.6(b) より，提案手法のディスク IO コストは比較手法のディスク IO コストに比べてわずかに大きいことが分かる．これは，図 3.6(d) が示すように，下位クラスタの数がクラスタの数に比べてやや大きいため，下位インデックスファイルの

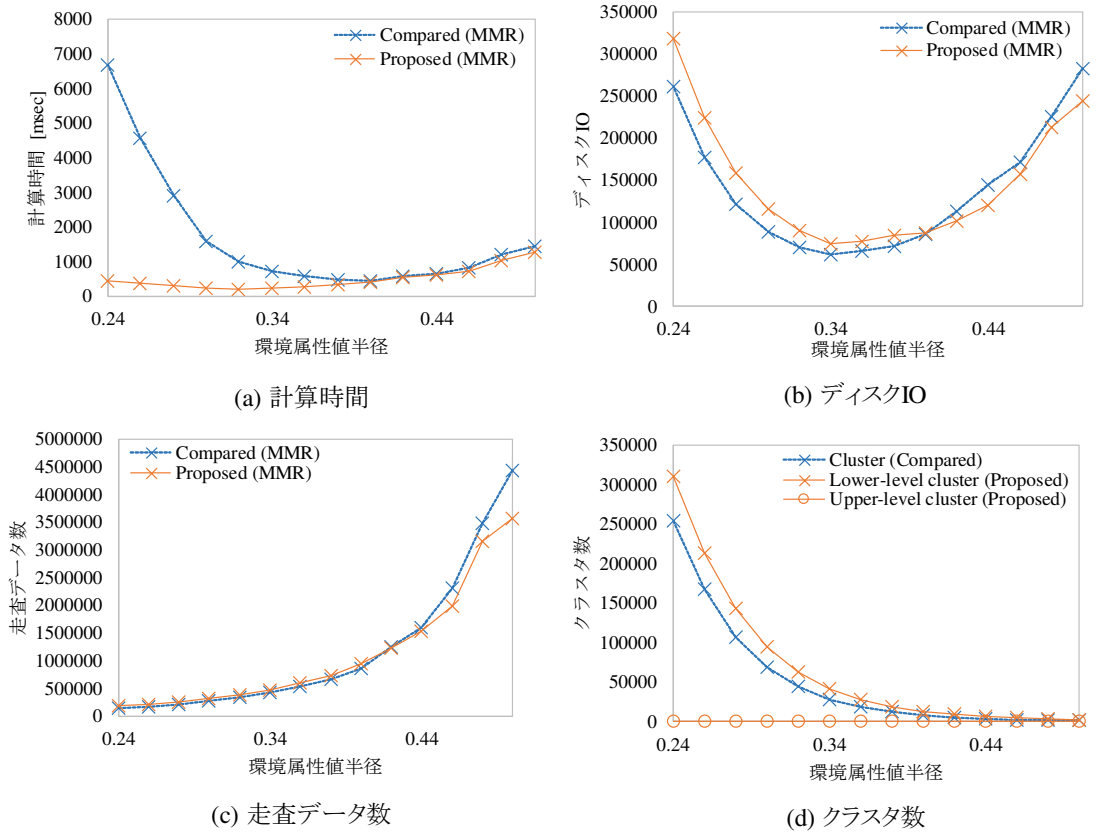


図 3.7: 環境属性値半径 r_2 の影響

合計サイズも大きくなり，初期状態で読み込むデータ量が大きいためである．

次に，環境属性の次元数 d が 20 の場合における，クラスタの環境属性値半径 r_2 を変化させたときの，計算時間，ディスク IO コスト，走査したデータの数，クラスタの数を図 3.7 に示す．図 3.7(a) より，環境属性値半径が大きい場合，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計算してしまい，走査対象から除外できたクラスタの数が少ないためである．一方，環境属性値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，環境属性値半径が小さくなると生成されるクラスタの数が増加し，走査対象のクラスタ集合の絞り込みのための追加の処理の計算コストが大きくなるためである．

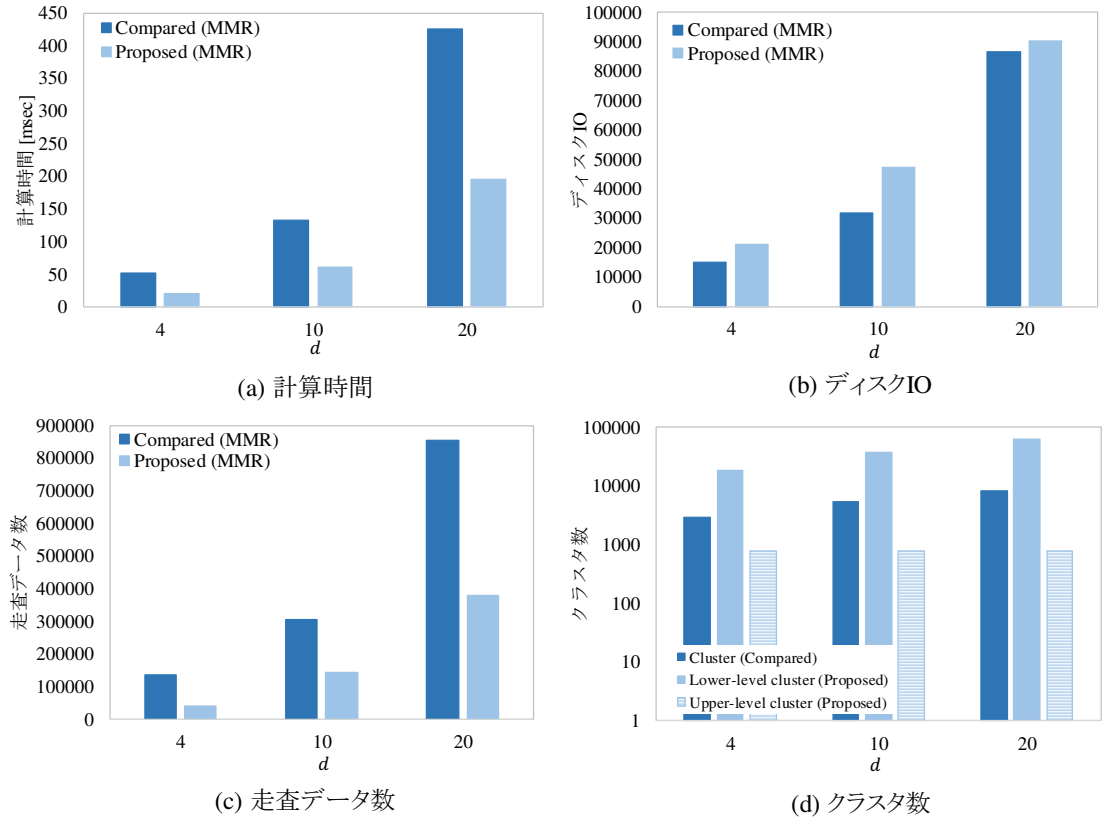


図 3.8: $d = 4, 10, 20$ における結果

しかし，提案手法は比較手法に比べて，空間半径が小さい場合の計算時間の増加の度合いが小さいことが分かる．このことから，提案手法は比較手法よりもクラスタの数が増加した場合に，走査対象のデータを絞り込むための計算コストを削減でき，クラスタの増加の影響を緩和できていることが確認できる．

ここで，環境属性の次元数が $d = 4, 10, 20$ のそれぞれの場合について，比較・提案手法の計算時間を最短にした r_1，r_2 を用いたときの結果を図 3.8 に示す．図 3.8(a) より，それぞれの次元数において，提案手法の計算時間は比較手法の計算時間の半分程度まで削減されていることが分かる．これは，比較手法よりも提案手法のほうが，走査対象のデータを絞り込むための計算コストが小さく，かつ走査対象のデータの数も小さいためである．図 3.8(d) より，提案手法の下位クラスタの数は比較手法のクラスタの数よりもやや大きいが，上位クラスタの数は環境属

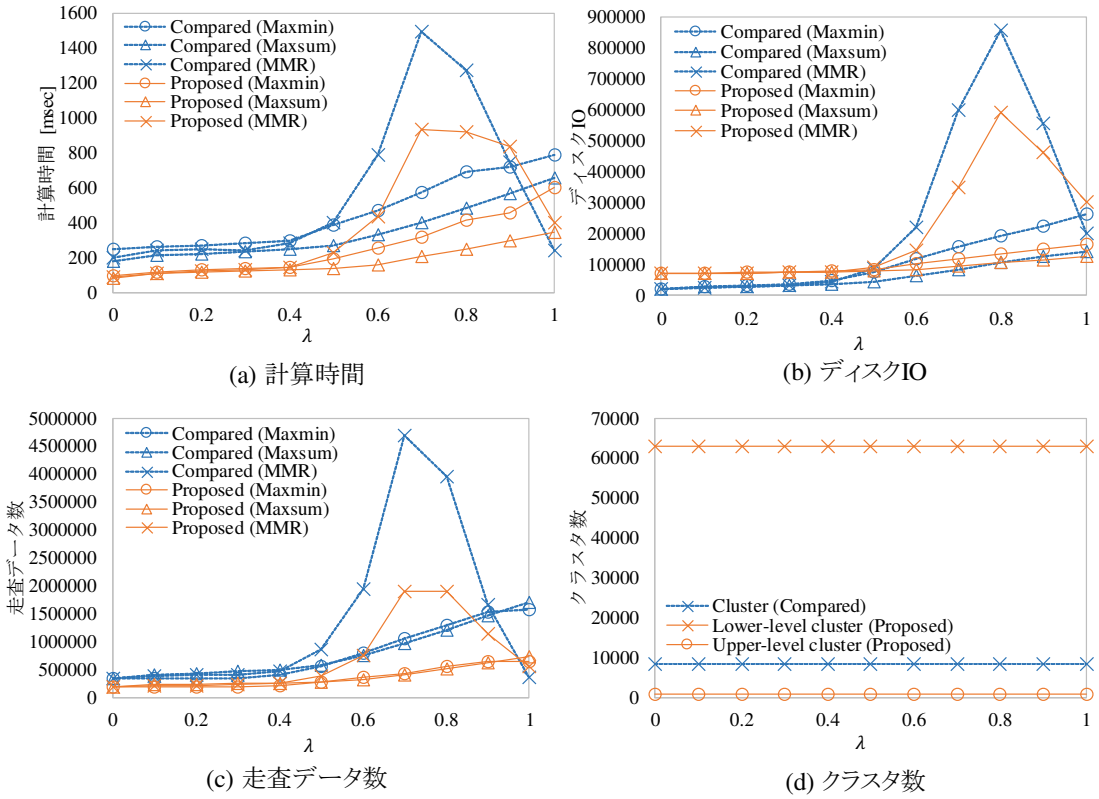
性の次元数に依存せず一定の小さい値に保たれている．このため，走査対象のクラスタ集合を絞り込む際，上位クラスタごとに絞り込むことで計算コストを小さくできている．また，図 3.7(a) より，提案手法では比較手法よりも環境属性値半径を小さく設定できることが分かる．上述したように，環境属性値半径を小さくするとクラスタの数が増加するが，提案手法では上位クラスタごとのデータの絞り込みにより，クラスタの増加の影響を緩和できる．一方で，比較手法ではクラスタの数を小さくするために環境属性値半径を大きく設定する必要がある．このため，図 3.8(c) に示されるように，比較手法は提案手法に比べて走査対象のデータの数が大きくなってしまい，計算時間が増加してしまう．一方で，図 3.8(b) に示されるように，比較手法のクラスタの数に比べて提案手法の下位クラスタの数はやや大きくなるため，最初に読み込むインデックスファイルのサイズが大きくなることから，ディスク IO コストは比較手法に比べてわずかに大きくなる．

以降の実験では，比較・提案手法それぞれにおいて計算時間を最短にした r_1 および r_2 を用いている．

3.4.4 その他のパラメータの影響

多様性の重要度 λ の影響

MaxMin，MaxSum，MMR 問題において多様性の重要度 λ を変化させた場合の計算時間，ディスク IO コスト，走査したデータの数，クラスタの数を図 3.9 に示す．λ はクエリパラメータでありクラスタ構造に影響を与えないため，図 3.9(d) が示す通り，比較・提案手法におけるそれぞれのクラスタの数は一定である．図 3.9(a)，(b)，(c) より，MaxMin，MaxSum 問題では，λ が大きくなるほど，各手法の計算時間，ディスク IO コスト，走査対象のデータの数が大きくなることが分かる．これは，λ が大きくなるほど目的関数におけるデータ間の距離の重要度が大きくなり，走査対象となるデータの分布する領域が大きくなるためである．走査対象のデータの数が大きくなるため，これにともないディスク IO コストも大きくなる．特に，MMR 問題では λ の影響が他の問題よりも大きく，$\lambda = 0.6$ より大きい値で各手法において計算時間が大幅に増加し，$\lambda = 0.9, 1.0$ の場合は，提案手



(a) 計算時間 (b) ディスクIO

(c) 走査データ数 (d) クラスタ数

図 3.9: 多様性の重要度 λ の影響

法の計算時間が比較手法の計算時間をやや上回っている．MMR 問題において λ を大きく設定すると，データのスコアに対してデータ間の空間距離が支配的になり，データのスコアがほとんど考慮されずに多様集合が計算される．このような多様集合は，ユーザの興味に適合するとは限らないデータから構成されるため，モバイルセンサデータベースにおける Top-k 検索結果の多様化で想定されるアプリケーションの 1 つである，ユーザが注目すべき領域であるホットスポットの検出には適さない．一般的に，ユーザの興味に適合した有用な結果を得るためには，λ をある程度小さく設定し，データ間の距離だけでなくデータのスコアに対しても重み付けする必要があり，この場合は提案手法の計算時間は各問題について，比較手法の計算時間よりも短くなっている．

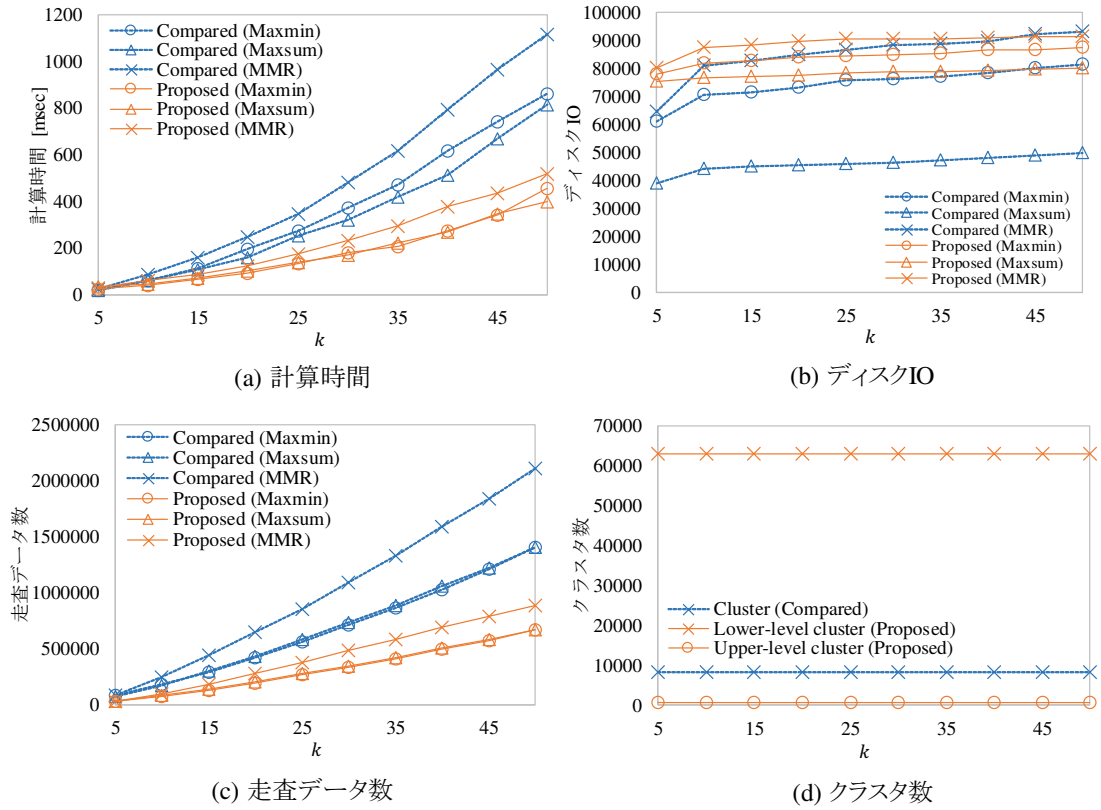


図 3.10: 要求するデータの数 k の影響

要求するデータの数 k の影響

要求するデータの数 k を変化させた場合の計算時間，ディスク IO コスト，走査したデータの数，クラスタの数を図 3.10 に示す．λ と同様に k もクエリパラメータであるため，図 3.10(d) が示す通り，比較・提案手法におけるそれぞれのクラスタの数は一定である．図 3.10(a) より，k が小さい場合は，比較・提案手法における計算時間にほとんど差がないことが分かる．これは，提案手法では最初に多様集合に追加するスコアが最大のデータを探索する際に，すべての下位クラスタの代表データを走査するためである．k が小さい場合はこのような初期化処理の計算コストが大きいため，各手法の計算時間は各問題についてほぼおなじ値となる．一方，k が大きい場合は，提案手法は比較手法よりも走査対象のデータを絞り込むための計算コストが小さく，かつ走査対象のデータの数を削減できるため，各問

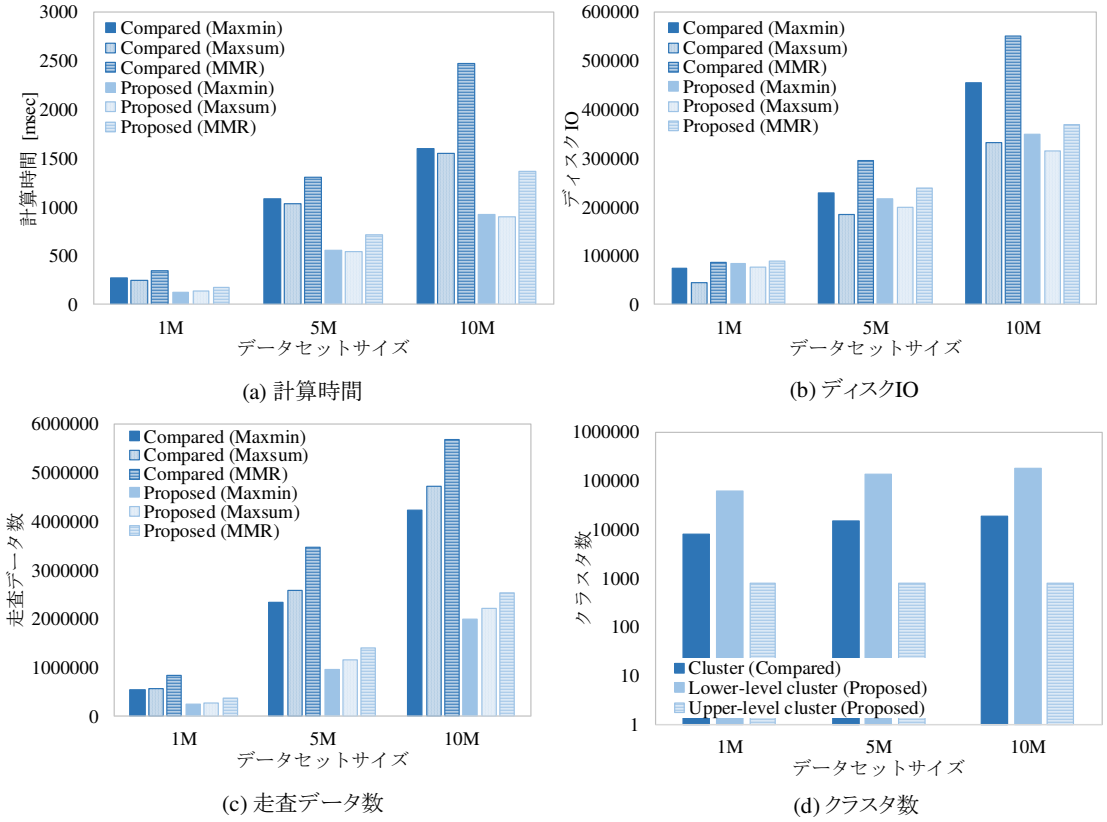


図 3.11: データセットサイズ N の影響

題について計算時間を比較手法の半分程度まで削減できている．

データセットサイズ N の影響

データセットサイズ N を変化させた場合の計算時間，ディスク IO コスト，走査したデータの数，クラスタの数を図 3.11 に示す．図 3.11(a) より，いずれのデータセットサイズの場合においても，各問題について提案手法の計算時間は，比較手法の計算時間よりも短くなることが分かる．また，図 3.11(d) より，データセットサイズが大きくなるほど，比較・提案手法におけるクラスタの数および下位クラスタの数は大きくなることが分かる．これにより，各手法の走査対象のデータの絞り込みのための計算コストも大きくなる．しかし，図 3.11(b)，(c) が示す通り，提案手法は比較手法よりもディスク IO コストおよび走査対象のデータの数を削減出

来ており，多様集合の計算時間は比較手法よりも 40 ～ 50%程度小さくなっている．

3.5　むすび

3.5.1　発展：階層クラスタのメンテナンス方法

オフライン事前階層的クラスタリング処理の後にも新たにデータはアップロードされるが，これらのデータを既存の階層クラスタに組み込むことで，階層クラスタのメンテナンスが可能である．ここで，新たにアップロードされたデータを o_{new} とする．階層クラスタのメンテナンスアルゴリズムを，Algorithm 7 に示す．3.3.1 項の冒頭で述べた通り，上位クラスタは上位クラスタの中心データの空間位置ベクトルを中心とした半径 r_1 の円内に存在するデータから，下位クラスタは下位クラスタの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から構成されている．よって，既存の上位・下位クラスタの中から，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良い．

まず，o_{new} が適合する上位クラスタの集合は，既存の上位クラスタの中心データの集合を対象として，Algorithm 5 で用いた retrieveNeighbors($o_{new}, r_1, \infty, \boldsymbol{UC}$) を実行することで取得できる（1 行目）．取得した上位クラスタの集合 $\boldsymbol{UC}^*$ の上位クラスタは，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上位クラスタ UC_i^* のクラスタメンバとする（3 行目）．次に，上位クラスタ UC_i^* の配下に存在する下位クラスタ $\boldsymbol{LC}_i$ の中から，o_{new} が適合する下位クラスタの集合を，下位クラスタの中心データの集合を対象として，retrieveNeighbors($o_{new}, \infty, r_2, \boldsymbol{LC}_i$) を実行することで取得する（4 行目）．取得した下位クラスタの集合 $\boldsymbol{LC}_i^*$ の下位クラスタは，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下位クラスタ LC_{ij}^* のクラスタメンバとし，LC_{ij}^* のクラスタファイルに o_{new} を挿入する（6，7 行目）．o_{new} が適合する上位クラスタおよび下位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（9，12 行目）

Algorithm 7 Algorithm for Maintenance of Hierarchical Clusters

Input: Set of upper-level clusters $\boldsymbol{UC}$, spatial radius r_1, environmental attribute radius r_2, new data item o_{new}

1: $\boldsymbol{UC}^* =$ retrieveNeighbors($o_{new}, r_1, \infty, \boldsymbol{UC}$)
2: **if** $\boldsymbol{UC}^*$ is not NULL **then**
3: $UC_i^* =$ random($\boldsymbol{UC}^*$)
4: $\boldsymbol{LC}_i^* =$ retrieveNeighbors($o_{new}, \infty, r_2, \boldsymbol{LC}_i$)
5: **if** $\boldsymbol{LC}_i^*$ is not NULL **then**
6: $LC_{ij}^* =$ random($\boldsymbol{LC}_i^*$)
7: Mark o_{new} with LC_{ij}^*.lowerClusterLabel and LC_{ij}^*.upperClusterLabel
8: **else**
9: Mark o_{new} as the center and initial representative of the new lower-level cluster
10: **end if**
11: **else**
12: Mark o_{new} as the center of the new upper-level and lower-level cluster
13: **end if**

3.5.2 まとめ

本章では，高次元な環境情報を扱う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した．提案手法では，事前にモバイルセンサデータに対し階層的クラスタリングを行う．オンラインクエリ処理では，上位クラスタおよび下位クラスタの上界を計算し，それぞれのクラスタ単位で走査対象のデータの絞り込みを行う．また，基準データの候補を上位クラスタごとに記憶することで，短時間で評価値の高い基準データを取得できる．

シミュレーション実験により，提案手法は第 2 章で提案した手法と比較して，モバイルセンサデータが高次元の場合でも多様集合をより短時間で取得できることを確認した．さらに，提案手法は比較手法よりも，環境属性値半径を小さくした場合のクラスタの増加に伴う計算コストの増加を緩和できることを確認した．

参加型センシングに参加する各モバイルセンサは非同期でセンシングを行うことから，検索範囲を指定する際は一定の時間幅を指定した時空間範囲を指定する必要がある．しかし，第 2 章および本章では，多様集合を計算する際に空間的多様性のみを考慮して計算を行っており，時間属性を考慮していなかった．このた

め，特定の時刻付近に生成されたデータが多様集合に含まれる可能性があり，時間的に冗長な結果が得られる場合がある．空間的および時間的に分散した多様集合を計算することで，検索範囲内のホットスポットをより効果的に検出できると考えられる．このような要求に対応するために，多様集合を計算するための最適化問題の再定義，およびその際の効率的な Top-k 検索結果の多様化手法の設計について検討する予定である．

第4章　格子グリッドベースのデータ構造を用いた継続的なTop-k検索結果の多様化手法

4.1　まえがき

1.3.3 項で述べた通り，モバイルセンサデータが時々刻々と到着するモバイルセンサストリーム環境において，地理的多様性を考慮した多様集合をモニタリングすることで，ホットスポットの地理的分布の変化をリアルタイムに追跡できる．モバイルセンサストリーム環境において多様集合をモニタリングする場合，多様集合内のデータが一定時間経過し削除される場合や，よりスコアが大きい，もしくはより地理的に分散するデータが生成された場合に，定期的に多様集合を更新する必要がある．しかし，前述の過去に収集され蓄積されたモバイルセンサデータベースにおける多様集合の計算と同様に，単純な手法では多様集合の更新に要する計算コストは大きい．

一方で，モバイルセンサストリーム環境において，第 2 章や第 3 章で説明した事前クラスタリング処理を行う手法を用いる場合，到着するデータや削除されるデータに対し，多様集合の更新の前に，クラスタ構造の更新を行う必要がある．クラスタ構造を利用することで多様集合の更新は短時間で可能であるものの，クラスタ構造の更新にかかる時間が非常に大きくなるため，結果的に多様集合の計算時間は短縮できない．

そこで本章では，データの更新コストが小さい格子グリッドベースのデータ構造を用いた，モバイルセンサストリーム環境における効率的な多様集合のモニタリング手法を提案する．格子グリッドの空間セルは，環境属性ごとにデータのリ

ストを保持しており，それぞれのリストは環境属性値が一定の範囲内であるデータを管理する．それぞれのデータを被覆する空間セルおよび管理するリストは定数時間で決定できるため，提案データ構造はストリーム環境に適している．提案アルゴリズムは，データを走査する途中で，データの走査をする必要が無いと判断した空間セルを枝刈りし，走査するデータの数を削減する．具体的には，最適なデータを走査する際，空間セル内のデータがとりうるスコアの上界を計算し，これを用いて空間セル内のデータの評価値が取りうる値の上界を計算する．その際に保持している評価値が最大のデータと評価値の上界を比較し，上界の方が小さい場合は，その空間セル内のほかのデータは走査する必要はない．また，空間セル内のデータのスコアについて，取りうる値が単調に減少するような順番で空間セル内のデータを走査することで，評価値の上界は減少する．これにより，空間セル内のデータの走査を途中で打ち切り，次の空間セルの走査に進める．結果として，走査するデータの数を大幅に削減しつつ，単純にモニタリングの対象のすべてのデータを走査することで得られる多様集合と，同一の検索結果を取得できる．

以下では，まず 4.2 節で関連研究について述べる．4.3 節で想定環境を紹介し，本章の問題を定義する．4.4 節でベースライン手法を紹介し，4.5 節で提案手法について説明する．その後，4.6 節でシミュレーション実験の結果を示す．最後に，4.7 節で本章のまとめと今後の課題について述べる．

4.2 関連研究

多次元データストリーム環境において，Top-k 検索や k 最近傍検索の結果を効率的に計算する手法に関する研究は，これまでに多数行われている [46, 47, 62, 63, 83, 93]．文献 [62] ではスコアが上位 k 個のデータを効率的にモニタリングするために，属性値空間を多次元グリッドに分割する．そして，各セル内のデータのスコアが取りうる上界を計算し，上界の大きさによってセルを走査する順序，データの走査を打ち切るタイミングを決定する．また，文献 [63] では指定された地点に対する k 個の最近傍データを効率的にモニタリングするために，属性値空間を多次元グリッドに分割する．Top-k データのモニタリングと同様に，各セル内のデータと

指定された地点との間の距離の上界を計算し，上界の大きさに基づいてデータを走査する．これらの研究で提案されているデータ構造は，それぞれデータのスコアかデータ間の距離のいずれか一方のみの上界しか計算できない．しかし，多様化問題ではデータのスコアおよびデータ間の距離の両方からデータの評価値が決定されるため，これらの既存手法はそのままでは多様化問題に適用できない．ここで，文献 [62] における多次元グリッドインデックスを環境属性値の多次元インデックスとして，文献 [63] における多次元グリッドインデックスを位置情報の 2 次元グリッドインデックスとして多様化問題に適用する場合を考える．この場合，各多次元グリッドセル内のデータが取りうる評価値の上界を計算できるため，走査対象となるデータの絞り込みが可能となり，短時間で多様集合が計算できる可能性がある．しかし，文献 [62] における多次元グリッドインデックスは，多次元グリッドセルの数が環境属性値の次元数に対し指数的に増加するため，環境属性値ベクトルの次元数が大きい場合メモリサイズが非常に大きくなる．またこのとき，データの追加は短時間で可能である一方で，データの削除については多数の多次元グリッドセルを巡回する必要があり，データ構造の更新コストが大きくなる．文献 [62] では最大で 6 次元程度を想定しているが，本研究では，第 3 章で想定したように様々な環境情報を観測する，より高次元なモバイルセンサデータを対象としている．そこで，データのスコアとデータ間距離の両方の上界を計算でき，かつ高次元のモバイルセンサデータについても適用可能なデータ構造と，データ構造を利用した検索結果の更新アルゴリズムが必要となる．

これまでに，継続的な Top-k 検索結果の多様化に関する研究がいくつか行われている [4, 20, 28, 29, 58]．文献 [28] では，MaxMin 問題における多様集合を継続的にモニタリングするための，木構造インデックスが提案されている．また，文献 [4] では，分散ストリーム環境において，サーバ間の通信コストおよび計算コストの削減を目的とした，多様集合のモニタリング手法が提案されている．これらの研究における多様集合モニタリング問題は，本章における問題定義と類似している．しかし，文献 [28] における木構造インデックスはデータの更新コストが大きく，データ更新が高頻度である場合に計算時間が爆発的に増加してしまうことが文献 [4] にて指摘されている．文献 [4] では，各サーバは自身が管理する検索範囲

内のデータセットから評価値が最大のデータを取得し，まとめ役となるマスターサーバに送信する．マスターサーバは，各サーバから受信したデータの中で評価値が最大のデータを選択し，これを多様集合に追加することで更新を行う．一方，本章では集中管理のシステムを想定している．文献 [4] において各サーバが管理するデータセットをすべて単一のサーバで管理し，管理するデータセットから評価値が最大のデータを取得することで，文献 [4] における多様集合と同一の結果が得られる．そのため本章では，文献 [4] における手法を集中管理のシステムに適用し，ベースライン手法として用いている．このベースライン手法では，多様集合に新たにデータを追加する場合や多様集合内の既存のデータと別のデータを交換する場合に，モニタリングの対象のデータセット全体を走査しなければならない．その結果，管理するデータ量が大きい場合，集中管理のシステムでは計算時間が大きくなってしまう．

4.3 想定環境と問題定義

本節では，本章の想定環境と問題定義について説明する．

4.3.1 システムおよびデータモデル

本章におけるストリーム環境では，単一の集中管理のサーバを想定する．サーバはデータストリームを受信し，現在時刻 t_c から W 以内に発生したデータを保持する．ここで，W はウィンドウサイズである．つまり，サーバは $t_c - W$ よりも以前に発生したデータをすべて削除する．

モバイルセンサ端末は第 2，3 章と同様に，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，検索範囲内に分布し，かつサーバが保持するデータ集合を O で表す．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 $o.\boldsymbol{z}$ を保持している．位置情報 $o.\boldsymbol{loc}$ は，経度 $o.x$ と緯度 $o.y$ によって表

される 2 次元平面上の点とし，環境属性値 $o.\boldsymbol{z}$ は d 次元のベクトル $o.z_i(i = 1, ..., d)$ で表される．

また，データ o のスコア $p(o)$，および 2 つのデータ u, v 間の多様性 $dist(u, v)$ は第 2，3 章と同様に，それぞれ式 (2.1)，(2.2) に基づいて計算される．

4.3.2 問題定義

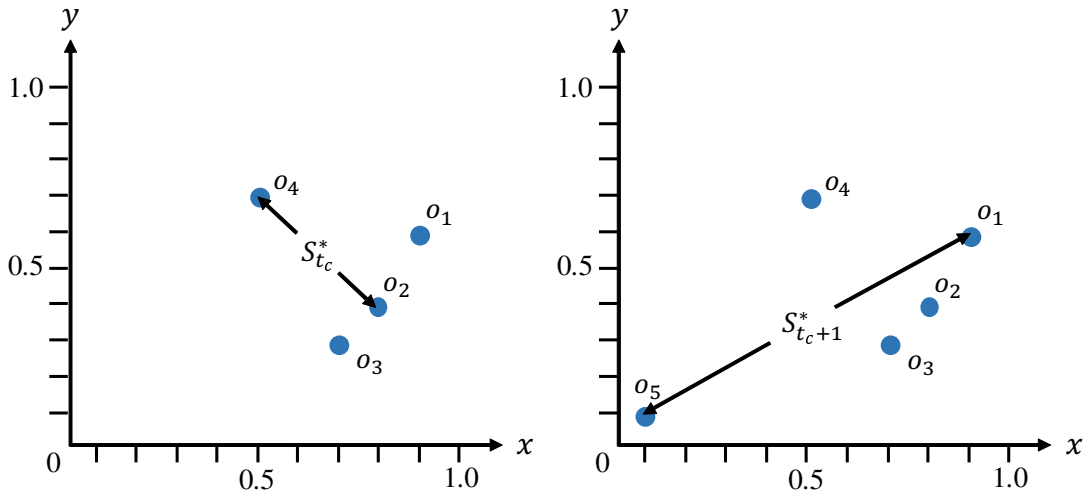
上述した環境属性値から算出されるデータのスコア，およびデータ間の位置情報から算出される空間距離に基づいて，モバイルセンサストリーム環境における多様集合モニタリング問題を定義する．

定義 (多様集合モニタリング問題)**.** 多様集合の大きさ k，検索範囲 $q.R$，およびサーバが保持している $q.R$ 上のデータ集合 O が与えられた時，この問題は，O の部分集合 $S^*_{t_c}$ を継続的に計算する．また，O の部分集合 $S^*_{t_c}$ は式 (4.1) を満たす．

$$S^*_{t_c} = \underset{S_{t_c} \subseteq O, |S_{t_c}|=k}{\arg\max} f(S_{t_c}) \tag{4.1}$$

ここで，$f(S_{t_c})$ は目的関数である．MaxMin，MaxSum，Maximal Marginal Relevance（MMR）における目的関数は，第 2，3 章と同様に，それぞれ式 (2.4)，(2.5)，(2.6) で表される．

サーバは，保持するデータ集合 O から多様集合 $S^*_{t_c}$ を計算し，クエリを発行したユーザに提供するために最新の多様集合を保持する．また，最新の多様集合を取得するために，更新の具体的な処理として，一定時間が経過したデータの削除，削除されたデータの分を補填するためのデータの追加，より評価値の高いデータと多様集合内のデータの交換などを行う．サーバが保持する最新の多様集合は，ユーザからの要求を受け取ったときや，多様集合の構成が変化したときに，ユーザに送信される．これにより，ユーザは多様集合をモニタリングできる．



(a) t_c における多様集合　　(b) $t_c + 1$ における多様集合

図 4.1: $k = 2$ において，$S^*_{t_c+1} \cap S^*_{t_c} = \emptyset$ となる例

要件の補足

以下の例 1 に示すように，多様集合モニタリング問題では，モニタリングの対象のデータ集合 O が変わると（ウィンドウがスライドすると），多様集合が完全に異なるデータ集合となる（$S^*_{t_c} \cap S^*_{t_c-1} = \emptyset$）可能性がある．

例 1. 図 4.1 を用いて，多様集合の変化の例を示す．2 次元のユークリッド空間におけるデータ集合 $O = \{(0.9, 0.6), (0.8, 0.4), (0.7, 0.3), (0.5, 0.7)\}$ を考え，MMR 問題において $k = 2, \lambda = 1.0$，すなわちデータのスコアを無視し，空間的多様性のみを考慮した多様集合を計算する．このとき，$S^*_{t_c} = \{(0.8, 0.4), (0.5, 0.7)\}$ である．ここで，データ $(0.1, 0.1)$ が発生し O に追加された場合を考える．このとき，$S^*_{t_c+1} = \{(0.9, 0.6), (0.1, 0.1)\}$ となり，$S^*_{t_c+1} \cap S^*_{t_c} = \emptyset$ となる．

例 1 から，O の変化が小さい場合でも，$S^*_{t_c}$ の構成が大きく変化する場合があることがわかる．この現象が頻繁に起こる場合，多様集合の変化を追跡することはユーザにとって困難である．そのため，$S^*_{t_c}$ に含まれるデータのうちいくつかは，スライディングウィンドウ上に存在する限り多様集合に残り続けることが望まし

い．このような要求は，文献 [4, 27, 28, 80] においても支持されている．そこで，多様集合 $S^*_{t_c-1}$ および $S^*_{t_c}$ 間の**差分の制限** $\delta(0 \leq \delta \leq k)$ を導入する．ユーザがパラメータ δ を指定することにより，差分の制限を実現できる．

制限 1 (差分の制限)**.** ユーザが指定するパラメータ δ が与えられた時，多様集合 $S^*_{t_c}$ は式 (4.2) を満たす．

$$|S^*_{t_c} \setminus S^*_{t_c-1}| \leq \delta \tag{4.2}$$

ここで，$S^*_{t_c-1}$ 内でウィンドウから削除されるデータの数が δ を超える場合，制限 1 が満たされない．これに対応するため，以下の制限を追加する．

制限 2 (制限 1 の例外処理)**.** $S^*_{t_c-1}$ 内でウィンドウから除かれるデータの数を e とする．$\delta < e$ である場合，多様集合 $S^*_{t_c}$ は式 (4.3) を満たす．

$$|S^*_{t_c} \setminus S^*_{t_c-1}| = e \tag{4.3}$$

以上の制限を考慮し，本章で扱う δ-多様集合モニタリング問題を定義する．

定義 (δ-多様集合モニタリング問題)**.** 多様集合の大きさ k，差分の制限 δ，サーバが保持するデータ集合 O が与えられた時，この問題は，O の部分集合 $S^*_{t_c}$ を継続的に計算する．また，O の部分集合 $S^*_{t_c}$ は式 (4.1)，制限 1 および 2 を満たす．

以降では文脈上明らかな場合は，δ-多様集合を多様集合のように略記する．

差分の制限を設けた δ-多様集合モニタリング問題も NP 困難であるため，近似解を計算するヒューリスティックアルゴリズムが必要である．そこで，本章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．

4.4 ベースライン手法

本節では，多様集合の初期化および更新のベースラインアルゴリズムについて説明する．これらのアルゴリズムは，文献 [4] の提案アルゴリズムを参考にしている．ただし，文献 [4] の提案アルゴリズムは分散ストリーム環境を想定して設計されているため，本節では集中管理システムを想定した設計に修正したアルゴリズムを用いる．

Algorithm 8 Initialization

1: $S^*_{t_c} \leftarrow \emptyset$
2: $o^* \leftarrow \arg\max_{o \in O} p(o)$
3: $S^*_{t_c} \leftarrow S^*_{t_c} \cup \{o^*\}$
4: **while** $|S^*_{t_c}| < k$ **do**
5: $\quad o^* \leftarrow \arg\max_{o \in O \setminus S^*_{t_c}} d_r(o, S^*_{t_c})$ (FindNextObject)
6: $\quad S^*_{t_c} \leftarrow S^*_{t_c} \cup \{o^*\}$
7: **end while**

4.4.1 初期化アルゴリズム

多様集合の初期化アルゴリズムを Algorithm 8 に示す．初期化アルゴリズムは，第 2，3 章の多様集合の計算と同様な手順で，サーバが保持する多様集合の初期化を行う．具体的には，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が最大のデータを，サイズが k となるまで多様集合に繰り返し追加する．よって，MaxMin 問題，MaxSum 問題および MMR 問題における評価値は，それぞれ式 (2.7)，(2.8)，(2.9) により定義される．以降は，このようなグリーディアルゴリズムにおいて追加するデータとして最適なデータ o^* を取得する操作を，FindNextObject と呼ぶ．

4.4.2 更新アルゴリズム

Algorithm 8 を実行し多様集合を初期化した後，各時刻で $S^*_{t_c}$ を更新する必要がある．ここで，多様集合を更新するベースラインアルゴリズムを Algorithm 9 に示す．

現在時刻 t_c において，$S^*_{t_c-1}$ に含まれており，かつスライディングウィンドウから除かれるデータの数を e とする．まず最初に，4 行目から 7 行目の反復により，FindNextObject を繰り返し実行し e 個のデータを追加する．これにより，$e = k$ の場合は $S^*_{t_c}$ を構築し直すこととなるため，Algorithm 9 は Algorithm 8 を実行する（19 行目）．以下では，$e < k$ の場合における更新の手順を説明する．

1，2 行目より，まずはじめに $S^*_{t_c}$ は，$S^*_{t_c-1}$ に含まれており，かつ現在時刻のスライディングウィンドウ上に存在するデータから構成される．e 個のデータを追加

Algorithm 9 Update

1: Delete expired data items from $S^*_{t_c-1}$
2: $S^*_{t_c} \leftarrow S^*_{t_c-1}, e \leftarrow k - |S^*_{t_c-1}|, \delta' \leftarrow \delta - e$
3: **if** $e < k$ **then**
4: **while** $|S^*_{t_c}| < k$ **do**
5: $o^* \leftarrow$ FindNextObject
6: $S^*_{t_c} \leftarrow S^*_{t_c} \cup \{o^*\}$
7: **end while**
8: **while** $\delta' > 0$ **do**
9: $o^{**} \leftarrow \arg\min_{o \in S^*_{t_c}} d_r(o, S^*_{t_c} \setminus \{o\})$
10: $S^*_{t_c} \leftarrow S^*_{t_c} \setminus \{o^{**}\}$
11: $o^* \leftarrow$ FindNextObject
12: $S^*_{t_c} \leftarrow S^*_{t_c} \cup \{o^*\}$
13: **if** o^* is equal to o^{**} **then**
14: **break**
15: **end if**
16: $\delta' \leftarrow \delta' - 1$
17: **end while**
18: **else**
19: Initialization
20: **end if**

した後は，目的関数の値 $f(S^*_{t_c})$ を向上させるために，最大 $\delta' = \delta - e$ 個のデータを交換できる．ここで，$\delta' \leq 0$ の場合は，すでに差分の制限以上の数のデータが変化しているため，データの交換は行われない．8 行目から 17 行目の交換の手続きでは，多様集合への貢献の度合いで交換の対象となるデータを決定する．具体的には，$S^*_{t_c}$ 中のデータで最小の評価値をとるデータ o^{**} であり，以下の式を満たす．

$$o^{**} = \arg\min_{o \in S^*_{t_c}} d_r(o, S^*_{t_c} \setminus \{o\}) \tag{4.4}$$

データ o^{**} を $S^*_{t_c}$ から削除し，この状態で FindNextObject を再び実行し，新しいデータ o^* を追加する．ここで，o^* と o^{**} が一致する場合，交換の手続きによって目的関数の値はこれ以上向上しないため，更新アルゴリズムは終了する．これら

の交換の手続きは，最大で δ' 個のデータが交換されるまで繰り返される．

4.4.3 計算コスト

このアルゴリズムの計算量は，スライディングウィンドウ上のデータの数 $|O|$ 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，単純にスライディングウィンドウ上のデータセット全体の走査が必要となり，計算量は $O(|O|)$ である．FindNextObject の操作に関しては，アルゴリズム実行時の多様集合のサイズを $|S^*_{t_c}| = k'$ とすると，$k'(|O| - k')$ 回のデータ間の空間距離の計算が必要となるため，計算量は $O(k'|O|)$ となる．このため，スライディングウィンドウ上のデータの数が大きくなると計算時間が長くなってしまう．

4.5 提案手法

本節では，本章の提案手法について説明する．ベースライン手法では，各反復において最適なデータを取得するために，$O \setminus S^*_{t_c}$ 内のすべてのデータを走査する必要があり，計算コストが大きくなってしまう．この計算コストを削減するため，第 2 章ではクラスタベースのデータ構造を利用する手法を提案した．この手法では，オフライン事前処理によって，空間的に近接し，かつ環境属性値ベクトルが類似するデータをクラスタ化する．オンラインクエリ処理では，クラスタ内の他のデータの評価値について，その上界はクラスタ半径から計算できる．これにより，最適なデータを含んでいる可能性のあるクラスタのみ走査すればよく，走査するデータの数を削減できる．

しかし，本章のようなストリーム環境にこの手法を適用する場合，各時刻でスライディングウィンドウに対し追加，削除されるデータについて，クラスタベースのデータ構造の更新が必要となる．クラスタの数が大きいほどクラスタの更新コストは大きくなるため，更新が頻繁に発生するストリーム環境ではクラスタベースのデータ構造は適さないと考えられる．そこで，多次元データストリームを扱う既存手法 [47, 62, 63, 83, 93] を踏襲し，提案手法ではスライディングウィンドウ

上のデータを格子グリッドベースのデータ構造で管理する．格子グリッドベースのデータ構造では，複雑な木構造インデックスやクラスタ構造に比べて，短時間でデータの追加，削除が可能である．また，提案データ構造を利用した効率的なFindNextObjectのアルゴリズムを提案する．

4.5.1項では格子グリッドベースのデータ構造について，4.5.2項では提案データ構造を利用した効率的なデータ探索方法について説明する．

4.5.1 格子グリッドベースのデータ構造

提案する格子グリッドベースのデータ構造を図4.2に示す．この例では簡単のために，環境属性の次元数を2（$o.\boldsymbol{z} = (o.z_1, o.z_2)$）とし，それぞれの値は$[0.0, 1.0]$の範囲で正規化されているものとする．空間セルの$x$次元および$y$次元の範囲は$\Delta_{xy}$である．このため，列$c_x$，行$c_y$の空間セル$G(c_x, c_y)$は，$o.x$が範囲$[c_x \cdot \Delta_{xy}, (c_x+1) \cdot \Delta_{xy})$，$o.y$が範囲$[c_y \cdot \Delta_{xy}, (c_y+1) \cdot \Delta_{xy})$内の値をとるすべてのデータを包含する．逆に，位置情報$(o.x, o.y)$を有するデータoが与えられた時，このデータを包含する空間セル$G(o.c_x, o.c_y)$は，$o.c_x = \lfloor \frac{o.x}{\Delta_{xy}} \rfloor$および$o.c_y = \lfloor \frac{o.y}{\Delta_{xy}} \rfloor$として定数時間で決定できる．各空間セルは，環境属性ごとのデータリストを保持する．各データリストは，環境属性値が一定の範囲内の値をとるデータを管理する．z_iにおけるデータリストの範囲はΔ_zである．このため，c_{zi}番目のデータリスト$G(c_x, c_y).L(i, c_{zi})$は，$o.z_i$が範囲$[c_{z_i} \cdot \Delta_z, (c_{z_i}+1) \cdot \Delta_z)$内の値をとるすべてのデータを包含する．逆に，$o.z_i$を有するデータ$o$が与えられた時，このデータを管理するデータリスト$G(o.c_x, o.c_y).L(i, o.c_{z_i})$は，$o.c_{z_i} = \lfloor \frac{o.z_i}{\Delta_z} \rfloor$として定数時間で決定できる．加えて，各環境属性についてデータを1つ以上含むデータリストの番号c_{z_i}の最大値を以下のように計算し各空間セルで記憶する．

$$G(c_x, c_y).\overline{c_{z_i}} = \max\{c_{z_i} \mid |L(i, c_{z_i})| > 0\} \tag{4.5}$$

これらの値は，後述するように空間セル内のデータが取りうる評価値の上界を計算するために用いられる．また，スライディングウィンドウ上で追加，削除されるデータについて，データリストにおいて効率よく追加，削除するために，データ

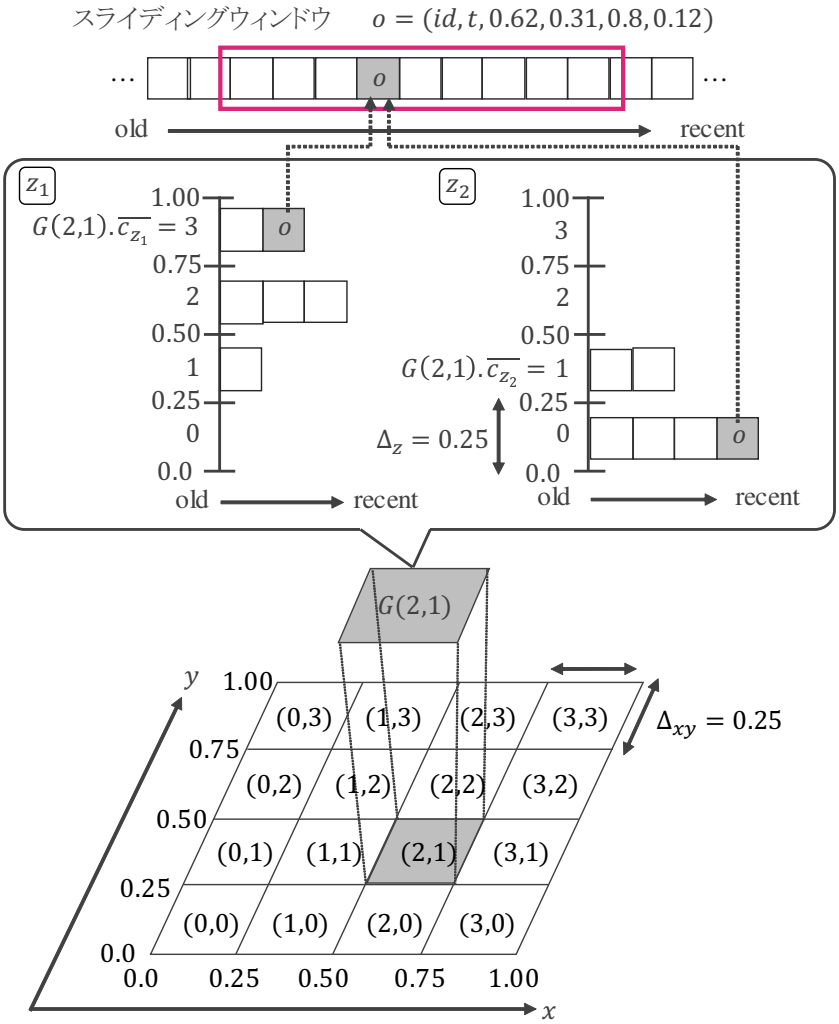


図 4.2: 格子グリッドベースのデータ構造

リストは先入れ先出し（First In First Out）方式で管理する．新しく到着したデータはリストの末尾に配置され，ウィンドウから削除されるデータはリストの先頭から除外されるため，データリストにおける追加，削除の操作はデータ 1 つにつき $O(1)$ で可能である．

データ追加例

提案データ構造にデータが追加される様子を図 4.2 に例示する．例では，範囲 Δ_{xy} および Δ_z はともに 0.25 である．$(o.x = 0.62, o.y = 0.31)$ に位置し，環境属性値 $(o.z_1 = 0.8, o.z_2 = 0.12)$ を有するデータ o は，$G(2,1).L(1,3)$ および $G(2,1).L(2,0)$ （$o.c_x = \lfloor \frac{0.62}{0.25} \rfloor = 2, o.c_y = \lfloor \frac{0.31}{0.25} \rfloor = 1, o.c_{z_1} = \lfloor \frac{0.8}{0.25} \rfloor = 3, o.c_{z_2} = \lfloor \frac{0.12}{0.25} \rfloor = 0$）に挿入される．

更新アルゴリズム

提案データ構造の更新アルゴリズムを Algorithm 10 に示す．各時刻にシステムに到着するデータおよびスライディングウィンドウから削除されるデータが存在する．データ集合 O_{ins} をシステムに到着するデータの集合とし，2 行目から 8 行目において各データが割り当てられる空間セルおよびデータリストを決定する．次に，11 行目から 13 行目において，各空間セルのすべてのデータリストを先頭から走査し，スライディングウィンドウから当該データを削除する．データリスト内で，データは到着した時刻順に配置されているため，スライディングウィンドウ上のデータを見つけた時点で次のデータリストの走査に進む．最後に，14 行目において，各環境属性について $G(c_x, c_y).\overline{c_{z_i}}$ を更新する．

計算量分析

格子グリッドベースのデータ構造の空間計算量について分析する．空間セルの数を c_{xy}，各属性のデータリストの数を c_z とすると，各空間セルは合計 dc_z 個のデータリストを持つ．よって，データを格納する土台となるデータ構造の空間計算量は，$dc_{xy}c_z$ となる．また，各データは環境属性値に応じてそれぞれの環境属性のデータリストに配置されるため，データ 1 つにつき d 個のポインタが張られる．よって，スライディングウィンドウ上のデータの数を n とすると，合計 nd 個のポインタが必要となる．以上より，格子グリッドベースのデータ構造の空間計算量は，$O(d(c_{xy}c_z + n))$ となる．

Algorithm 10 Update Grid Structure

1: $O_{ins} \leftarrow$ set of arriving data items
2: **for** data item $o \in O_{ins}$ **do**
3: $o.c_x \leftarrow \lfloor \frac{o.x}{\Delta_{xy}} \rfloor, o.c_y \leftarrow \lfloor \frac{o.y}{\Delta_{xy}} \rfloor$
4: **for** $i = 1$ to d **do**
5: $o.c_{z_i} \leftarrow \lfloor \frac{o.z_i}{\Delta_z} \rfloor$
6: Insert o into $G(o.c_x, o.c_y).L(i, o.c_{z_i})$
7: **end for**
8: **end for**
9: **for** every spatial cell $G(c_x, c_y)$ **do**
10: **for** $i = 1$ to d **do**
11: **for** every data list $G(c_x, c_y).L(i, c_{z_i})$ **do**
12: Delete expired data o from $G(c_x, c_y).L(i, c_{z_i})$
13: **end for**
14: $G(c_x, c_y).\overline{c_{z_i}} \leftarrow \max\{c_{z_i} \mid |L(i, c_{z_i})| > 0\}$
15: **end for**
16: **end for**

4.5.2 グリッド構造を利用した効率的な計算手法

本項では，提案データ構造を利用した効率的な FindNextObject のアルゴリズムについて説明する．提案アルゴリズムを Algorithm 11 に示す．提案アルゴリズムでは，空間セルおよびヒープ H が主要なデータ構造であり，ヒープ H は空間セルの走査する順序を決定する．まず，ヒープ H，暫定の最適なデータ o^*，暫定の最適なデータの評価値 $d_r(o^*, S^*_{t_c})$ を初期化する（1 行目）．次に，各空間セルについて，空間セル内のデータが取りうる評価値の上界（以降では，単に**空間セルの上界**と略記）を計算する（3 行目）．各空間セル $G(c_x, c_y)$ は，その評価値の上界 $\overline{d_r(G(c_x, c_y), S^*_{t_c})}$ をソートキーとし，ヒープに挿入される（4 行目）．

6 行目から 19 行目の反復において，ヒープ H の各エントリを走査する．空間セルの上界が，暫定の最大の評価値 $d_r(o^*, S^*_{t_c})$ を上回っている場合，空間セルは最適なデータを含んでいる可能性があるため，空間セル内のデータを走査する（8 行目）．データを走査する際にはまず，走査する環境属性 i を NextAttribute を実行することで決定する（9 行目）．NextAttribute は 1 から d の値をラウンドロビン方式

Algorithm 11 FindNextObject Leveraging Grid Structure

1: $H \leftarrow \emptyset, o^* \leftarrow NULL, d_r(o^*, S^*_{t_c}) \leftarrow -\infty$
2: **for** every spatial cell $G(c_x, c_y)$ **do**
3: Compute $\overline{d_r(G(c_x, c_y), S^*_{t_c})}$
4: Enheap in H entry $\langle \overline{d_r(G(c_x, c_y), S^*_{t_c})}, G(c_x, c_y) \rangle$
5: **end for**
6: **while** $H.top$ is not $NULL$ **do**
7: Deheap from H entry $\langle \overline{d_r(G(c_x, c_y), S^*_{t_c})}, G(c_x, c_y) \rangle$
8: **while** $d_r(o^*, S^*_{t_c}) < \overline{d_r(G(c_x, c_y), S^*_{t_c})}$ **do**
9: $i \leftarrow$ NextAttribute
10: $o \leftarrow G(c_x, c_y).L(i, \overline{c_{z_i}})$.next()
11: **if** o is not $NULL$ **then**
12: Compute $d_r(o, S^*_{t_c})$
13: Update o^* and $d_r(o^*, S^*_{t_c})$
14: **else**
15: $G(c_x, c_y).\overline{c_{z_i}} \leftarrow G(c_x, c_y).\overline{c_{z_i}} - 1$
16: $\overline{d_r(G(c_x, c_y), S^*_{t_c})} \leftarrow \overline{d_r(G(c_x, c_y), S^*_{t_c})} - \Delta_r$
17: **end if**
18: **end while**
19: **end while**
20: return o^*

で返す．環境属性値が大きく，評価値が大きい可能性の高いデータを優先的に走査するために，データリスト $G(c_x, c_y).L(i, \overline{c_{z_i}})$ から走査する．データリストからデータ o が得られた場合は o の評価値を計算し（12 行目），この評価値が暫定の最大値であった場合は，o^* および $d_r(o^*, S^*_{t_c})$ を更新する（13 行目）．一方，データリストからデータが得られなかった場合は，$G(c_x, c_y).L(i, \overline{c_{z_i}})$ 内のデータはすべて走査済みであるため，$G(c_x, c_y).\overline{c_{z_i}}$ をデクリメントし次のデータリストの走査へ進む（15 行目）．この際，環境属性値の上界が減少するため，これにともない評価値の上界も各問題に応じた値だけ減少する．（16 行目）．

次の反復において，$\overline{d_r(G(c_x, c_y), S^*_{t_c})} < d_r(o^*, S^*_{t_c})$ が満たされる場合，空間セル $G(c_x, c_y)$ 内の他のデータの評価値はデータ o^* の評価値より大きくなりえない．このため，空間セルの走査を打ち切ることができ，結果として走査対象のデータの

数を削減できる．また，重みベクトルの各重みが負の場合においても，データリストを逆順に走査することで同様の手続きが可能となる．

空間セルの上界の計算

ここで，Algorithm 11 の 3 行目における，空間セルの評価値の上界 $\overline{d_r(G(c_x,c_y),S^*_{t_c})}$ の計算方法について説明する．空間セルが含むデータの分布の詳細は不明なため，空間セル内に存在しうる仮想的なデータ v_i を考え，データ v_i が取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，多様集合内のデータに非依存の項と依存する項に分解する．

$$d_r^{min}(v_i,S^*_{t_c}) = \frac{1}{2}p(v_i) + \min_{u\in S^*_{t_c}}\{\frac{1}{2}p(u) + \lambda dist(v_i,u)\} \tag{4.6}$$

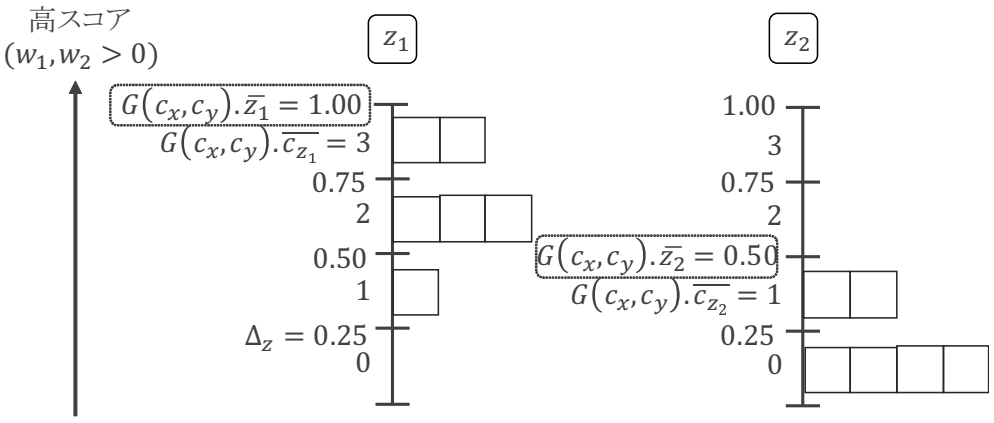
$$d_r^{sum}(v_i,S^*_{t_c}) = |S^*_{t_c}|p(v_i) + \sum_{u\in S^*_{t_c}}\{p(u) + 2\lambda dist(v_i,u)\} \tag{4.7}$$

$$d_r^{mmr}(v_i,S^*_{t_c}) = (1-\lambda)p(v_i) + \min_{u\in S^*_{t_c}}\{\lambda dist(v_i,u)\} \tag{4.8}$$

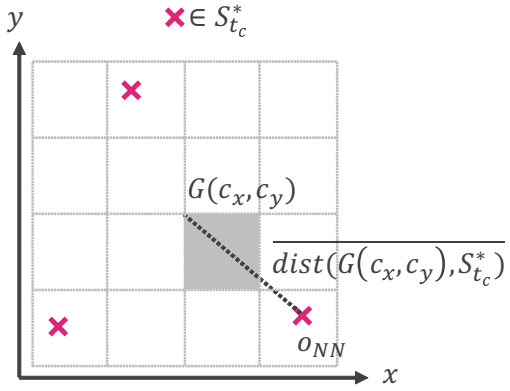
まず，多様集合内のデータに非依存の項 (第 1 項) が取りうる最大値 $\max_{v_i\in C_i}\{p(v_i)\} = \overline{p(G(c_x,c_y))}$ を計算する．ここでは，重みベクトルの各重みは正の値であると想定する．図 4.3 (a) に示すように，各環境属性値の上界 $G(c_x,c_y).\overline{z_i}$ は，4.5.1 項で説明した $G(c_x,c_y).\overline{c_{z_i}}$ を用いることで $G(c_x,c_y).\overline{z_i} = \Delta_z\cdot(G(c_x,c_y).\overline{c_{z_i}}+1)$ のように計算できる．式 (2.1) から，データのスコアは環境属性値の重み付き和によって計算される．このため，$\overline{p(G(c_x,c_y))}$ は以下のように $G(c_x,c_y).\overline{z_i}$ の重み付き和によって計算される．

$$\begin{aligned}\max_{v_i\in G(c_x,c_y)}\{p(v_i)\} &= \overline{p(G(c_x,c_y))}\\ &= \sum_{i=1}^{d} q.w_i\cdot G(c_x,c_y).\overline{z_i}\end{aligned} \tag{4.9}$$

重みベクトルの各重みが負の値である場合も，$G(c_x,c_y).\overline{c_{z_i}}$ の代わりに，データを 1 つ以上含むリストの番号 c_{z_i} の最小値を用いることで計算できる．



(a) 第1項の上界の計算



(b) 第2項の上界の計算

図 4.3: 空間セルの上界の計算

次に，多様集合内のデータに依存する項（第 2 項）が取りうる最大値を計算する．MaxMin 問題および MMR 問題のそれぞれについて，空間セルの中心座標と多様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，値が最小となる多様集合内のデータを $o_{NN} \in S^*_{t_c}$ とする．図 4.3 (b) に示すように，第 2 項の上界は空間セル $G(c_x, c_y)$ の各頂点と o_{NN} との間の空間距離のうち最大のもので計算できる．このときの仮想データ v_i，データ o_{NN} 間の取りうる最大の空間距離を $\overline{dist(G(c_x, c_y), o_{NN})}$ とすると，MaxMin 問題および MMR 問題における第 2 項

が取りうる最大値は，以下の式で表される．

$$\max_{v_i \in G(c_x,c_y)} \{\min_{o \in S^*_{t_c}} \{\frac{1}{2}p(o) + \lambda dist(v_i, o)\}\} = \frac{1}{2}p(o_{NN}) + \lambda\overline{dist(G(c_x, c_y), o_{NN})} \quad (4.10)$$

$$\max_{v_i \in G(c_x,c_y)} \{\min_{o \in S^*_{t_c}} (\lambda dist(v_i, o))\} = \lambda\overline{dist(G(c_x, c_y), o_{NN})} \quad (4.11)$$

また，MAXSUM 問題において，仮想データ v_i の位置として，多様集合内のそれぞれのデータ $o \in S^*_{t_c}$ から最も離れた点を仮定する．勿論このような，仮想データが複数の位置情報を有する仮定は成り立ち得ないが，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 項が取りうる最大値は，以下の式で与えられる．

$$\max_{v_i \in G(c_x,c_y)} \{\sum_{o \in S^*_{t_c}} \{p(o) + 2\lambda dist(v_i, o)\}\} = \sum_{o \in S^*_{t_c}} \{p(o) + 2\lambda\overline{dist(G(c_x, c_y), o)}\} \quad (4.12)$$

これらの式から，MAXMIN 問題，MAXSUM 問題，MMR 問題における空間セルの上界を，それぞれ以下のように計算できる．

$$\overline{d^{min}_r(G(c_x, c_y), S^*_{t_c})} = \frac{1}{2}\overline{p(G(c_x, c_y))} + \frac{1}{2}p(o_{NN}) + \lambda\overline{dist(G(c_x, c_y), o_{NN})} \quad (4.13)$$

$$\overline{d^{sum}_r(G(c_x, c_y), S^*_{t_c})} = |S^*_{t_c}|\overline{p(G(c_x, c_y))} + \sum_{o \in S^*_{t_c}} \{p(o) + 2\lambda\overline{dist(G(c_x, c_y), o)}\} \quad (4.14)$$

$$\overline{d^{mmr}_r(G(c_x, c_y), S^*_{t_c})} = (1 - \lambda)\overline{p(G(c_x, c_y))} + \lambda\overline{dist(G(c_x, c_y), S^*_{t_c})} \quad (4.15)$$

次に，Algorithm 11 の 16 行目における，評価値の上界の減少量について説明する．環境属性 i のデータリスト $G(c_x, c_y).L(i, \overline{c_{z_i}})$ のデータをすべて走査すると，次のデータリストを走査するために $G(c_x, c_y).\overline{c_{z_i}}$ がデクリメントされる．この際，環境属性 i の上界が減少するため，式 (4.9) から，$\overline{p(G(c_x, c_y))}$ は $\Delta_z \cdot q.w_i$ だけ減少する．これにともない，式 (4.13)，(4.14)，(4.15) から，各問題における評価値の上

界 $\overline{d_r(G(c_x, c_y), S^*_{t_c})}$ もまた，それぞれ以下に示す式の値だけ減少する．

$$\Delta_r^{min} = \frac{1}{2} \cdot \Delta_z \cdot q.w_i \tag{4.16}$$

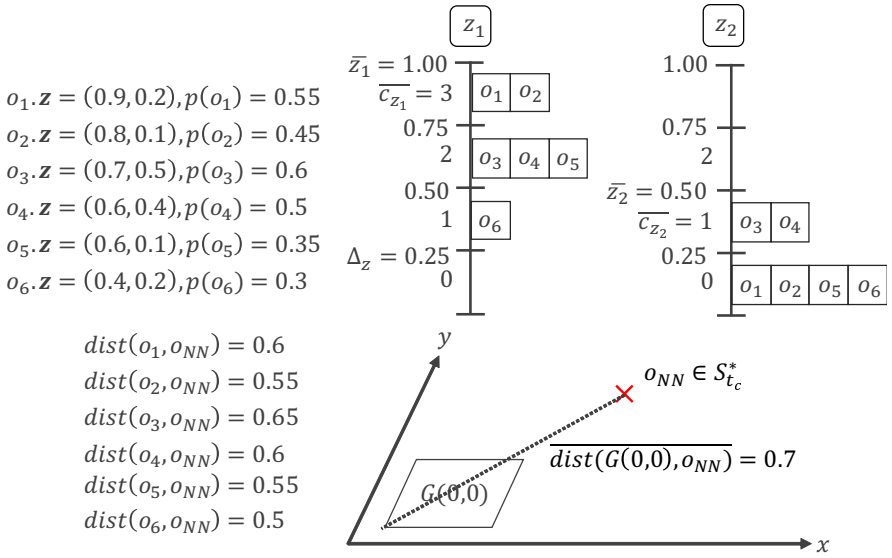
$$\Delta_r^{sum} = |S^*_{t_c}| \cdot \Delta_z \cdot q.w_i \tag{4.17}$$

$$\Delta_r^{mmr} = (1 - \lambda) \cdot \Delta_z \cdot q.w_i \tag{4.18}$$

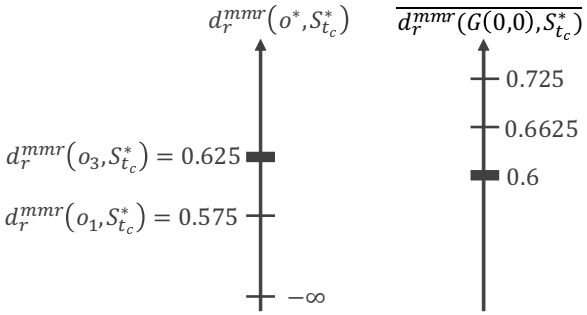
オンラインクエリ処理の例

図 4.4 を用いて Algorithm 11 における，$|S^*_{t_c}| = 1$ の状態における空間セル内のデータの走査の例を説明する．ここでは $w_1 = w_2 = 0.5$，$\lambda = 0.5$ の MMR 問題を考え，Algorithm 11 の 5 行目においてすべての空間セルがヒープに挿入された直後の状態を想定し，$d_r^{mmr}(o^*, S^*_{t_c})$ は初期値 $-\infty$ であるとする．最初にヒープから取り出された空間セル $G(0,0)$ について，データが図 4.4(a) のように格納されているとする．多様集合内のデータに依存する項が取りうる最大値は，$\overline{dist(G(0,0), o_{NN})} = 0.7$ とする．この値は，データを走査する過程で一定である．

まず，環境属性 1 のデータを含むリストのうち最大のものである，リスト $G(0,0).L(1,3)$ からデータ o_1 を取得する．現時点では，$o^* = o_1$ であり，データ o_1 の評価値 0.575 を記憶しておく．また，空間セルの上界は式 (4.9), (4.11), (4.15) から $\overline{d_r^{mmr}(G(0,0), S^*_{t_c})} = 0.725$ であり，依然として空間セル $G(0,0)$ 内にデータ o_1 より評価値の高いデータが存在する可能性があるため，データの走査を続ける．次に，環境属性 2 のデータを含むリストのうち最大のものである，リスト $G(0,0).L(2,1)$ からデータ o_3 を取得し，評価値を計算する．ここで，$d_r^{mmr}(o_3, S^*_{t_c}) = 0.625 > d_r^{mmr}(o_1, S^*_{t_c})$ であるため，o^* を o_3 に更新する．再びリスト $G(0,0).L(1,3)$ からデータ o_2 を取得し，評価値を計算する．ここで，$G(0,0).L(1,3)$ のデータがすべて走査されたため，環境属性 1 の次の走査データリスト番号 $\overline{c_{z_1}}$ がデクリメントされる．それと同時に，式 (4.18) によって算出される値だけ空間セルの上界は減少し，$\overline{d_r^{mmr}(G(0,0), S^*_{t_c})} = 0.6625$ となる．さらに，リスト $G(0,0).L(2,1)$ からデータ o_4 を取得し評価値を計算すると，次のリスト $G(0,0).L(2,1)$ も空となるため，上界が $\overline{d_r^{mmr}(G(0,0), S^*_{t_c})} = 0.6$ に更新される．ここで，上界 0.6 が最大の評価値 0.625 を下回るため，o_3 よりも評価値



(a) 空間セル$G(0,0)$が管理するデータ



(b) 空間セル$G(0,0)$の上界と暫定の最大の評価値の推移

図 4.4: 空間セル内のデータの走査の例

の大きいデータは空間セル $G(0,0)$ 内には存在しないと判断でき，空間セル $G(0,0)$ の探索を打ち切ることができる．よって，空間セル $G(0,0)$ 内に残っているデータ o_5, o_6 は走査する必要はない．

計算量分析

格子グリッドベースのデータ構造を利用した FindNextObject アルゴリズムの時間計算量について分析する．アルゴリズム実行時の多様集合のサイズを $|S^*_{t_c}| = k'$,

空間セルの数を s とする．まず最初に，すべての空間セルの評価値の上界を計算する．このためには，空間セル内のデータが取りうるスコアの上界，および空間セル内のデータと多様集合内のデータとの距離の取りうる上界の計算が必要であり，この処理の計算量は $s \cdot \{1 + (k' - 1)\} = sk'$ となる．次に，すべての空間セルについて評価値の上界をキーとしたヒープソートを行う必要があり，この処理の計算量は $s\log(s)$ である．最後に，すべての空間セルの評価値の上界が暫定の最大の評価値を下回るまで，走査するデータについて評価値の計算が必要となる．このとき走査するデータの数を n^* とすると，計算量は n^*k' となる．よって，格子グリッドベースのデータ構造を利用した FindNextObject アルゴリズムの時間計算量は，$O(k'(s + n^*) + s\log(s))$ である．

4.4.3 項で述べた通り，モニタリングの対象のデータ集合を O としたとき，ベースライン手法における FindNextObject アルゴリズムの計算量は $O(k'|O|)$ である．多くの場合，空間セルの数 s や走査されるデータの数 n^* は，$|O|$ に比べ大幅に小さい．このため，提案手法の計算量は，ベースライン手法の計算量に比べて小さくなる．

4.6 性能評価

本節では，本章における提案手法の性能を評価する．表 4.1 は各パラメータの値を示し，太字はデフォルト値とする．

4.6.1 データセット

本実験では，ウィンドウサイズが W の時間ベースのスライディングウィンドウを想定する．各時刻に N 個のデータがシステムに到着するものとし，スライディングウィンドウを 100 回スライドさせる．データの位置情報は，各次元の値が区間 $[0.0, 1.0]$ 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 4.5 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，4 次元から 10 次元まで設定した．これらの環境属性値の分布は，スライディングウィ

表 4.1: パラメータ

パラメータ	値
k（多様集合の大きさ）	5∼50 (**25**)
δ（差分の制限）	0∼25 (**5**)
λ（多様性の重要度）	0.0∼1.0 (**0.5**)
W（ウィンドウサイズ）	10∼100 (**10**)
d（環境属性の次元数）	4∼10 (**4**)
N（各時刻に到着するデータの数）	10K∼200K (**50K**)
w_i ($\boldsymbol{w}$ の各要素)	0.0∼1.0

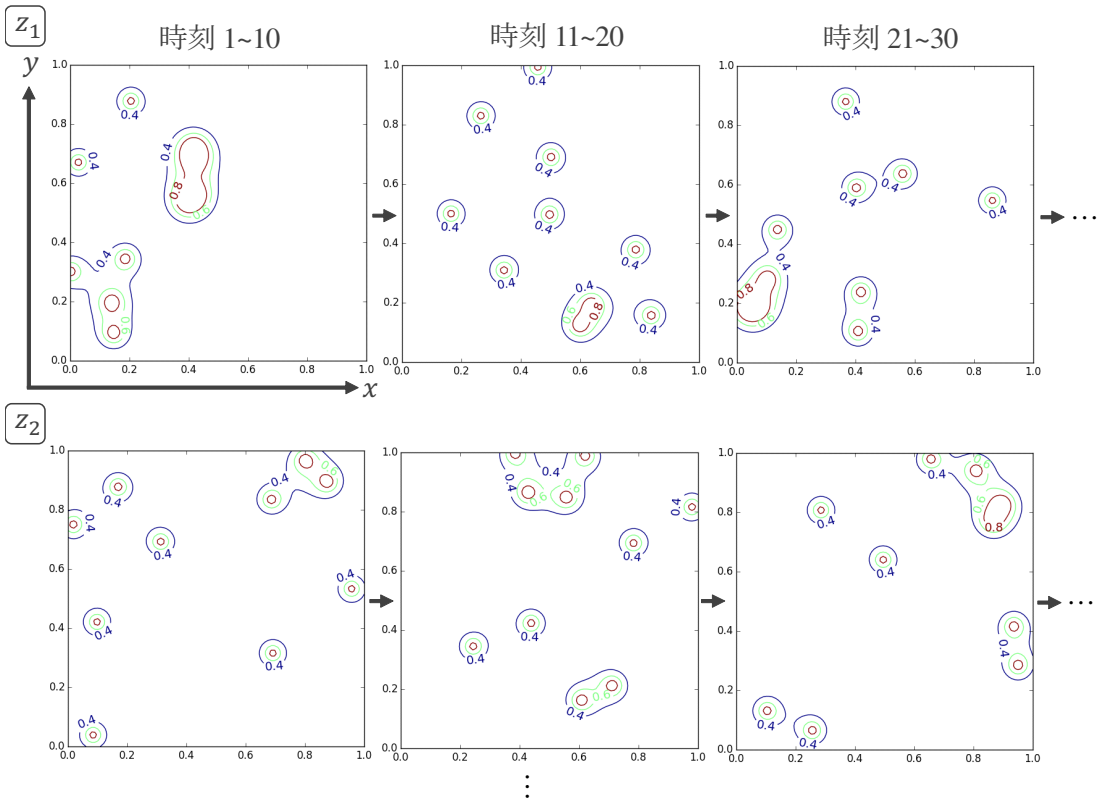


図 4.5: 環境属性値の分布

ンドウが 10 回スライドするたびに変化する．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，$N(0, 0.03)$ の正規分布に従う正規乱数を加算した．

4.6.2 比較手法

提案手法（以降のグラフ中では‘Grid-based’と表記）を，4.4節で説明したベースライン手法（‘Baseline’）および4.5節の冒頭で説明したクラスタベース手法（‘Cluster-based’）と比較した．

4.6.3 設定

すべてのアルゴリズムをJava8で実装し，Intel Xeon E5-2643 v2 @ 3.50GHz with 48.0 GB RAMを搭載するUbuntu 14.04.5 LTSで動作する計算機上で実験した．

実験においては，各時刻における多様集合，比較手法におけるクラスタベースのデータ構造および提案手法における格子グリッドベースのデータ構造の更新に要した計算時間の平均値を測定した．各実験で，重みベクトル $q.\boldsymbol{w}$ の各重みを $[0.0, 1.0]$ の範囲からランダムに設定した50個のクエリを作成し，結果の平均値を示す．

4.6.4 k, δ, λ の影響

はじめに，提案手法におけるグリッドベースのデータ構造，および比較手法におけるクラスタベースのデータ構造の更新に要する計算時間を図4.6に示す．k，δ および λ はすべてクエリパラメータであり，スライディングウィンドウ上のデータ集合に対して影響を及ぼさない．このため，これらのデータ構造を更新するのに要する計算時間は，k，δ および λ の変化に対して一定である．図4.6から，グリッドベースのデータ構造の更新時間は約20ミリ秒，クラスタベースのデータ構造の更新時間は約900ミリ秒と，提案データ構造の更新時間は比較手法に比べて大幅に小さいことが分かる．これは，4.5節の冒頭で説明したように，提案手法では新たに到着したデータを包含する空間セルを定数時間で決定できるためである．しかし比較手法では，新たに到着したデータを包含するクラスタを決定する計算コストは，クラスタの数に比例して大きくなってしまう．

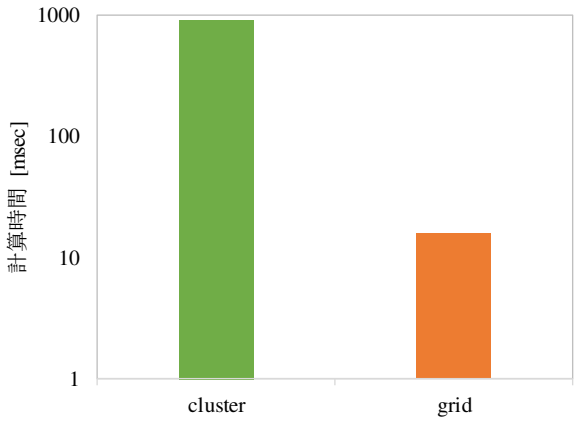
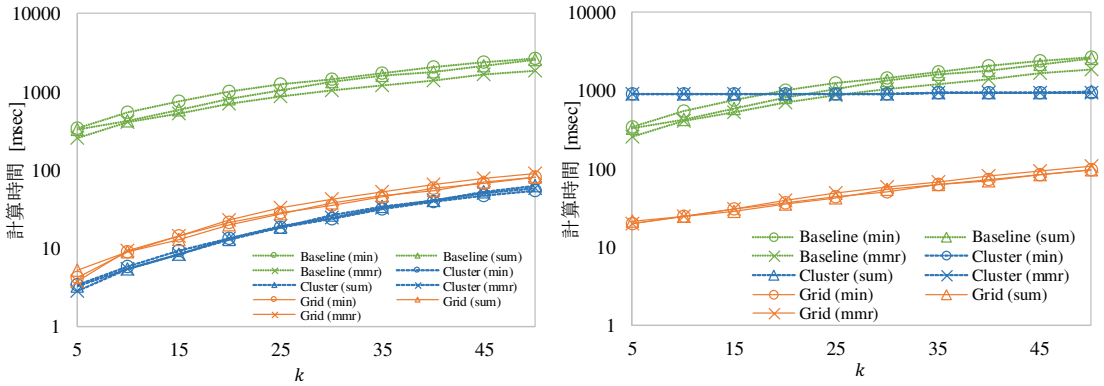


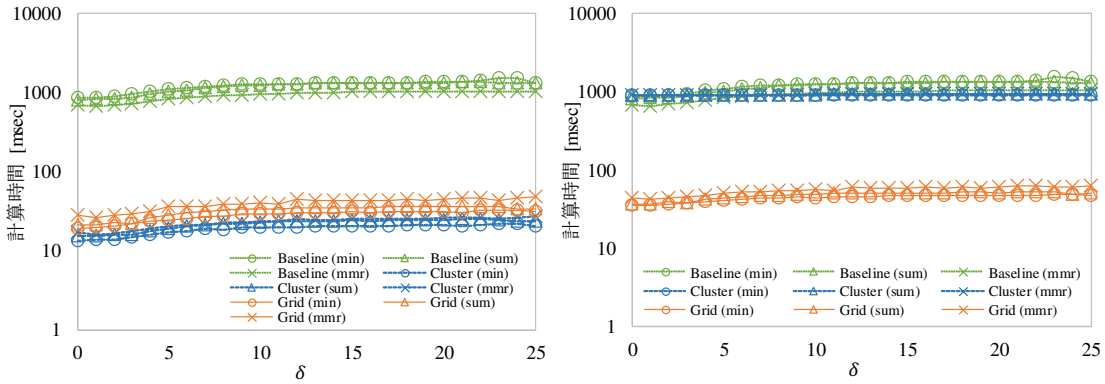
図 4.6: 各データ構造の更新時間



(a) 多様集合の更新時間　　(b) 更新の総計算時間

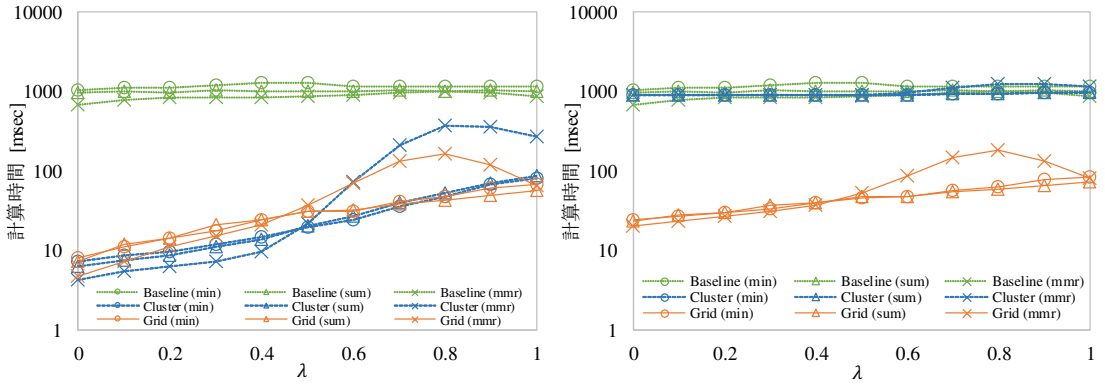
図 4.7: 多様集合の大きさ k の影響

次に，k および δ を変化させた場合の多様集合の更新に要する時間を，それぞれ図 4.7 (a) および 4.8 (a) に示す．また，比較手法および提案手法で用いるデータ構造の更新に要する時間も含めた総計算時間を，それぞれ図 4.7 (b) および 4.8 (b) に示す．図 4.7，4.8 より，すべての手法および各問題において，k および δ が大きくなるほど計算時間が増加していることが分かる．これは，k および δ が大きくなるほど，多様集合を改善するために新しいデータの追加，および既存のデータとの交換を行う回数が増加するためである．提案手法および比較手法は走査するデータの数を削減できるため，ベースライン手法に比べて多様集合の更新時間は大幅



(a) 多様集合の更新時間　　(b) 更新の総計算時間

図 4.8: 差分の制限 δ の影響



(a) 多様集合の更新時間　　(b) 更新の総計算時間

図 4.9: 多様性の重要度 λ の影響

に小さくなる．しかし，図 4.6 に示すようにクラスタベースのデータ構造の更新時間は大きく，結果として k が小さい場合，比較手法における総計算時間はベースライン手法の計算時間を上回っている．一方で，提案手法は提案データ構造および多様集合の更新時間がともに小さい．結果，総計算時間はベースライン手法では最大で約 1900 ミリ秒，比較手法では最大で約 660 ミリ秒であるのに対し，提案手法では最大でも約 110 ミリ秒と他の手法に比べて大幅に小さいことが分かる．

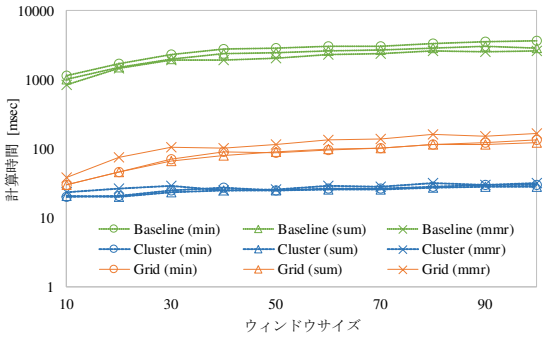
最後に，λ を変化させた場合の計算時間を図 4.9 に示す．図 4.9 より，すべての

手法および各問題において，λが大きくなるほど計算時間が増加していることが分かる．特に，提案手法および比較手法におけるλの影響は，ベースライン手法に比べて大きいことが分かる．提案手法および比較手法では，空間セルおよびクラスタ内のデータが取りうるスコアの上界を計算することで走査するデータの数を削減でき，結果として効率的に多様集合を更新できる．そのため，λが大きくなり空間的多様性の重要度が大きくなる一方で，データのスコアの重要度が小さくなるほどこれらの手法の有効性は小さくなる．特に，MMR問題においてはλの影響が大きく，$\lambda = 0.6$より大きな値で多様集合の更新に要する計算時間が増加し，$\lambda = 0.9, 1.0$の場合はやや小さくなっている．しかしながら，総計算時間はベースライン手法では最大で約1200ミリ秒，比較手法では最大で約1100ミリ秒であるのに対し，提案手法では最大でも200ミリ秒と他の手法に比べて大幅に小さいことが分かる．

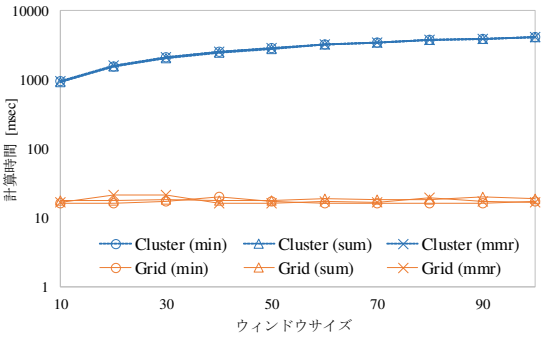
4.6.5 W, d, Nの影響

パラメータW，dおよびNはシステムが管理するスライディングウィンドウ上のデータセットに関連するパラメータであるため，これらが変化すると，提案手法および比較手法におけるデータ構造の更新に要する時間は変化する．本項では，それぞれのパラメータを変化させた場合のデータ構造の更新に要する時間を示す．

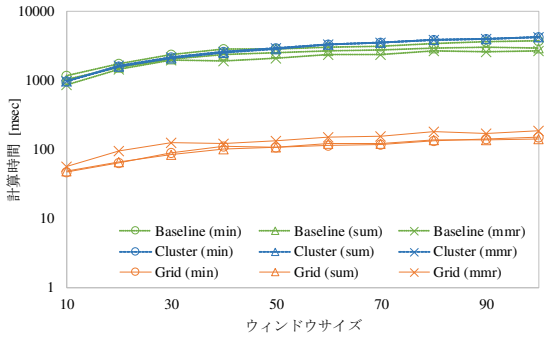
まず，ウィンドウサイズWを変化させた場合の計算時間を図4.10に示す．図4.10 (a)から，すべての手法および各問題において，Wが大きくなるほど多様集合の更新時間は増加することが分かる．これは，スライディングウィンドウ上のモニタリングの対象のデータの数が増加すると，新しいデータを追加もしくは交換するために走査するデータの数が増加するためである．また，図4.10 (b)から，Wが大きくなるほどクラスタベースのデータ構造の更新時間が増加することが分かる．これは，スライディングウィンドウ上のモニタリングの対象のデータの数が増加すると，クラスタの数も増加し，結果として各データを包含するクラスタを決定するための計算コストが大きくなるためである．一方で，Wが大きくなってもグリッドベースのデータ構造の更新時間は一定である．これは，各データを包



(a) 多様集合の更新時間



(b) データ構造の更新時間

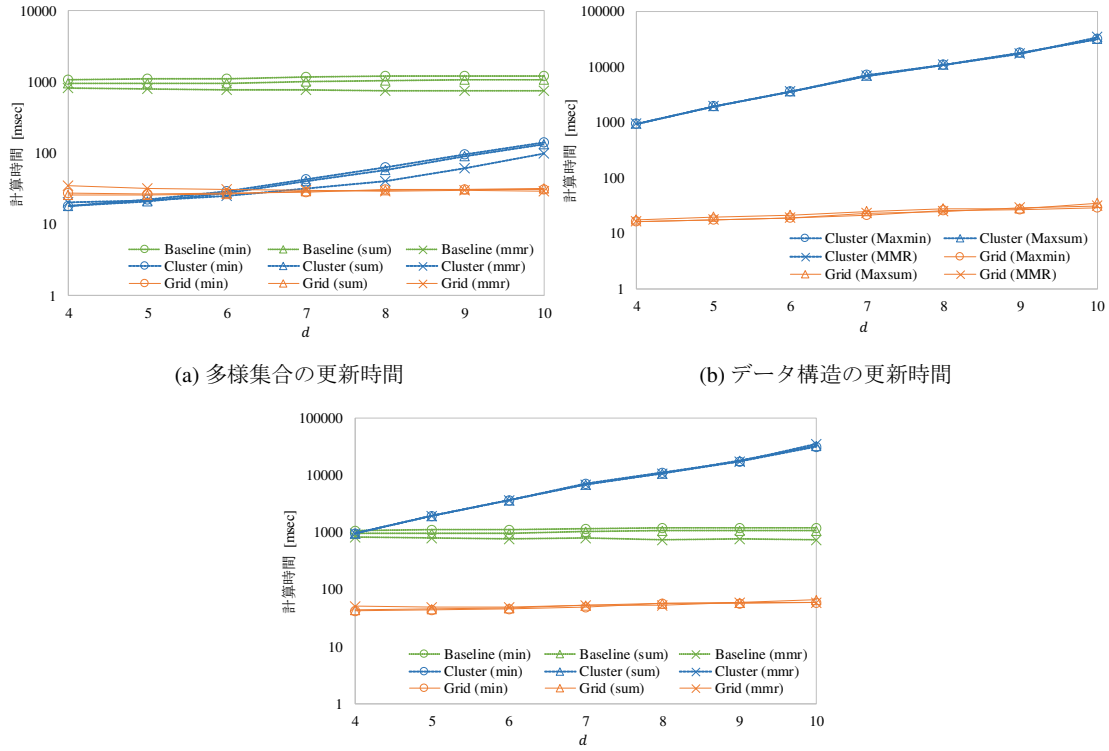


(c) 更新の総計算時間

図 4.10: ウィンドウサイズ W の影響

含する空間セルを決定するための計算コストが一定のためである．さらに，各時刻に到着するデータの数は一定であるため，結果としてグリッドベースのデータ構造の更新時間は変化しない．

次に，環境属性の次元数 d を変化させた場合の計算時間を図 4.11 に示す．図 4.11 (a) から，比較手法においてのみ，d が大きくなるほど多様集合の更新時間が増加することが分かる．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同士が類似しにくいためである．結果として，図 4.11 (b) に示すように，クラスタの数が急激に増加し，クラスタベースのデータ構造の更新時間もまた増加してしまう．ここで，クラスタの数を小さく抑制するためにクラスタ半径を大きくした場合，クラスタの数が減少することでデータ構造の更新コストは削減される．しかしこの場合は，第 2，3 章の性能評価で示した通り，クラスタ半径が大きいためクラスタの上界が大きくなることにより，クラスタの数が大きい場合より



(a) 多様集合の更新時間

(b) データ構造の更新時間

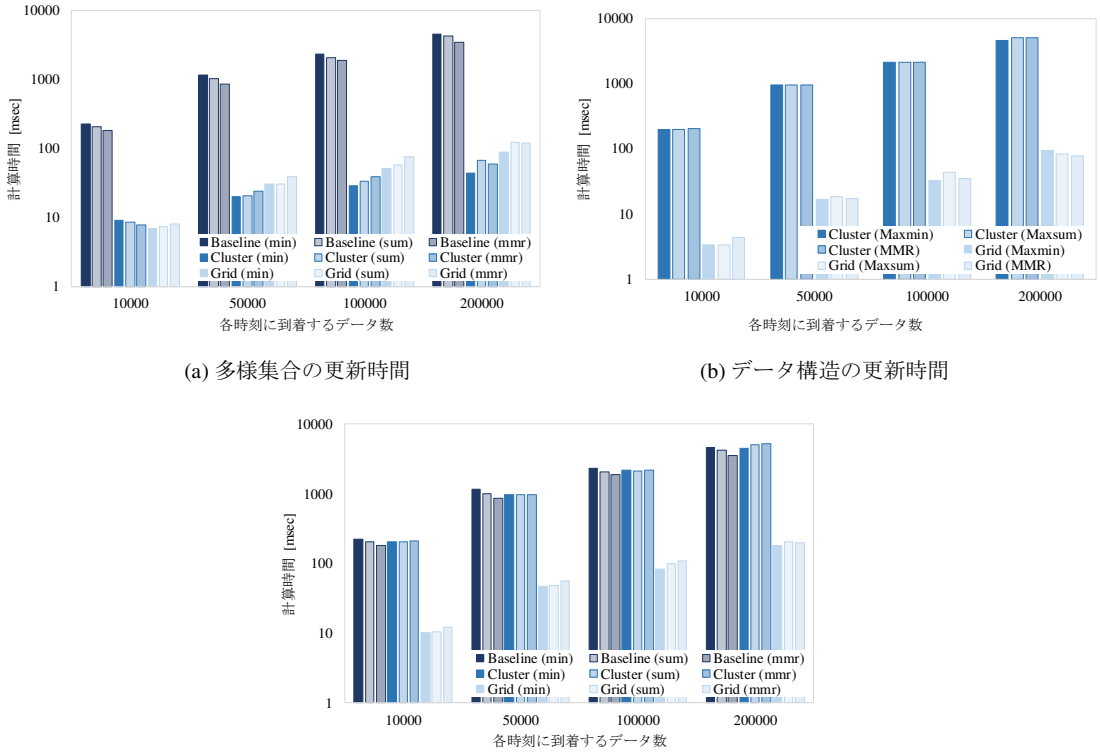
(c) 更新の総計算時間

図 4.11: 環境属性の次元数 d の影響

も走査の対象となるデータの数が増え，多様集合の更新コストが増大する．

最後に，各時刻に到着するデータの数 N を変化させた場合の計算時間を図 4.12 に示す．図 4.12 (a) から，すべての手法および各問題において，N が大きくなるほど多様集合の更新時間は増加することが分かる．さらに，図 4.12 (b) から，比較手法および提案手法におけるデータ構造の更新時間もまた増加することが分かる．これは，N が大きくなると，スライディングウィンドウ上のモニタリングの対象のデータが増加するほか，各時刻に到着するデータの数および削除されるデータの数もまた，それぞれ増加するためである．

以上のように，すべてのパラメータ設定および各問題において，提案手法における総計算時間は他の手法に比べて大幅に小さく，一貫して最も効率的であることが確認できる．具体的には，総計算時間はベースライン手法では最大で約 3200



(a) 多様集合の更新時間

(b) データ構造の更新時間

(c) 更新の総計算時間

図 4.12: 各時刻に到着するデータの数 N の影響

ミリ秒，比較手法では最大で約 26000 ミリ秒であるのに対し，提案手法では最大でも 200 ミリ秒と他の手法に比べて大幅に小さい．

4.7 むすび

本章では，モバイルセンサストリーム環境における継続的な Top-k 検索結果の多様化について取り組み，効率的な多様集合の更新手法を提案した．ベースライン手法では，各時刻にスライディングウィンドウ上のすべてのデータについて評価値を計算し，最適なデータを探索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，スライディングウィンドウ上のデータは更新が容易な格子グリッドを用いて構造化される．各空間セルごとに評価値の上界を計算し，

上界が暫定の最大の評価値を下回った段階で空間セル内のデータの走査を打ち切ることができる．結果として，走査するデータの数を大幅に削減できる．

シミュレーション実験により，提案手法における格子グリッドベースのデータ構造は比較手法におけるクラスタベースのデータ構造と比較して，より短時間で更新できることを確認した．さらに，多様集合の更新処理について，提案アルゴリズムはクラスタベースのアルゴリズムと同程度の計算効率を達成でき，結果的に短時間で多様集合を更新できることを確認した．特に，単位時間あたりに到着するデータの数が大きくなっても短時間で多様集合を更新できるため，センシングに参加する端末保持者が増加した場合も，効率的に多様集合をモニタリングできる．

本章における多様集合モニタリング問題では，スライディングウィンドウ上のすべてのモバイルセンサデータを平等に扱っている．しかし，地理空間上のホットスポットの出現をいち早く検出するために，データの評価値がモニタリングの対象のデータセット内で最大ではなくても，優先的に検索結果に含めるほうが望ましい場合がある．このような要求は，文献 [1, 21, 23, 84, 95] で扱われているような，データの評価値に対して経過時刻に対する減衰係数で重み付けし，現在時刻から時間的に離れたデータほど評価値を小さく見積もることで実現できる．そこで，このような減衰係数を用いた最適化問題を再定義し，その際の効率的な多様集合の更新手法を設計する予定である．

第5章　結論

5.1　本論文のまとめ

本論文では，モバイルセンサデータベースにおける Top-k 検索結果の多様化手法について議論した．

まず，第 1 章では，近年注目されているユーザ参加型センシングと，ユーザ参加型センシングによって収集されるセンサデータからなるモバイルセンサデータベースについて述べた．また，モバイルセンサデータから，ユーザが注目すべき地理空間上の領域であるホットスポットを効果的に検出するために，モバイルセンサデータベースにおける Top-k 検索結果の多様化が重要であることを明らかにし，クエリ処理における問題点について述べた．

第 2 章では，過去に収集され蓄積されたモバイルセンサデータベースにおいて，事前にクラスタリング処理を施すことによって，短時間で多様集合を計算する手法を提案した．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，空間的に近接し，かつ環境属性値が似ているデータ同士をクラスタ化し，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．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から，クラスタ内のデータが取りうる評価値の上界を計算する．これにより，計算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．提案手法の有効性を示すために，シミュレーション実験による性能評価を行った．その結果より，提案手法はすべてのデータを走査する単純なグリーディアルゴリズムと比較して，同一の多様集合をより短時間で取得できることを確認した．さらに，要求するデー

タの数kやデータセットサイズNを変化させた場合も，提案手法は比較手法に比べ，計算時間およびディスクIOコストを削減できることを確認した．

第3章では，第2章の手法における，環境情報が高次元である場合の計算効率の低下の影響を緩和する手法を提案した．提案手法のオフライン事前階層的クラスタリングでは，空間的に近接するすべてのデータはある1つの上位クラスタのメンバとし，上位クラスタの配下でメンバデータの環境属性値の類似度に基いてさらに下位クラスタに分割する．空間的に近接するデータからなる上位クラスタの数は，環境属性の次元数の増加に非依存であり，一貫して小さく保たれる．提案手法のオンラインクエリ処理では，階層クラスタ構造を利用し，上位クラスタ内のデータが取りうる評価値の上界を計算することで，上位クラスタごとに走査対象のデータを絞り込める．そのため提案手法では，第2章で提案した手法と比べて，走査すべきデータをより短時間で絞り込める．提案手法の有効性を示すために，シミュレーション実験による性能評価を行った．その結果より，提案手法は第2章で提案した手法と比較して，モバイルセンサデータが高次元の場合でも多様集合をより短時間で取得できることを確認した．さらに，提案手法は比較手法よりも，環境属性値半径を小さくした場合のクラスタの増加に伴う計算コストの増加を緩和できることを確認した．

第4章では，モバイルセンサデータが時々刻々と到着するモバイルセンサストリーム環境において，効率的に多様集合をモニタリングする手法を提案した．提案手法では，スライディングウィンドウ上のデータを格子グリッドベースのデータ構造で管理する．格子グリッドの空間セルは，環境属性ごとにデータのリストを保持しており，それぞれのリストは環境属性値が一定の範囲内であるデータを管理する．それぞれのデータを被覆する空間セルおよび管理するリストは定数時間で決定できるため，提案データ構造は追加・削除されるデータについて短時間で更新できる．提案アルゴリズムでは，最適なデータを探索する際，空間セル内のデータがとりうるスコアの上界を計算できる．また，この上界は空間セル内のデータの走査が進むにつれて減少し，これにより空間セル内のデータの走査を途中で打ち切り，次の空間セルの走査に進める．結果として，走査するデータの数を大幅に削減しつつ，モニタリングの対象のすべてのデータを走査することで得

られる多様集合と，同一の検索結果を取得できる．提案手法の有効性を示すために，シミュレーション実験による性能評価を行った．その結果より，提案手法における格子グリッドベースのデータ構造は，第 2 章で提案した手法におけるクラスタベースのデータ構造と比較して，より短時間で更新できることを確認した．さらに，多様集合の更新処理について，提案アルゴリズムは第 2 章で提案したアルゴリズムと同程度の計算効率を達成でき，結果的に短時間で多様集合を更新できることを確認した．

データのスコアとデータ間の多様性を考慮した多様集合の計算は，従来の Top-k 検索に対して指摘されている検索結果の冗長性を解決する手法であり，これまで数多くの研究が行われている．しかし，環境モニタリングを目的として Top-k 検索結果の多様化をモバイルセンサデータベースに適用した研究は，筆者が知る限りこれまでに存在しなかった．本論文で提案した手法は，すべてのデータを複数回走査する必要のある既存手法で得られる多様集合と同一の結果を，より短時間で計算できる．本論文の提案手法は，任意の環境属性値に対する興味，地理的多様性の重要度に対応できる．さらに同様のアプローチを用いて，任意の最適化問題において短時間で多様集合を計算できる．従って本研究の成果は，1.1 節で例示したようなアプリケーションを，一般ユーザが所持する端末のセンサデータを用いて実現することに向けて，大きな進展をもたらすものである．

5.2 検討課題

本論文における提案手法について，さらなる応用や，実環境への適用を考慮すると，各章のむすびで述べたものに加え，解決しなければならない課題がいくつか残されている．本節では，多様化問合せの可用性を向上するため，また，想定するアプリケーションに対して良質な多様集合を取得するために考慮すべき技術課題について述べる．

5.2.1　適切な多様性の重要度 λ の自動設定

Top-k 検索結果の多様化問合せのクエリパラメータである多様性の重要度 λ は，多様集合の地理的な分散度合いを決定するもので，ユーザが任意の値を設定できる．本論文では，λ を変化させることで異なる多様集合が得られるため，ユーザが λ を手動で変えながら繰り返し多様化クエリを発行することで，ユーザの目的に合致した結果を得ることを想定している．しかし，モバイルセンサデータやホットスポットの空間的な分布によって適切な λ は異なり，手動での λ の設定はユーザにとって難しいと考えられる．

一方，文献 [5, 35] では，それぞれ Web 文書検索，商品検索において，多様化の対象となるデータ集合の分布や，暫定的な多様集合の構成から多様性の重要度を自動で調整する手法を提案している．モバイルセンサデータの空間的分布や属性値の分布，および暫定的な多様集合の構成から，クエリパラメータ λ を自動で設定できる枠組みは，本研究においても適用できる．例えば，すべてのデータのスコアの最大値と，すべてのデータ間の距離の最大値は一般的に等しくないため，これらが等しく評価されるような λ をシステム側で計算することで，ユーザの興味と地理的な分散度合いをバランス良く考慮した多様集合が取得できると考えられる．今後，具体的な自動設定のアルゴリズムの設計について検討する予定である．

5.2.2　外れ値を考慮した多様化問題の定義

ユーザ参加型センシングでは，ユーザが普段通りの生活をしながらセンシングを行う場合があるため，例えば携帯端末を鞄の中に入れたままセンシングするといった状況が想定される．このような場合，センシングしたい環境情報が正しく計測できず，ある環境属性値について誤差の大きい外れ値が観測される可能性がある．このような状況では，外れ値によってデータのスコアが引き上げられ，多様集合が外れ値をとるデータを含みやすくなる．外れ値に起因して，本来注目すべきホットスポットではない領域で生成されたデータが検索結果に含まれることは，想定するアプリケーションにおいて好ましくない．

一方，データセットから外れ値を示すデータを検出する研究はこれまでに多数

行われている [3, 22, 48, 57]．これらの研究で提案されている手法を検索の前処理として実行し，データごとに得られた外れ値と判定される確率や確信度は，多様集合の計算の際に利用できる．例えば，前処理の段階で外れ値ではないと判定されたデータについては，クエリ処理の際に外れ値の判定を行う必要はなく，計算コストの削減が見込まれる．今後，外れ値を考慮した多様集合の計算手法の具体的なアルゴリズムの設計について検討する予定である．

謝辞

本研究を推進するにあたり，懇切なる御指導と惜しみない御助言を頂きました大阪大学大学院情報科学研究科マルチメディア工学専攻 原 隆浩教授に謹んで御礼申し上げます.

本研究において，直接の御指導，御助言，御討論を頂きました大阪大学西尾章治郎総長に深く御礼申し上げます.

本論文をまとめるにあたり，大変有益な御指導と御助言を多数賜りました大阪大学大学院情報科学研究科マルチメディア工学専攻 鬼塚 真教授，藤原 融教授に心より感謝申し上げます.

講義，学生生活を通じて，学問に取り組む姿勢をご教授頂きました大阪大学大学院情報科学研究科マルチメディア工学専攻　下條真司教授，松下康之教授，萩田紀博教授に厚く感謝申し上げます.

本研究において，ともに研究を進め，直接の御助言，御協力，御討論を頂きましたミズーリ工科大学 Sanjay Kumar Madria 教授に深く御礼申し上げます.

本研究において，多大なる御助言，御協力，御支援を頂きました独立行政法人情報通信研究機構　寺西裕一氏，大阪大学データビリティフロンティア機構 春本 要教授，大阪大学サイバーメディアセンター 義久智樹准教授，大阪大学大学院情報科学研究科マルチメディア工学専攻 前川卓也准教授，天方大地助教，白川真澄招聘研究員に深謝致します.

本研究において，ともに研究を進め，多大なるご協力を頂きました大阪大学大学院情報科学研究科マルチメディア工学専攻 高 博奇氏，西尾俊哉氏，林田秀平氏，田 一鳴氏，下岸北斗氏，曹 文祺氏，加藤慎也氏，呂 遠氏に深く御礼申し上げます.

本研究を進めるにあたり，多くの御討論や御助言を頂きました大阪大学大学院情報科学研究科マルチメディア工学専攻 原研究室の諸氏に心より感謝申し上げます.

最後に，私のこれまでの人生，そして研究生活を送る上で，暖かい支援と理解を頂きました家族や友人に，心から感謝致します.

参考文献

[1] Aggarwal, C. C., and Philip, S. Y.: A framework for clustering uncertain data streams, in *Proc. Int. Conf. on Data Engineering (ICDE 2008)*, pp. 150–159 (2008).

[2] Alamri, A., Ansari, W. S., Hassan, M. M., Hossain, M. S., Alelaiwi, A., and Hossain, M. A.: A survey on sensor-cloud: architecture, applications, and approaches, *Journal of Distributed Sensor Networks*, Vol. 9, No. 2 (2013).

[3] Albanese, A., Pal, S. K., and Petrosino, A.: Rough sets, kernel set, and spatiotemporal outlier detection, *IEEE Trans. on Knowledge and Data Engineering*, Vol. 26, No. 1, pp. 194–207 (2014).

[4] Amagata, D., and Hara, T.: Diversified set monitoring over distributed data streams, in *Proc. Int. Conf. on Distributed and Event-based Systems (DEBS 2016)*, pp. 1–12 (2016).

[5] Angel, A., and Koudas, N.: Efficient diversity-aware search, in *Proc. Int. Conf. on Management of Data (ACM SIGMOD 2011)*, pp. 781–792 (2011).

[6] Babcock, B., and Olston, C.: Distributed top-k monitoring, in *Proc. Int. Conf. on Management of Data (ACM SIGMOD 2003)*, pp. 28–39 (2003).

[7] Beckmann, N., Kriegel, H.-P., Schneider, R., and Seeger, B.: The R*-tree: an efficient and robust access method for points and rectangles, in *Proc. Int. Conf. on Management of Data (ACM SIGMOD 1990)*, Vol. 19, pp. 322–331 (1990).

[8] Beeferman, D., and Berger, A.: Agglomerative clustering of a search engine query log, in *Proc. Int. Conf. on Knowledge Discovery and Data Mining (ACM SIGKDD 2000)*, pp. 407–416 (2000).

[9] Bentley, J. L.: Multidimensional binary search trees used for associative searching, *Journal of ACM Communications*, Vol. 18, No. 9, pp. 509–517 (1975).

[10] Beygelzimer, A., Kakade, S., and Langford, J.: Cover trees for nearest neighbor, in *Proc. Int. Conf. on Machine Learning (ICML 2006)*, pp. 97–104 (2006).

[11] Bingham, E., and Mannila, H.: Random projection in dimensionality reduction: applications to image and text data, in *Proc. Int. Conf. on Knowledge Discovery and Data Mining (ACM SIGKDD 1999)*, pp. 245–250 (1999).

[12] Boim, R., Milo, T., and Novgorodov, S.: Diversification and refinement in collaborative filtering recommender, in *Proc. Int. Conf. on Information and Knowledge Management (ACM CIKM 2011)*, pp. 739–744 (2011).

[13] Brown, A., Franken, P., Bonner, S., Dolezal, N., and Moross, J.: Safecast: successful citizen-science for radiation measurement and communication after Fukushima, *Journal of Radiological Protection*, Vol. 36, No. 2 (2016).

[14] Bruno, N., Chaudhuri, S., and Gravano, L.: Top-k selection queries over relational databases: mapping strategies and performance evaluation, *ACM Trans. on Database Systems (TODS)*, Vol. 27, No. 2, pp. 153–187 (2002).

[15] Capannini, G., Nardini, F. M., Perego, R., and Silvestri, F.: Efficient diversification of web search results, *Proceedings of the VLDB Endowment*, Vol. 4, No. 7, pp. 451–459 (2011).

[16] Carbonell, J., and Goldstein, J.: The use of MMR, diversity-based reranking for reordering documents and producing summaries, in *Proc. Int. Conf. on Research and Development in Information Retrieval (ACM SIGIR 1998)*, pp. 335–336 (1998).

[17] Carvalho, C., Gomes, D. G., Agoulmine, N., and de Souza, J. N.: Improving prediction accuracy for WSN data reduction by applying multivariate spatio-temporal correlation, *Journal of Sensors*, Vol. 11, No. 11, pp. 10010–10037 (2011).

[18] Catallo, I., Ciceri, E., Fraternali, P., Martinenghi, D., and Tagliasacchi, M.: Top-k diversity queries over bounded regions, *ACM Trans. on Database Systems (TODS)*, Vol. 38, No. 2, pp. 1–44 (2013).

[19] Chaudhuri, S., and Dayal, U.: An overview of data warehousing and OLAP technology, *Proc. Int. Conf. on Management of Data (ACM SIGMOD 1997)*, Vol. 26, No. 1, pp. 65–74 (1997).

[20] Chen, L., and Cong, G.: Diversity-aware top-k publish/subscribe for text stream, in *Proc. Int. Conf. on Management of Data (ACM SIGMOD 2015)*, pp. 347–362 (2015).

[21] Chen, L., Cong, G., Cao, X., and Tan, K.-L.: Temporal spatial-keyword top-k publish/subscribe, in *Proc. Int. Conf. on Data Engineering (ICDE 2015)*, pp. 255–266 (2015).

[22] Cheng, T., and Li, Z.: A multiscale approach for spatio-temporal outlier detection, *Trans. on GIS*, Vol. 10, No. 2, pp. 253–263 (2006).

[23] Cormode, G., Shkapenyuk, V., Srivastava, D., and Xu, B.: Forward decay: a practical time decay model for streaming systems, in *Proc. Int. Conf. on Data Engineering (ICDE 2009)*, pp. 138–149 (2009).

[24] Dai, R., and Akyildiz, I. F.: A spatial correlation model for visual information in wireless multimedia sensor networks, *IEEE Trans. on Multimedia*, Vol. 11, No. 6, pp. 1148–1159 (2009).

[25] Ding, C., and He, X.: K-means clustering via principal component analysis, in *Proc. Int. Conf. on Machine Learning (ICML 2004)*, p. 29 (2004).

[26] Drosou, M., and Pitoura, E.: Diversity over continuous data., *Journal of IEEE Data Eng. Bull.*, Vol. 32, No. 4, pp. 49–56 (2009).

[27] Drosou, M., and Pitoura, E.: Dynamic diversification of continuous data, in *Proc. Int. Conf. on Extending Database Technology (EDBT 2012)*, pp. 216–227 (2012).

[28] Drosou, M., and Pitoura, E.: Diverse set selection over dynamic data, *IEEE Trans. on Knowledge and Data Engineering*, Vol. 26, No. 5, pp. 1102–1116 (2014).

[29] Drosou, M., Stefanidis, K., and Pitoura, E.: Preference-aware publish/subscribe delivery with diversity, in *Proc. Int. Conf. on Distributed and Event-based Systems (DEBS 2009)*, pp. 1–12 (2009).

[30] Estrin, D.: Participatory sensing: applications and architecture, *IEEE Internet Computing*, Vol. 14, No. 1, pp. 12–42 (2010).

[31] Fraternali, P., Martinenghi, D., and Tagliasacchi, M.: Top-k bounded diversification, in *Proc. Int. Conf. on Management of Data (ACM SIGMOD 2012)*, pp. 421–432 (2012).

[32] Gieseke, F., Heinermann, J., Oancea, C., and Igel, C.: Buffer kd trees: processing massive nearest neighbor queries on GPUs, in *Proc. Int. Conf. on Machine Learning (ICML 2014)*, pp. 172–180 (2014).

[33] Gkorgkas, O., Vlachou, A., Doulkeridis, C., and Nørvåg, K.: Finding the most diverse products using preference queries., in *Proc. Int. Conf. on Extending Database Technology (EDBT 2015)*, pp. 205–216 (2015).

[34] Gollapudi, S., and Sharma, A.: An axiomatic approach for result diversification, in *Proc. Int. Conf. on World Wide Web (WWW 2009)*, pp. 381–390 (2009).

[35] Hasan, M., Kashyap, A., Hristidis, V., and Tsotras, V.: User effort minimization through adaptive diversification, in *Proc. Int. Conf. on Knowledge Discovery and Data Mining (ACM SIGKDD 2014)*, pp. 203–212 (2014).

[36] Hasan, M., Mueen, A., and Tsotras, V.: Distributed diversification of large datasets, in *Proc. Int. Conf. on Cloud Engineering (IC2E 2014)*, pp. 67–76 (2014).

[37] Hasenfratz, D., Saukh, O., Walser, C., Hueglin, C., Fierz, M., Arn, T., Beutel, J., and Thiele, L.: Deriving high-resolution urban air pollution maps using mobile sensor nodes, *Journal of Pervasive and Mobile Computing*, Vol. 16, Part B, pp. 268–285 (2015).

[38] He, B., Li, Y., Huang, H., and Tang, H.: Spatial–temporal compression and recovery in a wireless sensor network in an underground tunnel environment, *Journal of Knowledge and Information Systems*, Vol. 41, No. 2, pp. 449–465 (2014).

[39] Houle, M. E., and Nett, M.: Rank cover trees for nearest neighbor search, in *Proc. Int. Conf. on Similarity Search and Applications*, pp. 16–29 (2013).

[40] Ilyas, I. F., Beskales, G., and Soliman, M. A.: A survey of top-k query processing techniques in relational database systems, *ACM Trans. on Computing Surveys (CSUR)*, Vol. 40, No. 4, pp. 1–58 (2008).

[41] Jabeur, N., Zeadally, S., and Sayed, B.: Mobile social networking applications, *ACM Communications*, Vol. 56, No. 3, pp. 71–79 (2013).

[42] Jiang, H., Cheng, J., Wang, D., Wang, C., and Tan, G.: Continuous multi-dimensional top-k query processing in sensor networks, in *Proc. Int. Conf. on Computer Communications (INFOCOM 2011)*, pp. 793–801 (2011).

[43] 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. on Parallel and Distributed Systems*, Vol. 23, No. 9, pp. 1668–1680 (2012).

[44] Kanungo, T., Mount, D. M., Netanyahu, N. S., Piatko, C. D., Silverman, R., and Wu, A. Y.: An efficient k-means clustering algorithm: analysis and implementa-

tion, *IEEE Trans. on pattern analysis and machine intelligence*, Vol. 24, No. 7, pp. 881–892 (2002).

[45] Khan, W. Z., Xiang, Y., Aalsalem, M. Y., and Arshad, Q.: Mobile phone sensing systems: a survey, *IEEE Communications Surveys & Tutorials*, Vol. 15, No. 1, pp. 402–427 (2013).

[46] Kontaki, M., Papadopoulos, A. N., and Manolopoulos, Y.: Continuous top-k dominating queries in subspaces, in *Panhellenic Conference on Informatics (PCI 2008)*, pp. 31–35 (2008).

[47] Koudas, N., Ooi, B. C., Tan, K.-L., and Zhang, R.: Approximate NN queries on streams with guaranteed error/performance bounds, in *Proc. Int. Conf. on Very Large Data Bases (VLDB 2004)*, pp. 804–815 (2004).

[48] Kut, A., and Birant, D.: Spatio-temporal outlier detection in large databases, *Journal of Computing and Information Technology*, Vol. 14, No. 4, pp. 291–297 (2006).

[49] Lakshmanan, L. V., Pei, J., and Zhao, Y.: QC-Trees: an efficient summary structure for semantic OLAP, in *Proc. Int. Conf. on Management of Data (ACM SIGMOD 2003)*, pp. 64–75 (2003).

[50] Lane, N. D., Miluzzo, E., Lu, H., Peebles, D., Choudhury, T., and Campbell, A. T.: A survey of mobile phone sensing, *IEEE Communications Magazine*, Vol. 48, No. 9, pp. 140–150 (2010).

[51] Lazaridis, I., and Mehrotra, S.: Progressive approximate aggregate queries with a multi-resolution tree structure, in *Proc. Int. Conf. on Management of Data (ACM SIGMOD 2001)*, Vol. 30, pp. 401–412 (2001).

[52] Ma, Q., He, C., Wu, J., Liu, Z., Zhang, Q., and Sun, Z.: Quantifying spatiotemporal patterns of urban impervious surfaces in China: an improved assessment using nighttime light data, *Journal of Landscape and Urban Planning*, Vol. 130, pp. 36–49 (2014).

[53] Madria, S., Kumar, V., and Dalvi, R.: Sensor cloud: a cloud of virtual sensors, *Journal of IEEE Software*, Vol. 31, No. 2, pp. 70–77 (2014).

[54] Malhotra, B., Nascimento, M. A., and Nikolaidis, I.: Exact top-k queries in wireless sensor networks, *IEEE Trans. on Knowledge and Data Engineering*, Vol. 23, No. 10, pp. 1513–1525 (2011).

[55] Mamoulis, N., Cao, H., Kollios, G., Hadjieleftheriou, M., Tao, Y., and Cheung, D. W.: Mining, indexing, and querying historical spatiotemporal data, in *Proc. Int. Conf. on Knowledge Discovery and Data Mining (ACM SIGKDD 2004)*, pp. 236–245 (2004).

[56] McCallum, A., Nigam, K., and Ungar, L. H.: Efficient clustering of high-dimensional data sets with application to reference matching, in *Proc. Int. Conf. on Knowledge Discovery and Data Mining (ACM SIGKDD 2000)*, pp. 169–178 (2000).

[57] McDonald, D., Sanchez, S., Madria, S., and Ercal, F.: A communication efficient framework for finding outliers in wireless sensor networks, in *Proc. Int. Conf. on Mobile Data Management (MDM 2010)*, pp. 301–302 (2010).

[58] Minack, E., Siberski, W., and Nejdl, W.: Incremental diversification for very large sets: a streaming-based approach, in *Proc. Int. Conf. on Research and Development in Information Retrieval (ACM SIGIR 2011)*, pp. 585–594 (2011).

[59] Mitton, N., Papavassiliou, S., Puliafito, A., and Trivedi, K. S.: Combining cloud and sensors in a smart city environment, *Journal of Wireless Communications and Networking*, No. 1, pp. 247–256 (2012).

[60] Mokbel, M. F., Ghanem, T. M., and Aref, W. G.: Spatio-temporal access methods, *Journal of IEEE Data Eng. Bull.*, Vol. 26, No. 2, pp. 40–49 (2003).

[61] Mokbel, M. F., Xiong, X., and Aref, W. G.: SINA: scalable incremental processing of continuous queries in spatio-temporal databases, in *Proc. Int. Conf. on Management of Data (ACM SIGMOD 2004)*, pp. 623–634 (2004).

[62] Mouratidis, K., Bakiras, S., and Papadias, D.: Continuous monitoring of top-k queries over sliding windows, in *Proc. Int. Conf. on Management of Data (ACM SIGMOD 2006)*, pp. 635–646 (2006).

[63] Mouratidis, K., Papadias, D., and Hadjieleftheriou, M.: Conceptual partitioning: an efficient method for continuous nearest neighbor monitoring, in *Proc. Int. Conf. on Management of Data (ACM SIGMOD 2005)*, pp. 634–645 (2005).

[64] Nikzad, N., Verma, N., Ziftci, C., Bales, E., Quick, N., Zappi, P., Patrick, K., Dasgupta, S., Krueger, I., Rosing, T. Š., et al.: Citisense: improving geospatial environmental assessment of air quality using a wireless personal exposure monitoring system, in *Proc. Int. Conf. on Wireless Health*, pp. 1–8 (2012).

[65] Papadias, D., Kalnis, P., Zhang, J., and Tao, Y.: Efficient OLAP operations in spatial data warehouses, *Journal of Advances in Spatial and Temporal Databases*, pp. 443–459 (2001).

[66] Porkaew, K., Lazaridis, I., and Mehrotra, S.: Querying mobile objects in spatio-temporal databases, *Journal of Advances in Spatial and Temporal Databases*, pp. 59–78 (2001).

[67] Rana, R. K., Chou, C. T., Kanhere, S. S., Bulusu, N., and Hu, W.: Ear-phone: an end-to-end participatory urban noise mapping system, in *Proc. Int. Conf. on Information Processing in Sensor Networks (IPSN 2010)*, pp. 105–116 (2010).

[68] Reddy, S., Estrin, D., Hansen, M. H., and Srivastava, M. B.: Examining micro-payments for participatory sensing data collections, in *Proc. Int. Conf. on Ubiquitous Computing (UbiComp 2010)*, pp. 33–36 (2010).

[69] Reddy, S., Samanta, V., Burke, J., Estrin, D., Hansen, M., and Srivastava, M.: MobiSense - mobile network services for coordinated participatory sensing, in *Proc. Int. Symposium on Autonomous Decentralized Systems (ISADS 2009)*, pp. 231–236 (2009).

[70] Shanmugasundaram, J., Fayyad, U., and Bradley, P. S.: Compressed data cubes for olap aggregate query approximation on continuous dimensions, in *Proc. Int. Conf. on Knowledge Discovery and Data Mining (ACM SIGKDD 1999)*, pp. 223–232 (1999).

[71] Sharifzadeh, M., and Shahabi, C.: Vor-tree: R-trees with voronoi diagrams for efficient processing of spatial nearest neighbor queries, *Proceedings of the VLDB Endowment*, Vol. 3, No. 1-2, pp. 1231–1242 (2010).

[72] Silberstein, A. S., Braynard, R., Ellis, C., Munagala, K., and Yang, J.: A sampling-based approach to optimizing top-k queries in sensor networks, in *Proc. Int. Conf. on Data Engineering (ICDE 2006)*, pp. 68–68 (2006).

[73] Tao, Y., Hristidis, V., Papadias, D., and Papakonstantinou, Y.: Branch-and-bound processing of ranked queries, *Journal of Information Systems*, Vol. 32, No. 3, pp. 424–445 (2007).

[74] Tao, Y., and Papadias, D.: The mv3r-tree: a spatio-temporal access method for timestamp and interval queries, in *Proc. Int. Conf. on Very Large Data Bases (VLDB 2001)*, pp. 431–440 (2001).

[75] Tao, Y., Papadias, D., and Sun, J.: The TPR*-tree: an optimized spatio-temporal access method for predictive queries, in *Proc. Int. Conf. on Very Large Data Bases (VLDB 2003)*, pp. 790–801 (2003).

[76] van Leuken, R. H., Garcia, L., Olivares, X., and van Zwol, R.: Visual diversification of image search results, in *Proc. Int. Conf. on World Wide Web (WWW 2009)*, pp. 341–350 (2009).

[77] Vassiliadis, P., and Sellis, T.: A survey of logical models for OLAP databases, *ACM SIGMOD Record*, Vol. 28, No. 4, pp. 64–69 (1999).

[78] Vee, E., Srivastava, U., Shanmugasundaram, J., Bhat, P., and Yahia, S. A.: Efficient computation of diverse query results, in *Proc. Int. Conf. on Data Engineering (ICDE 2008)*, pp. 228–236 (2008).

[79] Vuran, M. C., Akan, Ö. B., and Akyildiz, I. F.: Spatio-temporal correlation: theory and applications for wireless sensor networks, *Journal of Computer Networks*, Vol. 45, No. 3, pp. 245–259 (2004).

[80] Wang, H., Cai, Y., Yang, Y., Zhang, S., and Mamoulis, N.: Durable queries over historical time series, *IEEE Trans. on Knowledge and Data Engineering*, Vol. 26, No. 3, pp. 595–607 (2014).

[81] Wu, M., Xu, J., Tang, X., and Lee, W.-C.: Monitoring top-k query in wireless sensor networks, in *Proc. Int. Conf. on Data Engineering (ICDE 2006)*, pp. 143–143 (2006).

[82] Wu, M., Xu, J., Tang, X., and Lee, W.-C.: Top-k monitoring in wireless sensor networks, *IEEE Trans. on Knowledge and Data Engineering*, Vol. 19, No. 7, pp. 962–976 (2007).

[83] Xiong, X., Mokbel, M. F., and Aref, W. G.: SEA-CNN: scalable processing of continuous k-nearest neighbor queries in spatio-temporal databases, in *Proc. Int. Conf. on Data Engineering (ICDE 2005)*, pp. 643–654 (2005).

[84] Yang, Y., Pierce, T., and Carbonell, J.: A study of retrospective and on-line event detection, in *Proc. Int. Conf. on Research and Development in Information Retrieval (ACM SIGIR 1998)*, pp. 28–36 (1998).

[85] Yokoyama, M., and Hara, T.: Efficient top-k result diversification for mobile sensor data, in *Proc. Int. Conf. on Distributed Computing Systems (ICDCS 2016)*, pp. 477–486 (2016).

[86] 横山正浩, 原隆浩：モバイルセンサデータベースにおける階層的クラスタリングを用いた Top-k 検索結果の多様化について, データ工学と情報マネジメントに関するフォーラム (DEIM2017) (2017).

[87] 横山正浩, 原隆浩：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, 情報処理学会論文誌, Vol. 58, No. 2, pp. 481–494 (2017).

[88] 横山正浩, 原隆浩, Madria, S. K.：モバイルセンサデータストリーム環境における継続的な Top-k 検索結果の多様化について, 電子情報通信学会技術研究報告, Vol. 117, No. 212, pp. 7–12 (2017).

[89] Yokoyama, M., Hara, T., and Madria, S. K.: Efficient diversified set monitoring for mobile sensor stream environments, in *Proc. Int. Conf. on Big Data (Big Data 2017)*, pp. 500–507 (2017).

[90] 横山正浩, 原隆浩, 西尾章治郎：モバイルセンサデータベースにおける効率的な Top-k 検索結果の多様化について, Web とデータベースに関するフォーラム (WebDB Forum 2015) 論文集, Vol. 2015, pp. 118–125 (2015).

[91] Yu, A., Agarwal, P. K., and Yang, J.: Top-k preferences in high dimensions, *IEEE Trans. on Knowledge and Data Engineering*, Vol. 28, No. 2, pp. 311–325 (2016).

[92] Yu, C., Lakshmanan, L., and Amer-Yahia, S.: It takes variety to make a world: diversification in recommender systems, in *Proc. Int. Conf. on Extending Database Technology (EDBT 2009)*, pp. 368–378 (2009).

[93] Yu, X., Pu, K. Q., and Koudas, N.: Monitoring k-nearest neighbor queries over moving objects, in *Proc. Int. Conf. on Data Engineering (ICDE 2005)*, pp. 631–642 (2005).

[94] Zhang, J., Wei, Q., and Chen, G.: A heuristic approach for λ-representative information retrieval from large-scale data, *Journal of Information Sciences*, Vol. 277, pp. 825–841 (2014).

[95] Zhong, S.: Efficient streaming text clustering, *Journal of Neural Networks*, Vol. 18, No. 5, pp. 790–798 (2005).