The University of Osaka
Institutional Knowledge Archive

Title	アドホックネットワークにおけるTop-k検索のためのルーティング技術に関する研究
Author(s)	天方, 大地
Citation	大阪大学, 2015, 博士論文
Version Type	VoR
URL	https://doi.org/10.18910/52022
rights	
Note	

The University of Osaka Institutional Knowledge Archive : OUKA

https://ir.library.osaka-u.ac.jp/

The University of Osaka

アドホックネットワークにおける
Top-k検索のためのルーティング技術に関する研究

提出先　　大阪大学大学院情報科学研究科

提出年月　2015年1月

天方　大地

関連発表論文

1. 学会論文誌発表論文

1. Amagata, D., Sasaki, Y., Hara, T., and Nishio, S., “Efficient Multi-dimensional Top-k Query Processing in Wireless Multi-hop Networks,” Mobile Information Systems (採録済).

2. 天方 大地, 佐々木 勇和, 原 隆浩, 西尾 章治郎, “モバイルアドホックネットワークにおけるクラスタを用いた Top-k クエリルーティング手法,” 情報処理学会論文誌, Vol. 55, No. 5, pp. 1555–1567 (2014 年 5 月).

3. 天方 大地, 佐々木 勇和, 原 隆浩, 西尾 章治郎, “アドホックネットワークにおける Top-k 検索のためのルーティング手法,” 情報処理学会論文誌, Vol. 54, No. 8, pp. 2036–2047 (2013 年 8 月).

2. 国際会議発表論文

1. Amagata, D., Sasaki, Y., Hara, T., and Nishio, S., “CTR: An Efficient Cluster-based Top-k Query Routing in MANETs,” Proc. ACM Int. Conf. on Advanced in Mobile Computing & Multimedia, pp. 225–234 (December 2014).

2. Amagata, D., Sasaki, Y., Hara, T., and Nishio, S., “A Robust Routing Method for Top-k Queries in Mobile Ad Hoc Networks,” Proc. IEEE Int. Conf. on Mobile Data Management (MDM), pp. 251–256 (June 2013).

3. Amagata, D., Sasaki, Y., Hara, T., and Nishio, S., “A Routing Method for Top-k Query Processing in Mobile Ad Hoc Networks,” Proc. IEEE Int. Conf. on Advanced Information Networking and Applications (AINA) , pp. 161–168 (Mar. 2013).

3. 国内会議発表論文（査読付）

1. 天方 大地, 佐々木 勇和, 原 隆浩, 西尾 章治郎, “分散データベースにおける効率的な確率的最近傍データ検索手法,” 第 7 回 Web とデータベースに関するフォーラム (WebDB Forum 2014), A7-4 (2014 年 11 月).

2. 天方 大地, 佐々木 勇和, 原 隆浩, 西尾 章治郎, “アドホックネットワークにおける多次元データ Top-k 検索手法,” 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) 論文集, pp. 212–219 (2013 年 12 月).

3. 天方 大地, 佐々木 勇和, 原 隆浩, 西尾 章治郎, “モバイルアドホックネットワークにおけるクラスタを用いた Top-k 検索のためのルーティング手法,” 情報処理学会マルチメディア，分散，協調とモバイル (DICOMO) シンポジウム論文集, pp. 561–568 (2013 年 7 月).

4. 天方 大地, 佐々木 勇和, 原 隆浩, 西尾 章治郎, “アドホックネットワークにおける Top-k 検索のためのルーティング手法の拡張,” 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), pp. 9–16 (2012 年 10 月).

5. 天方 大地, 佐々木 勇和, 原 隆浩, 西尾 章治郎, “アドホックネットワーク上の効率的な Top-k 検索のためのルーティング手法,” 情報処理学会マルチメディア，分散，協調とモバイル (DICOMO) シンポジウム論文集, pp. 1236–1244 (2012 年 7 月).

以上

内容梗概

近年，無線通信端末のみで自律分散的にネットワークを構築するアドホックネットワークへの関心が高まっており，緊急災害時などのインフラが利用できない環境での利用が期待されている．また，ビッグデータ時代と称されるように，情報量の増加と多様化に伴い，大量のデータの中からユーザが必要とするデータのみを検索する技術が求められている．ユーザが指定した検索条件からデータをスコア付けし，その上位 k 個のデータを検索する Top-k 検索は，その一つの手段であり，集中管理データベースや，Peer-to-Peer(P2P) ネットワークおよび無線センサネットワークなどの分散環境において，これまでに多くの研究が行われている．アドホックネットワークにおいても Top-k 検索は実用的な検索手段であり，災害地において優先的な救援を必要とする重症患者の検索支援など，様々な状況で効果的に利用できる．そのため，筆者の研究グループでは，これまでにアドホックネットワークにおける Top-k 検索手法を提案した．しかし，既存手法では，ネットワーク内の全ての端末にメッセージを送信して上位 k 個のデータを検索するため，通信量および遅延が大きく，特に大規模なネットワークにおいて効率的ではない．

そこで，本研究では，アドホックネットワークにおいて，効率的な Top-k 検索を実現するためのルーティング手法を提案する．無線で通信を行うアドホックネットワークでは，パケットロスや端末の電力消費を抑制する必要があるため，通信量の削減が重要である．ネットワーク内の全ての端末にメッセージを送信する手法は，このような観点から効率的ではなく，特にパケットロスにより必要なデータを取得できなくなるという問題が生じる．この問題を解決するためには，上位 k 個のデータを取得するために必要な端末のみで検索を行なうアプローチが有効であり，これを可能とする Top-k 検索のためのルーティング手法を実現する．検索に必要な端末にのみメッセージを送信することにより，通信量の増加を抑制し，パケットロスを削減できるため，検索精度を維持することができる．また，ネットワークを構成する端末が自由に移動するモバイルアドホックネットワークでは，ネットワークトポロジが動的に変化するため，メッセージの送信経路を静的に管理すると検索精度が低下してしまう．そのため，小さい通信量でメッセージの送信経路

を動的に管理することが重要な課題である．また，Top-k 検索の検索条件（スコアリング関数）は，統一されている場合（またはデータに対してスコアが一意の場合），およびユーザが個々に指定できる場合が考えられる．本研究では，これら二つに対する効率的な Top-k 検索のためのルーティングについて考えるが，後者の場合，条件によってデータの順位が変わるため，どのような条件においても必要な端末にのみメッセージを送信することを可能とするアプローチが必要である．

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 章では，序論として研究背景と目的について述べる．

第 2 章では，モバイルアドホックネットワークを想定し，高順位のデータを取得するためにクエリメッセージを転送する必要がある隣接端末を経路表により管理し，この経路表を利用して Top-k 検索を行うルーティング手法を提案する．各端末は，検索を要求された順位のデータに対して，その検索先となる隣接端末にのみクエリメッセージを送信することにより，無駄なメッセージの送信を削減する．端末の移動により，経路表で管理している隣接端末との無線リンクが切断された場合には，その端末への経路探索を行なうことにより経路表を修正し，正確な Top-k 検索を継続する．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，ネットワークトポロジ変化の激しい環境や大規模なネットワークに対応することを目指し，クラスタを用いた Top-k 検索のためのルーティング手法を提案する．この手法では，1 ホップ先（隣接端末）までに存在する端末でクラスタを構成する，1 ホップクラスタリングを行い，このとき，高順位のデータを持つ端末をクラスタヘッドとする．クラスタを用いたメッセージのルーティングでは，クラスタヘッドに向かってメッセージが送信される．そのため，提案手法では，上位のデータを持つ端末（クラスタヘッド）にクエリメッセージが送信され，上位 k 個のデータの取得に必要がない端末への無駄な通信を削減できる．また，各端末はクラスタヘッドのみとの無線リンクを確認すればよいため，トポロジ変化に対応しつつ，無駄なメッセージの送信を削減できる．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，静的なアドホックネットワークにおける，多次元データに対する

Top-k 検索のためのルーティング手法を提案する．この Top-k 検索では，ユーザが個々に検索条件を指定するため，データのスコアは検索条件ごとに異なる．提案手法では，データの特性（属性値）を考慮し，まず高順位となりやすいデータを持つ端末をクラスタヘッドにする．これにより，第 3 章の提案手法と同様に，上位となるデータを持つ端末に向けてクエリメッセージを送信できる．また，各端末はクラスタリングの際に，隣接端末が保持する上位 k 個に含まれる可能性があるデータを把握し，上位 k 個に含まれるデータを保持していない端末へのクエリメッセージの送信を削減する．さらに，クエリメッセージに，クエリ送信経路上における上位 k 個のデータの情報を添付することにより，隣接端末以外の情報も取得し，クエリメッセージの無駄な転送をさらに削減できる．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，今後の研究課題について述べ，本論文のまとめとする．

目 次

第 1 章　序章 **1**
1.1　研究背景 1
1.2　研究動機 2
1.2.1　通信量の削減 3
1.2.2　モバイル環境への対応 4
1.2.3　多様な検索条件への対応 5
1.3　研究内容 7
1.4　本論文の構成 11

第 2 章　経路表を用いた Top-k 検索のためのルーティング手法 **13**
2.1　まえがき 13
2.2　想定環境 14
2.3　関連研究 15
2.3.1　モバイルアドホックネットワークにおけるルーティング手法 15
2.3.2　P2P ネットワークにおける Top-k 検索手法 16
2.3.3　無線センサネットワークにおける Top-k 検索手法 17
2.3.4　モバイルアドホックネットワークにおける Top-k 検索手法 . 18
2.4　提案手法 19
2.4.1　経路表の作成 19
2.4.2　Top-k 検索 20
2.4.3　経路表の更新 22
2.4.4　提案手法の問題点 24
2.5　拡張手法 24
2.5.1　Top-k 検索 25

2.5.2 経路表の更新 28
2.6 性能評価 30
2.7 むすび 36

第 3 章 クラスタを用いた Top-k 検索のためのルーティング手法 39
3.1 まえがき 39
3.2 関連研究 40
3.3 提案手法 41
3.3.1 クラスタの構築 41
3.3.2 Top-k 検索のためのメッセージ処理 42
3.3.3 スコア更新への対応 50
3.4 性能評価 51
3.5 むすび 57

第 4 章 多次元データに対する Top-k 検索のためのルーティング手法 59
4.1 まえがき 59
4.2 想定環境と問題定義 60
4.2.1 想定環境 60
4.2.2 問題定義 61
4.3 関連研究 63
4.3.1 集中管理データベースにおける多次元データに対する Top-k 検索 63
4.3.2 分散ネットワークにおける多次元 Top-k 検索 64
4.4 提案手法 66
4.4.1 クラスタの構築 66
4.4.2 フィルタの作成 69
4.4.3 Top-k 検索 72
4.5 性能評価 76
4.5.1 初期処理コストおよびデータ更新処理コスト 79
4.5.2 Top-k 検索 80

4.6 むすび 87

第 5 章 結論 **89**

5.1 本論文のまとめ 89

5.2 検討課題 91

謝辞 **95**

第1章　序章

1.1　研究背景

近年，データの多種多様化に伴い，大量のデータの中から，ユーザの趣向に応じてデータの順位を決定し，その上位 k 個のデータを検索する Top-k 検索への関心が高まっている [13, 31, 39, 45]．また，一つの基地局（サーバ）が全てのデータを管理するのではなく，複数のサーバにより分散的にデータを管理する，分散ネットワークが普及している [12, 21, 78]．アドホックネットワークは分散ネットワークの一つであり，特別な通信基盤を必要とせず，端末間のみで通信を行うことができる．さらに，IEEE802.11[38] や Bluetooth[73]，Zigbee[90]，Wifi Direct[81] などをはじめとする無線通信技術の発展と，スマートフォンやタブレット端末などの計算機の小型化や高性能化に伴い，ユーザの持つ無線通信機器を搭載した端末により，無線アドホックネットワークを構築できる環境が整いつつある．アドホックネットワークでは，既存のインフラを必要としないという利便性から，緊急災害時 [61] や車々間通信 [67]，イベント会場での広告 [46] などへの応用が期待されている．特に緊急災害時では，既存の通信基盤が破損し，使用できない状況が想定されるため，レスキュー隊員が持つ端末でアドホックネットワークを構築することが考えられる．また，このような環境では，救援のための資源（救急車などの医療リソースや人員など）が限られている場合が想定される．このとき，重症患者や破損度の高い建物などの優先度の高いデータを Top-k 検索を用いて取得することにより，円滑な救援活動を支援できる．例えば，図 1.1 では，災害時において，レスキュー隊員が持つ端末でアドホックネットワークを構築している様子を示している．各レスキュー隊員は，自身が持つ端末に被災者の情報を入力する．医療資源に限りがあり（例えば 2 台の救急車），重症患者から対応する場合には，Top-k

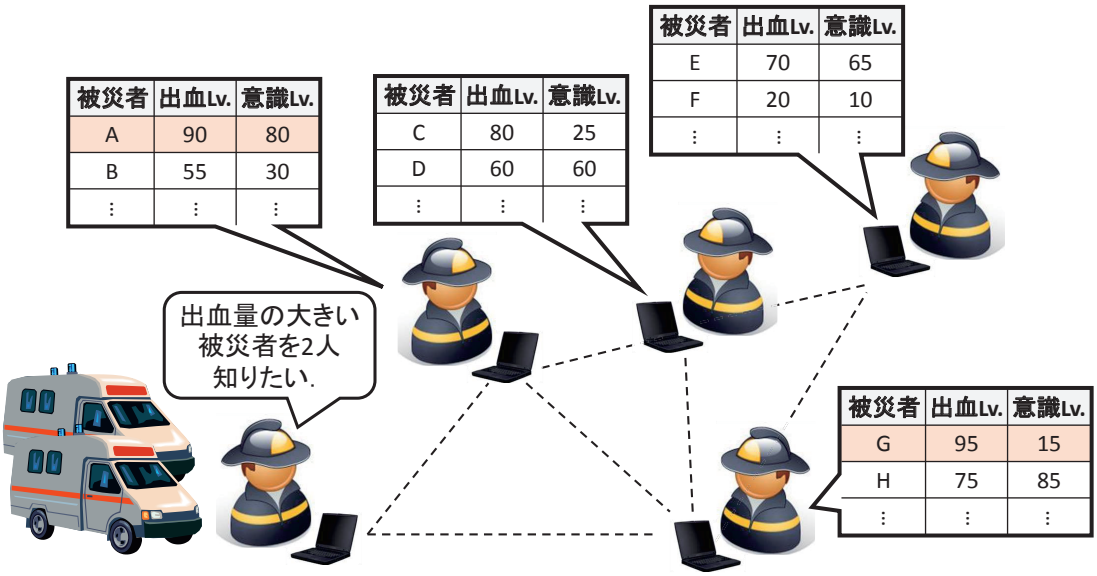


図 1.1: アドホックネットワークにおける Top-k 検索例

検索（$k = 2$）を行うことにより，出血量の大きい被災者のデータを効率的に取得できるため，迅速な対応が可能である.（この例では，出血レベルによりデータをランキングしており，A および G が上位 2 個のデータとなる.）

1.2　研究動機

アドホックネットワークでは，通信帯域に限りがあるため，効率的にデータを取得するためには通信量を削減する必要がある．さらに，アドホックネットワークで Top-k 検索を行なう際，対応すべき点が二つある．一つは，端末の移動によるネットワークトポロジ変化に対応することである．もう一方は，多様な検索条件に対応することである．Top-k 検索の際にユーザが指定する検索条件は，重要視するデータを決定するものであるため，ユーザごとに異なる検索条件が指定されることが考えられる．以下では，これらについて詳しく議論する．

1.2.1 通信量の削減

アドホックネットワークでは，各端末は無線通信により，通信範囲内に存在する端末（例えば自身から半径 100[m] 以内に存在する端末）と直接通信することが可能である．しかし，無線通信では，通信量が増加すると帯域が圧迫され，パケットロスが発生してしまう．アドホックネットワークで Top-k 検索を実行している間にパケットロスが発生すると，必要なデータの取得に失敗してしまう可能性が高い．この問題を解決するためは，無駄なメッセージおよびデータの送信を可能な限り削減する必要がある．例えば図 1.1 におけるクエリ発行端末は，出血量が最も大きい 2 人のデータを要求している．このとき，A および G 以外のデータの送信は無駄に通信量が増加する原因となる．

ここで，これまでに提案されている Top-k 検索手法について，アドホックネットワークへの応用を考える．まず，全てのデータを 1 台の端末が管理し，その端末にアクセスすることでデータを取得する集中管理型の方法 [22, 28, 31] が挙げられる．しかし，データの更新が起きた際にデータを転送するオーバヘッドや，1 台の端末のみに通信が集中することにより，頻繁にパケットロスが発生するといった観点から，このアプローチはアドホックネットワークでは有効ではない．分散環境において効率的に Top-k 検索を行う手法が，Peer-to-Peer(P2P) ネットワーク [1, 20] や無線センサネットワーク [49, 82] の分野において多数提案されている．しかし，P2P ネットワークでは，ネットワーク内の任意の端末と直接通信して上位 k 個のデータを取得することを想定している [78, 79]．そのため，通信範囲に制限があるアドホックネットワークには適用可能ではない．一方，無線センサネットワークは，無線通信や端末のバッテリの制限など，アドホックネットワークと類似した特徴をもつ．そこで，消費電力や帯域の圧迫といった問題を解決するために，Top-k 検索を行う際，上位 k 個のデータの取得に必要な端末のみでメッセージ処理を行う手法が提案されている．しかし，無線センサネットワークでは，全ての端末が特定の位置に固定されており，さらに 1 台のシンクのみが Top-k クエリを発行端末することを想定しているため，単純にアドホックネットワークに適用することは困難である．

アドホックネットワークにおいても，検索精度の維持と通信量の削減を目的と

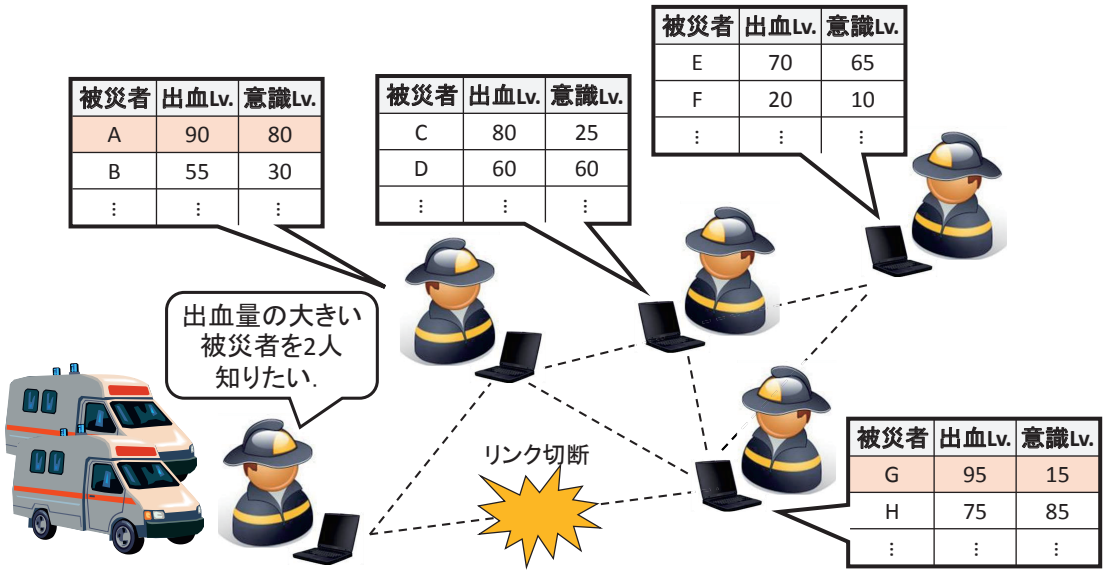


図 1.2: 無線リンク切断の例

した Top-k 検索手法が提案されている [33, 63, 64]．しかし，これらの手法では，フラッディング [77] を用いてクエリメッセージの転送を行っており，ネットワーク内の全ての端末にクエリメッセージが転送される．そのため，上位 k 個のデータの取得に必要がない多くの端末が，検索クエリの転送，およびデータの返信を行い，無駄に通信量が大きくなってしまう．この問題は，端末数が増加するとともに深刻化し，ブロードキャストストーム [75] と呼ばれ，通信量の増加やパケット衝突によるパケットロスが頻繁に発生し，検索結果の精度が低下する原因となる．そのため，クエリメッセージの無駄な転送を削減し，上位 k 個のデータを取得するために必要な端末にのみクエリメッセージを送信するクエリルーティング手法を設計し，検索精度を維持しつつ，通信量を削減することが求められる．

1.2.2　モバイル環境への対応

アドホックネットワークでは，ネットワークを構成する端末を持つユーザが自由に移動し，ネットワークトポロジが動的に変化する環境が考えられる．このようなモバイルアドホックネットワークでは，端末の通信範囲内に存在する端末（隣接端末）が動的に変化するため，静的に隣接端末の情報を管理すると，正確に Top-k

検索を行えない．例えば，クエリメッセージやデータを送信する際，特定の端末にのみ送信するように設計すると，無線リンクの切断が起きた場合，メッセージが送信できず，検索クエリの発行端末は上位 k 個のデータを取得できない．図 1.2 は，その具体例として，検索クエリの発行端末とデータ G を持つ端末との無線リンクが切断した状況を示している．このとき，データ G を持つ端末は，クエリ発行端末に直接メッセージを送信できないため，他の端末を経由してデータ G を送信する必要がある．このように，ネットワークトポロジ変化の影響を避けるためには，高速に Top-k 検索を行いつつ，ネットワークトポロジ変化が起きた場合にも正確な検索結果を取得可能なルーティング手法を設計する必要がある．

モバイルアドホックネットワークにおける Top-k 検索手法は，文献 [33, 63, 64] 以外にも提案されており，文献 [66] では，文献 [34] で提案された方法でデータの複製が配置されている環境における Top-k 検索手法を提案している．文献 [65] では，ネットワークトポロジ変化の影響を受けない Top-k 検索のためのデータ複製配置手法が提案されている．これらの手法では，他の端末が持つデータを複製し，検索範囲の拡大を抑制することにより，通信量の削減を実現している．しかし，これらの手法では，クエリメッセージの転送にはフラッディングを用いており，端末数が多い場合に通信量が増加する．さらに，データのスコアまたは順位を用いて複製を配置するため，データの更新が起きるとデータの配置が非効率となる点，および複製すべきデータの転送によって通信量が増加する点が問題となる．これらのオーバヘッドがボトルネックとなるため，本研究ではデータの複製は用いず，データの更新が起きた場合においても，小さい通信量でメンテナンスを行う．

1.2.3 多様な検索条件への対応

Top-k 検索は，ユーザが指定した検索条件をもとにデータにスコアを付与し，その上位 k 個のデータを検索するものである．ここで，アドホックネットワーク内の端末に格納されているデータは，複数の属性値を保持していることが考えられる．例えば，図 1.1 のように，災害地における被災者のデータは，出血レベルや意識レベルなどの属性値を持つことが考えられる．ユーザが Top-k 検索を行なう際

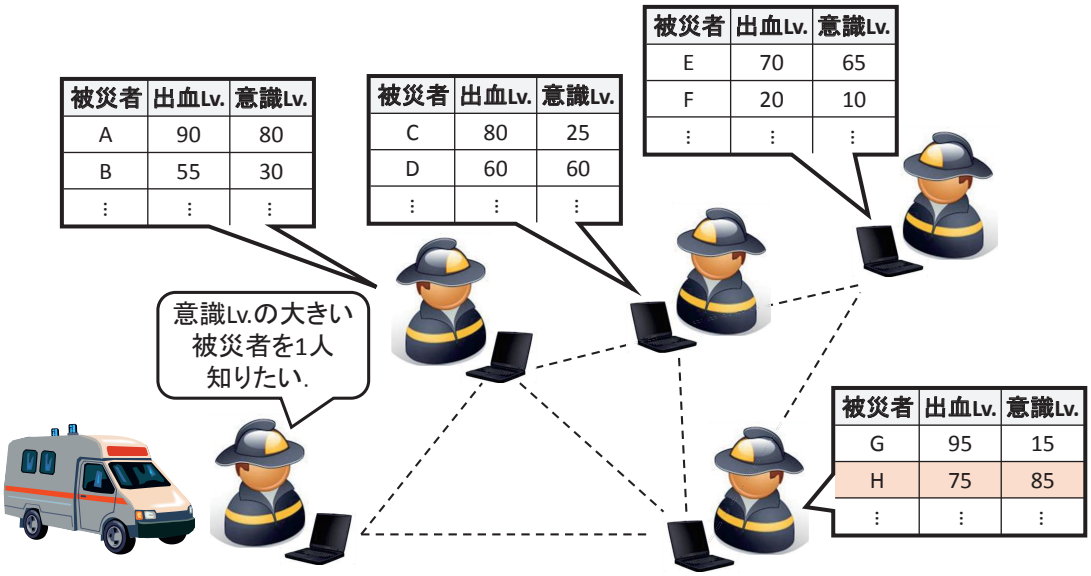


図 1.3: 異なる検索条件が指定された場合の Top-k 検索の例

に指定する検索条件は，どの属性を，どの程度重視するかを反映させるものであるため，ユーザごとに検索条件が異なる場合が予想される．例えば，医療チームの専門や対応可能な医療資源から，優先すべき事項（検索条件）を決定し，救援活動を行うものが該当する．図 1.1 では，出血レベルのみを重視し，データをランキングしており，1 位のデータは G となる．一方，図 1.3 の例では，意識レベルのみからランキングしているため，1 位のデータは H となる．

アドホックネットワークにおける Top-k 検索の既存手法 [63, 65, 66] は，ネットワーク内で指定される検索条件が一つであると想定しているため，多様に検索条件が指定される環境では効率的でない．また，文献 [33, 64] における提案手法では，任意の検索条件に対応できるが，前述の通り，フラッディングを用いてクエリメッセージを転送している問題がある．アドホックネットワークにおける Top-k 検索では，通信量を可能な限り削減する必要があるため，多様に検索条件が指定される場合においても，検索に必要な端末のみで Top-k 検索を行うことが望ましい．このとき，検索に必要な端末もそれぞれ異なるため，クエリメッセージを送信する隣接端末を，条件に応じて動的に決定することが重要である．

1.3 研究内容

本研究では，アドホックネットワークにおける Top-k 検索問題について取り組む．検索精度を維持しつつ，通信量を削減するためには，上位 k 個のデータの取得に必要な端末のみで検索を行うアプローチが有効であり，これを実現する Top-k 検索のためのルーティング手法の提案を目的する．また，Top-k 検索のためのルーティングについて考えるとき，検索条件やネットワークトポロジの変化に影響せず，効率的に上位 k 個のデータを検索できるものが望ましい．この手法を実現するためのステップとして，検索条件およびネットワーク環境にそれぞれ制約を置き，必要な端末のみで Top-k 検索を実行できる手法について考え，クエリルーティングの有効性について検証する．つまり，本論文では，以下の課題について考える．

- モバイルアドホックネットワークにおいて，統一された検索条件の下で Top-k 検索を行う環境に対する効率的なクエリルーティング.
- 静的なアドホックネットワークにおいて，ユーザがそれぞれ検索条件を指定し，Top-k 検索を行う環境に対する効率的なクエリルーティング.

前者の，検索条件が統一されている場合は，指定される検索条件が一つである場合に該当する．ユーザがそれぞれ独立して検索条件を指定する場合，多様な検索条件に対応する必要があるが，例えば，ネットワークを構成する医療チームが予め検索条件を決定しておくことも考えられるため，この制約は妥当であると考える．また，後者の，静的なアドホックネットワーク環境の具体的なものについて考える．救援活動などのアプリケーションでは，レスキュー隊員の担当領域が決まっていることが予想される．このとき，レスキュー隊員の移動は局所的であり，ネットワークはトポロジの変化が起きず，静的なものと想定できる．

各章では，上のそれぞれの想定の下において，以下の研究課題に取り組む.

- 経路表を用いた Top-k 検索のためルーティング手法（第 2 章）

  統一された検索条件の下で Top-k 検索を行う場合，各端末が上位のデータを取得するために必要な情報を経路表で管理することにより，無駄なメッセージの転送を削減できる．例えば，各データを取得するためにクエリメッセー

ジを転送する必要がある隣接端末を管理することで，検索を要求された順位に対する隣接端末を特定でき，その端末にのみクエリメッセージを送信すれば，上位 k 個のデータを持つ端末にアクセスできる．これを実現するために，経路表を用いた Top-k 検索のためのルーティング手法を提案する．提案手法では，まず各端末が，ネットワーク内で初めて Top-k 検索が行われる際に経路表を作成し，経路表で管理された情報に従って Top-k 検索を行う．Top-k 検索の際には，クエリメッセージに検索要求順位を添付し，このメッセージを受信した端末は，要求された順位のデータのみを検索する．これにより，クエリメッセージの無駄な転送やデータの送信を削減できる．しかし，この手法では，各データを検索するためのメッセージは，それぞれ単一の経路で転送されるため，端末の移動によりネットワークトポロジの変化が起きた場合のメッセージ転送の信頼性が低い．そこで，提案手法を拡張し，クエリメッセージを受信した端末がそれぞれ k 位までのデータを検索し，メッセージの転送を複数経路により行うことで，ネットワークトポロジ変化の影響を小さくする．拡張手法は，提案手法よりもメッセージの転送率を高くできるため，検索精度を高く維持することが可能である．

- クラスタを用いた Top-k 検索のためのルーティング手法（第 3 章）

  モバイルアドホックネットワークでは，ネットワークトポロジが頻繁に変化する環境や，多くの端末により構成される大規模なネットワーク環境も数多く存在する．そのため，Top-k 検索手法においても，このような環境に対応することが重要である．特に，端末数が多い場合，全ての端末がクエリメッセージを送信してしまうと，通信量が大きくなり，帯域が圧迫されるため，検索精度が低下してしまう．そのため，検索に必要な端末のみで Top-k 検索を行うアプローチは効果的である．ここで，端末数が多い場合には，隣接端末の数も多くなるため，必要なデータを取得するためのクエリメッセージの転送先となる端末を，適切に選択することが重要である．さらに，ネットワークトポロジが頻繁に変化する環境にも対応するため，小さい通信量で隣接端末の情報を更新する必要がある．これらの問題を同時に解決し，小さい通信量で高い検索精度を維持する Top-k 検索を実現する．

そこで，クラスタを用いたTop-k検索のためのルーティング手法を提案する．この手法では，ネットワーク内で初めてTop-k検索が行われる際に1ホップクラスタを構築し，クラスタを管理する端末であるクラスタヘッド間でクエリメッセージのルーティングを行うことにより，上位k個のデータを取得する．このクラスタリングでは，上位のデータを保持する端末をクラスタヘッドとしているため，高順位のデータを持つ端末間のみによる検索を実現でき，無駄な通信を削減できる．さらに，各端末はクラスタヘッドのみとの無線リンクを管理すればよいため，ネットワークトポロジの変化や隣接端末が多い場合にも容易に対応できる．クラスタヘッドは定期的にメッセージを送信することにより，各端末は無線接続しているクラスタヘッドを確認できるため，メンテナンスにかかる通信量も小さい．

- 多次元データに対するTop-k検索のためのルーティング手法（第4章）

  ネットワーク内の各端末が，それぞれ検索条件を指定してTop-k検索を行う場合，上位k個のデータも検索条件によって異なる．特に，データが複数の属性をもっている多次元データの場合，各属性（次元）をどの程度重視するかなどがユーザによって異なるため，様々な検索条件が指定される．例えば，救援活動を行っている場合，状況やレスキュー隊員（グループ）によって救援の優先度を決定するための事項（出血レベルや意識レベルなど）が異なる場合や，物理的にアクセスできる範囲を考慮し，検索範囲を制限することが考えられる．救援活動に代表される多くのアプリケーションでは，短い検索時間を要求するため，データの取得に必要な端末のみで迅速にTop-k検索を行うことが望ましい．ここで，上位k個のデータの取得に必要となる端末も検索条件によって異なるため，クエリメッセージを転送する隣接端末を，与えられた検索条件によって動的に決定する必要がある．また，指定される検索条件は事前に知り得ないため，あるデータが取り得る順位を予め把握しておくことは不可能である．つまり，上位k個に含まれるデータも事前に把握できないため，Top-k検索中に，k位以内となり得るデータ，およびk位以内になり得ないデータの判断を行う必要がある．この判断を効率よく行い，無駄なデータの送信を削減することが重要である．

本研究では，ネットワークトポロジが変化しないアドホックネットワークを想定し，いかなる検索条件が指定された場合においても，検索に必要がない端末へのクエリメッセージの転送を削減し，正確に Top-k 検索を行うルーティング手法を提案する．提案手法では，いかなる検索条件においても高順位となりやすいデータを持つ端末をクラスタヘッドとするクラスタリングを行う．これにより，第 3 章の手法と同様に，高順位のデータを持つ端末間での Top-k 検索を実現する．また，クラスタリングの際に，隣接端末が保持する上位 k 個に含まれる可能性があるデータの情報を収集する．ある検索条件が与えられたとき，隣接端末が保持するデータのスコアを計算し，それらが上位 k 個に含まれ得るものか確認する．このとき，上位 k 個に含まれる可能性がないデータのみを保持している端末である場合，クエリメッセージの無駄な転送を抑制することができ，通信量を削減するとともに，高速な Top-k 検索を実現できる．さらに，クエリメッセージに，転送経路中に存在するデータのうち，上位のもののスコアを添付することにより，隣接端末以外の端末が持つデータ集合における上位のデータのスコアを把握することができる．これにより，上位 k 個のデータを取得するためにクエリメッセージを送信する必要がない隣接端末をさらに特定でき，クエリメッセージの不要な転送をさらに削減し，無駄なデータの返信を防ぐことが可能である．

アドホックネットワークにおける Top-k 検索のためのルーティング手法を実現するという問題に対して，各章では，その部分問題を解決する．具体的には，Top-k 検索の検索条件，およびネットワーク環境の観点から，それぞれ異なる想定環境において手法を提案する．第 2 章の提案手法は，統一された検索条件を想定し，ネットワークトポロジの変化が起きた場合にもある程度対応できる．第 3 章においても，統一された検索条件を想定しているが，ネットワークトポロジの変化が激しく起こる環境への適応という問題に取り組んでいる点で，第 2 章と異なる．第 4 章では，ネットワークトポロジの変化がない環境において，Top-k 検索を開始するユーザがそれぞれ検索条件を指定する状況を想定している．

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，各端末は統一された検索条件に従って Top-k 検索を行う環境を想定し，経路表を用いてクエリメッセージを転送するルーティング手法を提案する．経路表はネットワーク内で初めて Top-k 検索が行われる際に作成され，各順位のデータに対するクエリメッセージの送信先となる端末の識別子（宛先）が管理されている．提案手法では，検索クエリの発行端末が自身の経路表を参照し，検索要求順位を添付したクエリメッセージを，k 位までの各順位に対応する宛先端末にそれぞれ送信する．また，拡張手法では，Top-k 検索の際，各端末が基本的に，経路表の k 位までの宛先端末にクエリメッセージを送信する．このとき，クエリメッセージの受信および送信状況から，転送が不要な宛先端末を判断し，その端末を送信先から除外する．提案手法，および拡張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，ネットワークトポロジの頻繁な変化，および大規模なネットワークに対応するため，クラスタを用いた Top-k 検索のためのルーティング手法を提案する．この手法では，ネットワーク内で初めて Top-k 検索が行われる際に 1 ホップクラスタを構築する．このとき，高順位のデータを持つ端末をクラスタヘッドにすることにより，クラスタヘッド間で Top-k 検索を実行できる環境を整える．クエリメッセージは，クラスタヘッド間で転送され，複数のクラスタに属している端末がクエリメッセージを中継する．また，データまでのホップ数を利用し，クエリメッセージの送信元端末に比べて自身のほうが近いデータのみを検索することにより，上位 k 個のデータの取得に必要なクラスタヘッドにのみクエリメッセージを送信する．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，静的なアドホックネットワーク環境において，多次元データに対する Top-k 検索のためのルーティング手法を提案する．提案手法は，隣接端末が保持するデータの情報をクラスタリングの際に取得し，クラスタを利用してクエリメッセージを送信する．提案手法では，クエリメッセージに，このメッセージの転送経路中に存在するデータの中で上位のもののスコアを添付することで，各端

末に自律的に不要なデータを把握させる．この効果をさらに高めるために，提案手法におけるクラスタリングでは，いかなる検索条件においても高順位となりやすいデータを持つ端末をクラスタヘッドとする．クエリメッセージはクラスタヘッドに向かって送信されるため，クエリメッセージには，より高順位のデータの情報が添付されるようになり，無駄なデータの返信を削減しやすく，かつ，検索に必要がない端末へのクエリメッセージの送信も削減できる．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，今後の研究課題について述べ，本論文のまとめとする．

第 2 章は文献 [2, 3, 4, 5, 6] で公表した結果に，第 3 章は文献 [8, 9, 10] で公表した結果に，第 4 章は文献 [7, 11] で公表した結果に基づき論述する．

第2章　経路表を用いたTop-k検索のためのルーティング手法

2.1　まえがき

アドホックネットワークにおいてTop-k検索を実現する単純なアプローチとして，メッセージを受信した端末がそのメッセージをブロードキャストする処理を繰り返すフラッディング[77]という方法により，クエリメッセージをネットワーク内の全ての端末に送信し，各端末は自身が把握するデータのうち上位k個のものを返信する方法が考えられる．この方法は，理論的に正確に上位k個のデータを取得できるものではあるが，最終結果に含まれないデータも多く送信される．そのため，アドホックネットワークでは，通信量が増加することによるパケットロスが頻繁に生じてしまい，必要なデータが取得できず，検索精度が低下してしまう問題がある．この問題を解決するために，上位k個に含まれないデータの返信を削減する手法[33, 63, 64]が提案されているが，クエリメッセージをフラッディングにより転送しているため，上位k個のデータの取得に必要がない多くの端末もクエリメッセージの送信やデータの返信を行っており，通信量を効果的に削減できていない．

そこで本章では，モバイルアドホックネットワークにおいて，上位k個のデータの取得に必要な端末のみで検索を行うことを目的とするルーティング手法を提案する．提案手法では，まずTop-k検索のための経路表を作成する．この経路表では，各順位のデータに対して，クエリメッセージの送信先となる隣接端末（宛先）を管理しており，Top-k検索の際には，この経路表に基づいてクエリメッセージを転送する．検索クエリの発行端末は，検索を要求するデータの順位を添付したクエリメッセージを，k位までの各宛先にユニキャストにより送信する．クエリメッ

セージを受信した端末は，自身の経路表を参照し，検索を要求された順位に対応する宛先にクエリメッセージを転送する．要求された全ての順位のデータを保持している端末は，要求されたデータのみを返信する．これにより，上位 k 個となるデータの取得に不要な端末へのクエリメッセージの転送，および上位 k 個に含まれないデータの返信を抑制できる．ネットワークトポロジの変化に対応するため，無線リンク切断を検知した場合，切断先の端末までの経路探索を行うことにより経路表を修正し，正確な Top-k 検索を継続する．さらに，本章では，この提案手法を拡張し，複数経路でクエリメッセージを転送する手法も提案する．拡張手法は，提案手法よりもネットワークトポロジの変化に効果的に対応することを目的としており，メッセージの転送率を向上し，検索精度を高く維持する．

以下では，2.2 節で本章の想定環境について説明する．2.3 節で関連研究を紹介し，2.4 節で提案手法について述べる．さらに，2.5 節で提案手法を拡張し，2.6 節でシミュレーション実験の結果を示す．最後に 2.7 節で本章のまとめを行う．

2.2 想定環境

ネットワークモデル． ネットワーク内には，同等の性能をもつ n 台の端末（$N_1, N_2, ..., N_n$）が存在し，各々がデータ集合 O_i（$1 \leq i \leq n$）を保持している．また，各端末は自由に移動する環境を想定する．各端末は何らかの無線通信方式を用いて，通信範囲内の端末と双方向で通信できるものとし，通信範囲内に存在する端末を隣接端末と呼ぶ．

データモデル． ネットワーク内に存在するデータの集合を $O(= \cup O_i)$ とする．あるデータ $o_i \in O$ は，アプリケーションが予め設定したスコアリング関数 f によりスコア付けされており，$score(o_i) = f(o_i)$ である．簡単化のため，全てのデータのサイズは等しく，データの複製，データの消滅，および新たなデータの生成はないものとするが，データの更新は起こるものとする．そのため，データのスコアも更新される．

本研究では，アドホックネットワークを構成する各端末が，自身と他の端末のもつデータに対して Top-k 検索を行う環境を想定する．また，Top-k 検索における

検索条件は，予めアプリケーションが設定しており，各端末はその条件に従って Top-k 検索を行うものとする．そのため，本章では，データを 1 次元にマッピングすることができ，あるデータに対して，そのスコアは一意と考えることができる．本章における Top-k クエリを以下で定義する．

定義 1（Top-k クエリ）． 検索要求数 k，およびネットワーク内のデータ集合 O が与えられたとき，Top-k クエリは，O においてスコアが最も大きい k 個のデータを検索する．つまり，正解集合を *TOPK* とし，$o_i \in$ *TOPK*，$o_j \in O \backslash$ *TOPK* とすると，$score(o_i) \geq score(o_j)$ である．また，同じスコアのデータの順位の差別化は，任意の方法（例えば，データの識別子）により行われるとものの想定する．

本章では，ネットワーク内で実行される Top-k 検索において，各端末が指定できる k の最大値（k_{max}）は決まっており（例えば $k_{max} = 50$），その値を超える Top-k 検索は行われないものとする．

2.3 関連研究

アドホックネットワークでは，マルチホップ通信により，メッセージの転送が行われる [42, 61]．フラッディングによる転送では，無駄な転送やブロードキャストストーム問題などにより，通信量の増加やパケット衝突といった問題が生じるため，無駄な転送を抑制するためのルーティング手法が数多く提案されている [26, 43, 44]．また，データベース分野 [39] と同様に，P2P ネットワークや無線センサネットワークなどの分散ネットワークにおいても多くの Top-k 検索手法が提案されている．本節では，これらのうち，代表的な手法について紹介する．

2.3.1 モバイルアドホックネットワークにおけるルーティング手法

モバイルアドホックネットワークではネットワークトポロジの変化が生じるため，メッセージの転送経路を静的に決定できない．そのため，端末の移動に応じて動的に転送経路を変更する必要があり，効率的なルーティングのための研究が盛んに行われている．文献 [32, 55] では，一定のホップ数以内に存在する端末との

接続状況を常に管理し，そのホップ数を超える位置に存在する端末は，メッセージの転送要求が生じた場合に探索する，ハイブリッド型のルーティング手法を提案している．全ての端末との接続状況を管理する場合，オーバヘッドが非常に大きいため，接続状況の管理を，一定のホップ数以内に存在する端末に限定することにより，管理オーバヘッドを抑制できる．しかし，接続状況を管理するためには，全ての端末がビーコンを一定間隔で送信する必要があり，通信量の増加，およびパケット衝突の頻繁な発生の原因となる．文献 [51, 52, 53, 76] では，“メッシュ”の概念を利用したルーティング手法を提案している．ある端末がネットワーク内の別の端末（目的端末）にデータを転送する場合，これらの端末間でデータの転送経路を作成する．その際，目的端末から数ホップ以内に存在する端末がメッシュ端末の役割を果たす．メッシュ端末は，初めてデータを受信した際，ブロードキャストによりデータを転送する．これを初めて受信した他のメッシュ端末も同様にブロードキャストで転送することにより，目的端末から数ホップ以内のみの局所的なフラッディングを行う．これにより，端末が移動した場合においても，目的端末へデータを高確率で転送できる．アドホックネットワークでの Top-k 検索を考える場合，上位 k 個に含まれるデータを持つ端末は，最大 k 台である．これらの端末へクエリメッセージを転送するために，文献 [51, 52, 53, 76] と同様に，クエリメッセージの転送経路を構築すると，多くの端末がメッシュ端末となるため，フラッディングの範囲が拡大し，通信量の増加を引き起こす．これらからわかるように，モバイルアドホックネットワークにおいて検索に必要な端末のみで上位 k 個のデータを取得するためには，Top-k 検索に特化したルーティングが必要である．

2.3.2　P2P ネットワークにおける Top-k 検索手法

非構造 P2P ネットワークの分野では，これまでファイル共有やデータ検索などに関する研究が多く行われており，Top-k 検索に対してもいくつか手法が提案されている [1, 14, 50, 72].

文献 [14] で提案されている手法では，$i(\leq k)$ 回目の検索で i 位のデータを特定し，k 回の試行により上位 k 個のデータを取得する．検索クエリを発行する端末

は，全ての端末にクエリメッセージを送信し，クエリメッセージを受信した端末は，自身が持つデータ集合における 1 位のものを返信する．これにより，クエリ発行端末は 1 位のデータを特定できる．次に，1 位のデータを保持している端末に，その端末が持つデータ集合における 2 位のデータを返信してもらう．この手法は，不要なデータの返信は少ないが，上位 k 個のデータを取得するための試行回数が多く，検索時間が長い．モバイルアドホックネットワークでは，ネットワークトポロジ変化の影響を避けるため，検索に時間がかかるアプローチは有効でない．文献 [1] で提案されている手法はフラッディングを用いず，各端末が持つデータのスコアによってアクセスする端末を選択している．しかし，この手法では，任意の端末に 1 ホップでアクセスできる環境を想定しており，アドホックネットワークには適用できない．文献 [50, 72] では，クエリメッセージの転送経路上に存在する端末が持つデータ集合において，その上位 k 個のデータを返信する手法を提案している．この方法は，データを集約し，最終結果に含まれないデータの返信を削減することができる．しかし，アプローチとしては冒頭で紹介した単純な方法と本質的に同様であり，無駄なデータの返信によって生じる通信量は大きい．

2.3.3 無線センサネットワークにおける Top-k 検索手法

センサネットワークにおける Top-k 検索は，環境モニタリングなどに応用されている．センサ端末はバッテリ駆動のため，環境モニタリングを行う際は，ネットワークの長寿命化が求められる．そこで，多くの研究は，ネットワークの長寿命化を実現する手法を提案している．文献 [49] では，中継端末によるデータの集約方法を提案している．各端末は，自身が把握している上位 k 個のデータを返信し，これを受信した端末は，自身のデータと受信データの上位 k 個のデータを返信する．文献 [48] で提案されている手法も文献 [49] と同様の処理によりデータを返信するが，親端末のみでなく，兄弟端末へもデータの情報を送信する．これにより，不要なデータの返信をより多く削減できる．文献 [24] では，一定の検索精度を犠牲にしつつ，消費電力を削減する手法を提案している．文献 [69, 70] では，サンプリングにより検索結果に含まれるデータを推定する手法を提案しているが，

文献 [24] と同様に，正確な検索結果を保証しない．また，文献 [71] では，一定時間内に Top-k 検索を行うことを保証する手法を提案している．文献 [25, 57, 82, 83] では，フィルタを用いて，上位 k 個に含まれないデータの返信を抑制する手法を提案している．ネットワーク内の基地局端末は，受信データを基にフィルタを作成し，これをセンタ端末に送信する．センサ端末は，データの更新が発生した場合，それが上位 k 個に含まれるデータかどうか，フィルタから計算し，データを基地局端末に送信するか判断する．

上記の通り，k 位以内とならないデータの返信を削減し，電力消費を抑制する手法が数多く提案されている．しかし，センサネットワークにおける Top-k 検索手法では，検索クエリを発行する端末は基本的に 1 台の基地局端末のみである点，また，端末が固定であることを想定しているため，これらの手法をモバイルアドホックネットワークへ適用することは困難である．

2.3.4　モバイルアドホックネットワークにおける Top-k 検索手法

筆者の知る限り，モバイルアドホックネットワークにおける Top-k 検索手法は，文献 [33, 58, 63, 64, 65, 66] でのみ提案されている．文献 [58] での想定環境（仮想的な通貨など）は，経済モデル（economic scheme）を採用しており，Top-k 検索を行う端末は，仮想的な通貨を支払うことによりデータを得たり，報酬や罰則を与えることによりデータの送信確率を決定するなど，本論文で想定する環境と本質的に異なる．文献 [65, 66] では，データの複製が配置されている環境における Top-k 検索手法を提案しており，本章で想定する環境と異なる．文献 [33, 63, 64] では，ネットワーク内のデータの k 番目のスコアを推定することにより，上位 k 個に含まれないデータの返信を抑制している．しかし，これらの手法では，ネットワーク内の全ての端末がクエリメッセージ，およびクエリ応答を送信しており，検索に必要がない端末によるメッセージの転送が多く行われている．そのため，不要な通信量の増加によるパケットロスや，検索時間の増大などの問題が生じる．

2.4 提案手法

本節では，経路表を用いて，上位 k 個のデータ検索に必要がない端末へのメッセージ転送を削減する提案手法について説明する．まず，経路表の作成手順について説明し，その後，Top-k 検索，および経路表の更新方法について説明する．

2.4.1 経路表の作成

提案手法では，各端末は経路表を用いて Top-k 検索を行う．経路表は，データのスコアの順位（順位表），順位表中のデータの保持端末の識別子（保持端末リスト），および順位表中のデータを取得するためのクエリメッセージの送信先（宛先）により構成されており，ネットワーク内で初めて Top-k 検索を行う端末 N_{init} により，経路表の作成が開始される．

N_{init} は経路表作成メッセージをフラッディングし，このメッセージを初めて受信した端末は，送信元端末を自身の親端末として記録する．また，スコアの高い r_{max} 個のデータのスコア，およびデータの保持端末識別子を親端末に送信する．ここで，スコア更新に対応するため，$r_{max} > k_{max}$ とする．さらに，返信メッセージを受信した端末は，スコアの高い r_{max} 個のデータのスコア，その保持端末識別子，および受信メッセージの送信元端末を経路表作成リストとして記録し，どのデータがどの端末から返信され得るかを把握する．この手順により，N_{init} は経路表を作成することができ，経路表はスコアの降順によりソートされている．

N_{init} は順位表，および保持端末リストを添付したメッセージをフラッディングし，このメッセージを初めて受信した端末は，以下の手順で自身の経路表を作成する．

1. 受信した順位表，保持端末リストを経路表に格納する．
2. 全ての宛先をメッセージの送信元端末とする．
3. 自身が保持端末であるデータの宛先を自身に更新する．

順位	データのスコア	保持端末識別子	宛先
1	100	A	F
2	95	M	J
3	90	H	H
4	88	O	J
5	85	W	W
6	80	F	F
7	77	X	J
⋮	⋮	⋮	⋮
20	20	A	E

順位表　　保持端末リスト

図 2.1: 経路表の例

4. 経路表作成リスト中に含まれるデータと一致するデータの宛先を，経路表作成リストで記録した送信元端末に更新する．

2 番目の手順は，順位表の作成時に把握できなかったデータへの経路を補完するために行う．以上により，各端末は各データを検索する際のクエリメッセージの宛先となる端末を把握できる．図 2.1 は，図 2.2 における端末 I の保持する経路表を示す（$r_{max} = 20$）．この場合，I が 1 位のデータを検索する際のクエリメッセージの宛先は F であることを表している．

2.4.2　Top-k 検索

提案手法では，クエリ発行端末，およびクエリメッセージを受信した各端末は，検索要求順位を添付したクエリメッセージを，自身の経路表における各宛先にユニキャストする．また，各端末は要求されたデータのみを返信する．

検索クエリの転送． クエリメッセージは，$q_i =< N_{org}, N_{src}, L_{SR} >$ で構成されており，i はクエリの識別子，N_{org} はクエリ発行端末の識別子，N_{src} はクエリ送信端末の識別子，および L_{SR} は検索要求順位のリストである．クエリ発行端末は，経

路表における k 位までの各宛先に，それぞれ検索要求順位を添付したクエリメッセージを送信する．また，k 位までに同じ宛先が複数存在する場合，L_{SR} に検索要求順位をまとめることで，クエリメッセージの送信回数を不要に増やさない．クエリメッセージを初めて受信した端末は，その送信元端末（$q_i.N_{src}$）を親端末として記録し，クエリ発行端末と同様の操作によりクエリメッセージを転送する．検索要求順位として j 位を指定したクエリメッセージを受信した端末は，自身が j 位のデータを保持していない場合，自身の経路表における j 位の宛先にクエリメッセージを転送する．これにより，上位 k 個のデータの取得に必要な端末にのみクエリメッセージを転送することができる．

クエリ応答の返信．クエリ応答は，$r_i =< N_{src}, L_{data} >$ で構成されており，i はクエリの識別子，N_{src} はクエリ応答を送信する端末の識別子，および L_{data} は返信データのリストである．クエリメッセージを転送する必要がない端末，つまり，検索を要求されたデータを全て保持している端末は，親端末にクエリ応答を送信する．クエリ応答中の L_{data} には，要求されたデータのみ添付する．これにより，上位 k 個のデータ取得に必要のない端末へのクエリメッセージの転送を抑制し，不要なデータの返信による無駄な通信量が発生しない．クエリメッセージの中継端末は，自身が要求した順位のデータを全て受信するか，クエリメッセージを送信してから一定時間経過した場合，クエリ応答を親端末に送信する．さらに，クエリ応答を受信した端末は，自身の経路表とクエリ応答の送信元端末（$r_i.N_{src}$）を参照する．このとき，データの宛先とクエリ応答の送信元端末が異なる場合，そのデータの宛先をクエリ応答の送信元端末に更新する．

図 2.2 を用いて，提案手法における Top-k 検索例を説明する．端末 I が $k = 2$ の Top-k 検索を行うものとし，端末 A が 1 位のデータを保持し，端末 M が 2 位のデータを保持しているものとする．図における表は，各端末が保持する経路表において，各順位のデータに対する宛先を表したものである．I は，自身の経路表を参照し，1 位のデータを要求するクエリメッセージを宛先である F に，2 位のデータを要求するクエリメッセージを宛先である J にそれぞれ送信する．クエリメッセージを受信した各端末は，自身の経路表を参照し，要求された順位の宛先にクエリメッセージを転送する．クエリメッセージを受信した A，および M は，それぞれ

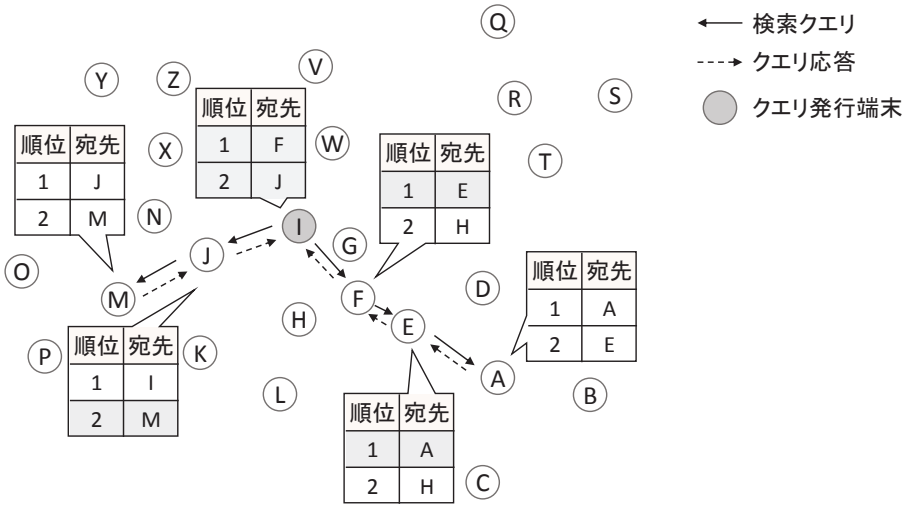


図 2.2: 提案手法における Top-k 検索例（$k = 2$）

1 位と 2 位のデータを保持しているため，要求されたデータのみを返信する．データはクエリメッセージの転送経路に沿って返信される．これにより，I は上位 2 個のデータを取得できる．

2.4.3　経路表の更新

提案手法では，無線リンク切断やデータのスコアの更新が起きた場合においても検索精度を維持するため，経路表を修正する必要がある．そこで本項では，経路表の修正方法について説明する．

リンク切断への対応．モバイルアドホックネットワークでは，端末の移動によりネットワークトポロジが動的に変化する．リンク切断が生じ，経路表における宛先にクエリメッセージを送信できない場合，およびクエリ応答を親端末に送信できない場合，検索精度が低下する．そのため，リンク切断を検出した端末は，経路探索メッセージを送信することにより，宛先，または親端末までの別経路を探索する．

経路探索メッセージの処理は，クエリメッセージ転送時，およびクエリ応答返信時において同様である．リンク切断を検出した端末は，TTL（Time To Live）を設定した経路探索メッセージをブロードキャストする．ここで，TTL はホップ数

Algorithm 1: 経路表修正アルゴリズム

Input: RT, $score(o_i)$, $p\text{-}score(o_i)$ //RT は経路表，$p\text{-}score(o_i)$ はデータ更新前のスコア
1 **case** $p\text{-}score(o_i) \geq RT[r_{max}-1]$.score and $score(o_i) \geq RT[r_{max}-1]$.score
2 Resort the recorded information in descending order of score
3 **case** $p\text{-}score(o_i) < RT[r_{max}-1]$.score and $score(o_i) \geq RT[r_{max}-1]$.score
4 Insert the information on o_i
5 Move down the rank of each data with score smaller than $score(o_i)$
6 **case** $p\text{-}score(o_i) \geq RT[r_{max}-1]$.score and $score(o_i) < RT[r_{max}-1]$.score
7 Delete the information on o_i
8 Raise the rank of each data with score smaller than $p\text{-}score(o_i)$

により転送範囲を指定する値である．経路探索メッセージを受信した端末は，自身がリンク切断先の端末，またはクエリ発行端末（クエリ応答返信時のみ）の場合，経路探索応答メッセージを返信する．それ以外の端末の場合，TTL を 1 減らし，その値が 1 以上であるならば経路探索メッセージをブロードキャストする．経路探索応答メッセージを受信した端末は，経路表におけるリンク切断先に該当する宛先を，経路探索応答メッセージの送信元端末に更新する．この処理により，リンク切断が起きた場合においても別経路を探索し，別経路の発見に成功した場合，クエリメッセージ，およびクエリ応答の転送を継続でき，検索精度を維持できる．

スコア更新への対応． データのスコアが更新された場合，データの順位が変わる可能性がある．提案手法では，自身のデータのスコアを更新した端末は，自身の経路表を修正し，順位が変わったデータを持つ端末にスコアの更新を通知する．データ o_i のスコアが更新された際，経路表を修正しなければならない場合が 3 通り（データの順位が変化する，新たに経路表に挿入する，経路表から削除する）ある．各端末は，アルゴリズム 1 に従って経路表を修正する．

自身のデータのスコアを更新した端末は，順位の変わったデータの宛先にスコア更新メッセージを送信し，このメッセージを受信した端末は，経路表を修正する．また，スコアが更新されたデータの宛先は，メッセージの送信元端末に更新する．その後，順位が変わったデータを持つ端末まで，データのスコアを更新した端末と同様の方法でスコア更新メッセージを転送する．これにより，経路表の修正がすぐに必要な端末（順位が変わったデータを持つ端末）にのみスコア更新を通知できる．また，スコア更新メッセージを受信しなかった端末は，Top-k 検索中

のクエリ応答の受信の際に経路表を修正できる．そのため，データのスコア更新通知にかかる通信量を削減しつつ，経路表の精度，および検索精度を維持できる．

2.4.4　提案手法の問題点

提案手法は，上位 k 個に含まれるデータの取得に必要な端末にのみクエリメッセージを送信することにより，通信量の削減を実現している．しかし，ネットワークトポロジの変化が激しい環境では，以下の要因により，検索精度が低下してしまうことが考えられる．

- 単一経路によるクエリメッセージの転送
  提案手法では，検索要求順位を添付したクエリメッセージを各宛先に送信する．そのため，各データまでのクエリメッセージの転送経路は 1 本となる．しかし，モバイルアドホックネットワークでは端末が自由に移動し，ネットワークトポロジが動的に変化するため，単一経路によるクエリメッセージの転送は信頼性に欠ける [87].

- 非効率な経路探索
  提案手法では，クエリメッセージ転送時にリンク切断が起きた場合，経路探索を行うことにより別経路を発見している．この手続きはメッセージの往復が必要であり，クエリメッセージを再転送するまでの遅延が生じる．この遅延により，新たなリンク切断が生じる可能性があり，クエリメッセージの転送率が低下してしまう．

以上の要因から，ネットワークトポロジが頻繁に変わる場合，提案手法では経路表に最新のネットワークトポロジを反映できず，検索精度が低下することが考えられる．2.5 節では，この問題を解決する拡張手法について述べる．

2.5　拡張手法

本節では，本章における提案手法の検索方法を拡張した手法について説明する．

拡張のアイデア．拡張手法は提案手法と同様の経路表を用いるが，複数経路による転送を行うことにより，クエリメッセージの転送率の向上を目指す．具体的には，各端末が経路表における k 位までの全ての宛先にクエリメッセージを送信する．2.4.1 項における説明のように，各端末は自律的に自身の経路表を構築するため，あるデータに対するクエリメッセージの宛先となる端末が分散する．そのため，k 位までの全ての宛先にクエリメッセージを送信すると，送信経路が分散し，上位 k 個のデータを持つ端末までのクエリメッセージの送信経路が複数となりやすい．しかし，宛先の数が多い場合，ユニキャストによる転送では通信量，および遅延が増加してしまう．そのため，拡張手法では経路表における k 位までの宛先に，マルチキャストによりクエリメッセージを送信する．さらに，受信，および傍受したクエリメッセージから，転送が不要な宛先を判断し，クエリメッセージの不要な転送を抑制しつつ，複数経路によるクエリメッセージの転送を実現する．

2.5.1 Top-k 検索

拡張手法では，経路表，および受信したクエリメッセージに添付されている情報から，自身の送信するクエリメッセージの転送先を決定する．また，クエリメッセージを受信した端末は，経路表における k 位のスコア以上のスコアをもつデータのみを返信する．本項では，Top-k 検索の手順について詳しく説明する．

検索クエリの転送．クエリメッセージは，$q_i =< N_{org}, k, L_{route}, QL_s, QL_p >$ で構成されている．L_{route} はクエリメッセージの転送経路上に存在する端末の識別子のリストであり，QL_s はクエリメッセージの宛先となる端末の識別子のリスト，QL_p はクエリメッセージの送信元端末の親が送信したクエリメッセージの QL_s である．

検索クエリの発行端末 N_{org} は，k を指定し，経路表における k 位までの宛先を QL_s に格納する．その後，クエリメッセージを送信する．なお，このメッセージにおいて，QL_p は $\emptyset$ である．また，L_{route} には N_{org} が格納されている．

クエリメッセージを受信した端末 N_j は，アルゴリズム 2 に従ってクエリメッセージの転送を行う．ここで，アルゴリズム 2 における $RT_j[i]$.QueryAddress は，N_j の経路表 RT_j の i 位の宛先を示す．また，$L_{neighbor}$ はクエリメッセージの送信元端

Algorithm 2: 検索クエリ転送アルゴリズム

1 **case** 検索クエリの受信
2 　**if** N_j receives the query for the first time **then**
3 　　N_j.parent $\leftarrow q_i.L_{route}[|q_i.L_{route}|-1]$
4 　　$N_j.L_{route} \leftarrow q_i.L_{route}$
5 　　**for** $i = 0$ to $k-1$ **do**
6 　　$N_j.QL_s \leftarrow RT_j[i]$.QueryAddress
7 　　$N_j.QL_s \leftarrow N_j.QL_s - q_i.QL_s - q_i.QL_p - q_i.L_{route}$
8 　　$N_j.QL_p \leftarrow q_i.QL_s$
9 　　**if** $N_p.QL_s = \emptyset$ **then**
10 　　Send r_i to N_j.parent
11 　　**else**
12 　　Set query timer
13 　**else**
14 　　$N_j.L_{neighbor} \leftarrow N_j.L_{neighbor} \cup < q_i.L_{route}[|q_i.L_{route}|-1], |q_i.L_{route}| >$
15 　　$N_j.QL_s \leftarrow N_j.QL_s - q_i.L_{route}[|q_i.L_{route}|-1]$
16 **case** 検索クエリの送信
17 　**if** query timer expired **then**
18 　　**if** $N_j.QL_s \neq \emptyset$ **then**
19 　　$q_i.L_{route} \leftarrow N_j.L_{route} \cup \{N_j\}$
20 　　$q_i.QL_s \leftarrow N_j.QL_s$
21 　　$q_i.QL_p \leftarrow N_j.QL_p$
22 　　Send q_i
23 　　**else**
24 　　Send r_i to the N_j.parent

末，およびその端末のクエリ発行端末からのホップ数のリストである．初めてクエリメッセージを受信した端末は，クエリメッセージの送信元端末に ACK を送信する．ACK を受信した端末は，その送信元端末を子端末に追加する．クエリメッセージを受信した端末は，上位 k 個の全てのデータを検索しようとする（5–6 行）．これにより，クエリメッセージの転送経路を複数にできる．さらに，各端末は受信したクエリメッセージに含まれる QL_s，QL_p，および L_{route} から，すでにクエリメッセージが転送されている端末を把握できるため，これらを宛先から除外する（7 行）．これにより，不要な転送を抑制しつつ，提案手法に比べてクエリメッセージの転送率を向上できる．

クエリ応答の返信． クエリ応答は，提案手法のものと同様であり，自身のクエリメッセージにおける QL_s が $\emptyset$ である端末 N_j は，クエリ応答の返信を開始する．ク

Algorithm 3: クエリ応答返信アルゴリズム

```
1  case クエリ応答の受信
2      for ∀o ∈ r_i.L_data do
3          flag ← 0
4          for x = 0 to r_max − 1 do
5              if score(o) = RT_j[x].score and o.holderID = RT_j[x].holderID then
6                  flag ← 1
7                  if RT_j[i].QueryAddress ≠ r_i.N_src then
8                      if ∃RT_j[i].QueryAddress ∈ N_j.children then
9                          N_j.children ← N_j.children −RT_j[i].QueryAddress
10                     RT_j[i].QueryAddress ← r_i.N_src
11         if flag = 0 //経路表に含まれていないデータを受信 then
12             Update RT_j //アルゴリズム 1
13     if N_q receives r_i from all children or pre-determined time has elapsed or N_q.QL_s = ∅ then
14         /* クエリ応答の送信 */
15         r_i.L_data ← {∀o ∈ O_q ∪ N_j.L_data | score(o) ≥ RT_j[k − 1].score}
16         r_i.N_src ← N_j
17         Send r_i to N_j.parent
18 case リンク切断時の処理
19     if N_j detects a link disconnection between itself and N_j.parent then
20         if N_j.L_neighbor ≠ ∅ then
21             N_j.parent ← {∃N_r ∈ N_j.L_neighbor | N_r.hopCt is the minimum in N_j.L_neighbor}
22             N_j.L_neighbor ← N_j.L_neighbor − N_j.parent
23             Send r_i to N_j.parent
24         else
25             Discard r_i
```

エリ応答の返信は，アルゴリズム 3 に従って行う．

各端末は，クエリ応答の受信により経路表を修正（2–12 行）し，経路表の精度を維持する．また，親端末とのリンク切断を検出した場合，$L_{neighbor}$ からクエリ発行端末までのホップ数が最小の隣接端末を選択し，その端末にクエリ応答を送信することにより，検索精度の低下を抑制する（19–25 行）．提案手法では，クエリ応答返信時にリンク切断を検出した場合，経路探索を行い，親端末までの別経路を発見する必要があった．しかし，複数経路によるクエリメッセージの転送を行う拡張手法では，隣接端末を把握している場合，経路探索を行わず，隣接端末にクエリ応答を再送する．そのため，経路探索にかかる通信量，および遅延を抑制でき，ネットワークトポロジの変化に対応しやすい．

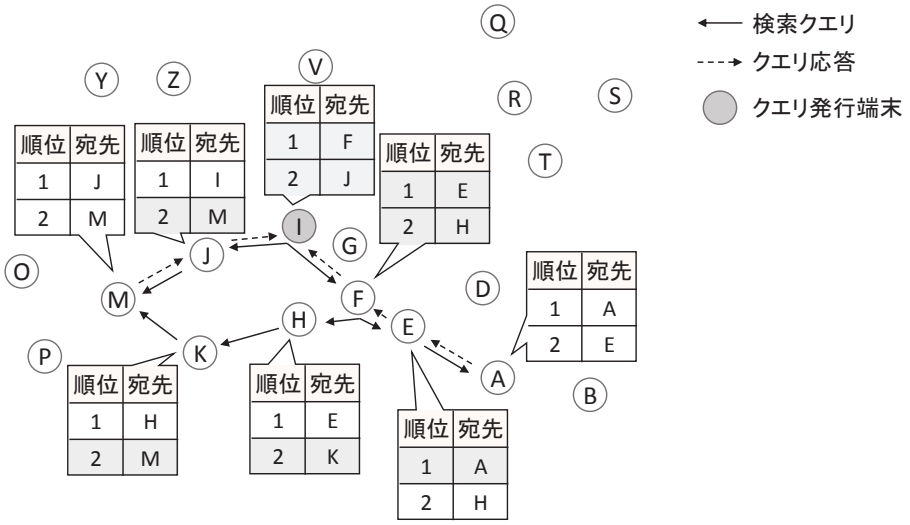


図 2.3: 拡張手法における Top-k 検索例（$k = 2$）

図 2.3 を用いて，拡張手法における Top-k 検索例（$k = 2$）について説明する．クエリ発行端末 I は，2 位までの宛先である端末 F と J にクエリメッセージを送信する．これを受信した端末 F は，同様に自身の経路表における 2 位までの宛先にクエリメッセージを送信する．ここで，J は 1 位の宛先が親端末の I であるため，端末 M にのみクエリメッセージを送信する．同様に，クエリメッセージを受信した端末は，不要な転送先を自律的に判断しつつ，クエリメッセージを送信する．図からわかるように，拡張手法は，端末 M まで複数経路によりクエリメッセージを転送している．例えば，端末 M–J 間でリンク切断が起きた場合においても，M は K からクエリメッセージを受信できるため，クエリメッセージの転送失敗を抑制できる．また，k 位以内のデータを保持している端末は，親端末にデータを返信し，クエリ発行端末 I は上位 k 個のデータを取得できる．

2.5.2　経路表の更新

リンク切断への対応． クエリメッセージ転送時に経路表における宛先とのリンク切断を検出した場合，経路探索メッセージ（*RREQ*）を送信することにより，別経路を発見する．拡張手法では，RREQ に自身の送信するクエリメッセージの情報を添付する．これにより，経路探索応答メッセージ（*RREP*）受信後に，クエリメッ

Algorithm 4: 経路探索アルゴリズム

```
1  case RREQ の受信
2      if N_j' receives RREQ for the first time then
3          N_j'.sender_node ← RREQ.N_src
4          if N_j' ∈ RREQ.N then
5              Send RREP to N_j'.sender_node
6              if N_j' has not received the query message then
7                  Execute Algorithm 2
8          else
9              if ∃N_dest ∈ RREQ.N, N_j'.L_neighbor ∪ N_j'.children then
10                 Send RREP to N_j'.sender_node
11             else
12                 if RREQ.TTL > 0 then
13                     TTL ← TTL −1
14                     Broadcast RREQ
15 case RREP の受信
16     Update the query addresses of corresponding (destination node's) ranks to the sender node
17     if N_j is the route request node then
18         if all destination nodes have already received the query message then
19             Send r_i to N_j.parent
20     else
21         Send RREP to N_j.sender_node
```

セージを再転送する必要がない．その結果，経路探索の遅延を可能な限り抑制しつつ，クエリメッセージの転送率を向上できる

端末 N_j が，自身の経路表内の宛先への別経路を探索する際に行う経路探索は，アルゴリズム 4 に従う．アルゴリズム 4 における，RREQ.N は，N_j がリンク切断を検出した宛先の集合を表す．経路探索のアルゴリズムは，基本的に提案手法と同様である．しかし，拡張手法では複数経路によりクエリメッセージを転送しているため，RREQ を受信した端末がリンク切断先の端末でない場合においても，リンク切断先の端末が，自身の隣接端末や子端末であるならば RREP を返信できる．つまり，リンク切断先の端末がすでにクエリメッセージを受信済みであるか把握している（4–10 行）．これにより，RREQ の不要な転送を抑制し，経路探索にかかる通信量，および遅延を抑制できる．

スコア更新への対応．提案手法では，データのスコアが更新された際，順位の変

化したデータの宛先にスコア更新メッセージを送信している．しかし，この方法ではクエリメッセージを転送する際，経路表を更新した端末を経由する必要があるため，順位の更新されたデータまでの経路が遠回りとなり，通信量が不要に発生してしまう．そこで拡張手法では，スコア更新メッセージを，経路表における k_{max} 位までの宛先に送信する．k_{max} 位までの宛先は，Top-k 検索中に更新されることが多いため，信頼性が高い．スコア更新メッセージを受信および傍受した端末も自身の経路表を修正し，このメッセージの宛先である端末は，経路表における k_{max} 位までの宛先にスコア更新メッセージを送信する．このとき，クエリメッセージ転送時と同様に不要な転送先は送信先から除外する．また，傍受時に経路表を修正することにより，自身が宛先となるメッセージや Top-k 検索を経ることなく，スコア更新メッセージの転送経路中に存在する端末を宛先に設定できる．これにより，遠回りとなる経路が構築されることを抑制する．経路表の修正方法については，アルゴリズム 1 に従う．

2.6　性能評価

本節では，提案手法，および拡張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す．本実験では，提案手法，拡張手法，および文献 [64] で提案された 2 フェーズ検索手法を評価し，ネットワークシミュレータ Qualnet4.0[68] を用いた．

比較手法． 2 フェーズ検索手法は，モバイルアドホックネットワークにおいて，本章と同じ想定のもとにおける Top-k 検索の既存手法 [33, 63, 64] の中で最も性能が良いことが示されている [64]．そこで，提案手法および拡張手法の有効性を示すため，2 フェーズ検索手法を比較手法とした．2 フェーズ検索手法は，検索フェーズと収集フェーズに分けて Top-k 検索を行う．まず，検索フェーズにおいて，クエリ発行端末がクエリメッセージをフラッディングする．このフェーズでは，ネットワーク内のデータのスコアを取得し，取得したスコアの k 番目の値を閾値に設定する．その後，収集フェーズにおいて，設定した閾値を添付したクエリメッセージをフラッディングし，これを受信した各端末は閾値以上のスコアをもつデータ

表 2.1: パラメータ設定

パラメータ	値
k（要求データ数）	1, 5, 10, 15, 20, 25, **30**, 35, 40, 45, 50
v（端末の移動速度）	0, **0.5**, 1.0, 1.5, 2.0, 2.5, 3.0
I_s（スコアの更新周期）	15, 30, 60, **90**, 120, 150, 300, 600

のみを返信する．この手法では，閾値を設定することにより，不要なデータの返信を抑制することができる．

シミュレーション環境． 800[m]×500[m] の 2 次元平面状の領域に 100 台の端末が存在する．各端末はランダムウェイポイント [19] に従い，v[m/sec] の速度で移動する．停止時間は 60[sec] とした．各端末は，IEEE802.11b を使用し，伝送速度 11[Mbps]，通信伝搬距離が 100[m] 程度となる送信電力でデータを送信する．ネットワーク内には，128[Byte] のデータが 2,000 個存在するものとし，各端末はそれぞれ 20 個のデータを保持するものとした．データのスコアは正規分布に従い，取り得る値は [1，999] の範囲に含まれる整数とした．また，データのスコアは，I_s[sec] ごとに更新される．ここで，提案手法，および拡張手法において，経路表に保持する順位 r_{max} を 100 とし，クエリメッセージ転送時やクエリ応答返信時にリンク切断を検出した際の経路探索の TTL は 2 とした．

本実験のパラメータを表 2.1 に示す．太字の値を基本的には用い，その影響を調べる際は，表中の値に変化させる．以上のシミュレーション環境において，各端末の初期位置をランダムに決定し，3[sec] ごとにランダムに選ばれた端末が Top-k 検索を行うものとした．シミュレーション開始から 600[sec] 経過した際の以下の評価値を調べた．

- 検索精度
  順位付き検索結果の性能を測る MAP(Mean Average Precision)[18] の値を検索精度とする．MAP は，各クエリの平均精度 AP(Average Precision) を平均

化したものである．AP および MAP は以下の式で求める．

$$AP_i = \frac{1}{k}\sum_{j=1}^{k}\frac{x}{j}\cdot e$$

$$MAP = \frac{1}{querynum}\sum_{i=1}^{querynum} AP_i$$

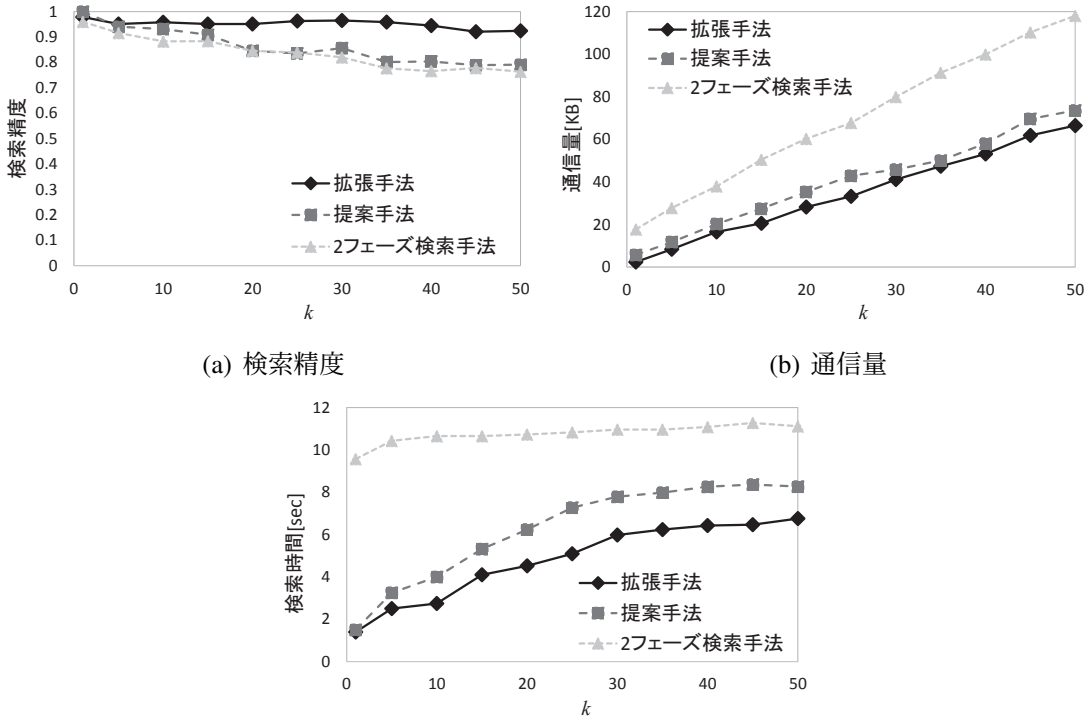
AP_i は i 番目のクエリの平均精度である．x は，取得した解の上位 j 個のうち正解集合の j 位以内である解の個数である．$querynum$ はクエリの発行数を示す．また，e は，j 番目の解が正解集合に含まれる場合に 1 であり，含まれない場合は 0 である．したがって，MAP は，より上位のデータを取得できているほど，高い値となる．

- 通信量
  発行した全ての検索クエリを処理するために送信された全てのメッセージの総バイトをクエリ発行数で割った値を通信量とする．

- 検索時間
  全検索クエリに対する，クエリ発行端末が検索クエリを発行してから検索結果を取得するまでの平均時間を検索時間とする．

k の影響． 要求データ数 k を変化させた時の各手法の性能を図 2.4 に示す．これらの図において，横軸は要求データ数 k を表し，縦軸は図 2.4(a) では検索精度，図 2.4(b) では通信量，および図 2.4(c) では検索時間を表す．

図 2.4(a) より，提案手法，および拡張手法は 2 フェーズ検索手法よりも検索精度を高く維持できていることがわかる．さらに，拡張手法は提案手法よりも高い検索精度を達成している．これは，複数経路によるクエリメッセージの転送，および効率的な経路探索により，クエリメッセージおよびクエリ応答の転送率が提案手法よりも高くなるためである．ここで，2 フェーズ検索手法は，検索フェーズ時のパケットロスの影響により，指定した閾値が正確でない場合がある．そのため，上位 k 個に含まれないデータの返信による無駄な通信量が生じ，収集フェーズ時にパケットロスが生じやすく，精度が低下する．



(a) 検索精度

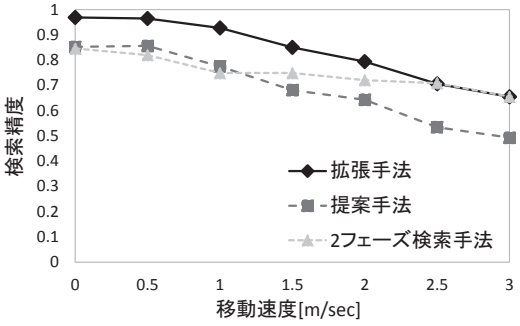
(b) 通信量

(c) 検索時間

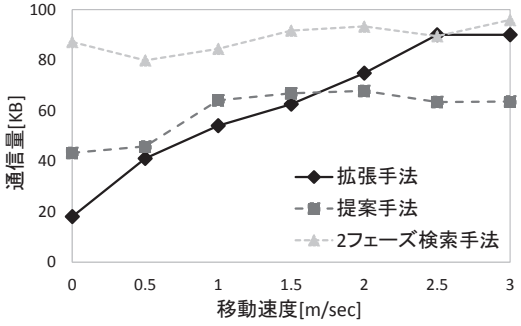
図 2.4: 要求データ数 k の影響

図 2.4(b) より，k が大きくなると全ての手法において通信量が大きくなっている．これは，k が大きくなると返信データ数が多くなり，クエリ応答のメッセージサイズが大きくなるためである．提案手法では，各宛先にユニキャストによりクエリメッセージを送信している．一方，拡張手法では，マルチキャストにより，全ての宛先に同時にクエリメッセージを転送している．拡張手法は複数経路を構築しているが，このマルチキャストの効果により，クエリメッセージの送信にかかる通信量を小さく抑えている．さらに拡張手法では，クエリ応答返信時にリンク切断が生じた場合，経路探索をせずに隣接端末に再送できる場合が多いため，経路探索にかかる通信量を抑制することができている．ここで 2 フェーズ検索手法は，フラッディングによるクエリメッセージの転送，および，上位 k 個に含まれないデータの返信が生じるため，他の 2 つの手法よりも通信量が大きい．

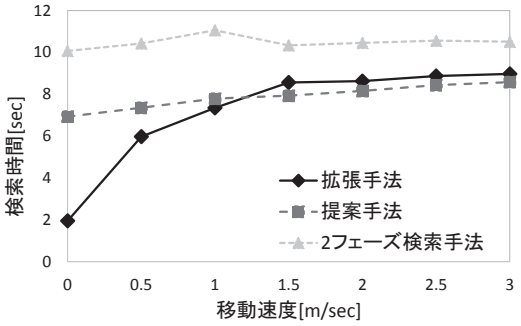
図 2.4(c) より，提案手法，および拡張手法は検索時間が 2 フェーズ検索手法より



(a) 検索精度



(b) 通信量



(c) 検索時間

図 2.5: 端末の移動速度 v の影響

も短い．提案手法および拡張手法は，2 フェーズ検索手法に比べて子端末数が少なく，子端末からのクエリ応答を待つ時間が短い．また，提案手法では単一経路によるクエリメッセージの転送を行っているため，リンク切断の機会が多く，経路探索による遅延が大きい．これらの理由から，拡張手法の検索時間が最短となる．

v の影響． 移動速度 v を変化させた時の各手法の性能を図 2.5 に示す．これらの図において，横軸は移動速度 v を表し，縦軸は図 2.5(a) では検索精度，図 2.5(b) では通信量，および図 2.5(c) では検索時間を表す．

図 2.5(a) より，拡張手法は提案手法より検索精度を高く維持していることがわかる．拡張手法では複数経路によるクエリメッセージの転送を行っているため，提案手法に比べてクエリメッセージの転送率が高い．さらに，拡張手法では経路探索時の RREQ にクエリメッセージの情報を添付しているため，クエリメッセージの転送遅延を可能な限り抑制している．これにより，ネットワークトポロジの変

化により対応できるため，検索精度を高く維持できている．

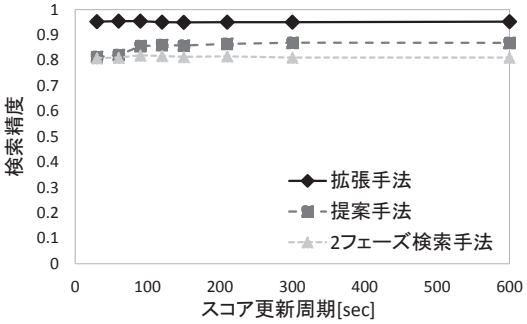
図 2.5(b) より，拡張手法では v が大きくなると，通信量が大きくなることがわかる．v が大きくなると，無線リンク切断の機会が多くなり，拡張手法，および提案手法では多くの端末が経路探索を行う．v が大きい場合，拡張手法は提案手法よりも通信量が大きい．これは，経路探索時の RREQ にクエリメッセージの情報を添付しているため，そのサイズが大きくなることが一因である．さらに，提案手法では図 2.5(a) からわかるように，検索精度が低く，拡張手法に比べてクエリ応答の返信数が少ない．その結果，提案手法の方が通信量が小さくなっている．

図 2.5(c) より，拡張手法および提案手法は，v が大きくなると検索時間が長くなることがわかる．v が大きくなると，無線リンク切断が多くなり，中継端末でクエリ応答が破棄される機会が多くなる．ここで 2 フェーズ検索手法は，多くの端末が最大待ち時間までクエリ応答を待ち，さらに 2 往復による検索を行うため，提案手法，および提案手法よりも検索時間が長くなる．

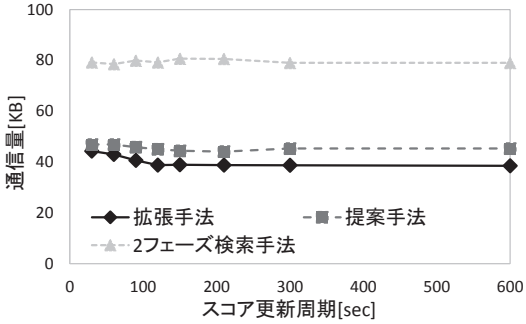
I_s の影響． スコア更新周期 I_s を変化させた時の各手法の性能を図 2.6 に示す．これらの図において，横軸はスコア更新周期 I_s を表し，縦軸は図 2.6(a) では検索精度，図 2.6(b) では通信量，および図 2.6(c) では検索時間を表す．

図 2.6(a) より，拡張手法では I_s が短い場合でも検索精度を高く維持できていることがわかる．これは，拡張手法におけるスコア更新メッセージは，経路表の更新頻度の高い k_{max} 位までの宛先に送信することから，経路表の精度を維持できるためである．提案手法におけるスコア更新メッセージの転送方法では，クエリメッセージの転送経路を遠回りのものにしてしまう．そのため，クエリメッセージ，およびクエリ応答の転送ホップ数が大きくなり，パケットロスが起きやすくなることも検索精度を低下させる要因となる．

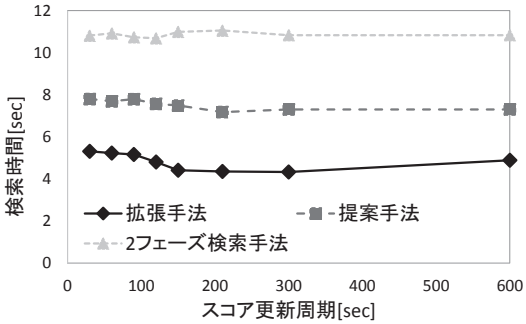
図 2.6(b)，および図 2.6(c) より，拡張手法では I_s が小さい場合，通信量が大きく，検索時間が長くなることがわかる．これは，スコア更新メッセージを経路表の k_{max} 位までの宛先に送信することから，クエリメッセージとのメッセージ衝突が生じ易く，再送の影響で通信量，および，遅延が増加するためである．



(a) 検索精度



(b) 通信量



(c) 検索時間

図 2.6: スコアの更新周期 I_s の影響

2.7　むすび

本章では，モバイルアドホックネットワークにおいて，検索に必要のない端末へのクエリメッセージの転送を抑止する Top-k 検索のためのルーティング手法を提案した．提案手法では，各端末が上位のデータを取得するためのクエリメッセージの宛先を管理した経路表を保持する．Top-k 検索の際は，要求するデータの順位を添付したクエリメッセージを，そのデータに対応する宛先に送信する．クエリメッセージを受信した端末も同様の操作を行うことにより，検索に必要な端末のみで Top-k 検索を行える．さらに，提案手法における単一経路でのクエリメッセージの転送方法が，ネットワークトポロジの変化が大きい場合に検索精度の低下を招くことを考慮して，複数経路でクエリメッセージを転送する拡張手法も提案した．拡張手法では，経路表の k 位までの宛先にクエリメッセージをマルチキャスト

し，これを受信した端末も同様の処理を行いつつ，受信したクエリメッセージに添付されている情報から不要な転送先を除外する．拡張手法は，クエリメッセージを複数経路で転送できるため，提案手法よりもクエリメッセージの転送率を向上できる．シミュレーション実験の結果から，提案手法，および拡張手法は，既存手法よりも小さい通信量で検索精度を維持していることを確認した．

拡張手法では，ネットワークトポロジの変化が比較的に小さい場合，検索精度を高く維持することができ，通信量および検索時間を共に小さくすることができる．つまり，アドホックネットワークを構成する端末を持つユーザの移動が局所的な場合などにおいて効果的である．一方，ネットワークトポロジの変化が頻繁に起こるような環境では，検索精度が低下してしまう．基本的に，Top-k 検索を行うユーザは，高い検索精度，および短い検索時間を望むことが考えられる．そのため，ネットワークトポロジが頻繁に変化する環境において，通信量および検索時間が小さく，かつ高い検索精度を維持できる方法が求められる．第 3 章では，この問題について取り組む．

第3章　クラスタを用いたTop-k検索のためのルーティング手法

3.1　まえがき

モバイルアドホックネットワークでは，ネットワークを構成する端末の移動によりネットワークトポロジが頻繁に変化する場合がある．このとき，クエリメッセージをフラッディングにより転送すると，ネットワークトポロジ変化の影響を受けないが，端末数が多い場合にはクエリメッセージの転送にかかる通信量が大きくなる．1 章で述べたように，アドホックネットワークで Top-k 検索を実行する際には，通信量を可能な限り小さくする必要があるため，ネットワークトポロジが頻繁に変化する場合や端末数が多い場合においても，必要な端末のみで Top-k 検索を行い，高い検索精度を維持することが望ましい．そこで本章では，上記のような環境においても，高い検索精度を維持しつつ，小さい通信量かつ短い検索時間で Top-k 検索を行う，クラスタを用いたルーティング手法である CTR[1]を提案する．

CTR では，上位となるデータを持つ端末がクラスタヘッドとなるようにクラスタリングを行う．クラスタヘッド間でのクエリルーティングを行うため，クエリメッセージは複数のクラスタに属しているゲートウェイノードを中継して送信される．クラスタヘッドは，上位となるデータまでのホップ数を管理し，自身が検索すべきデータを自律的に判断する．これにより，検索精度を維持しつつ，クエリメッセージの不要な転送を抑制できる．CTR では，クラスタヘッドのみが定期的にメッセージを送信し，これを受信した端末は自身が属するクラスタを把握できる．その結果，小さい通信量でトポロジ変化への対応，およびクエリメッセージの転送先の管理を行えるため，CTR の性能はトポロジ変化やクエリの発行頻度，お

[1] Cluster-based Top-k query Routing

よび端末数に大きく依存しない．また，メッセージの送信先となる端末とのリンク切断が生じた場合にも，経路探索を行わずにメッセージを転送できるため，通信量が小さく，低遅延な Top-k 検索を実現できる．

以下では，3.2 節で関連研究を紹介し，3.3 節で提案手法について述べる．3.4 節でシミュレーション実験の結果を示し，最後に 3.5 節で本章のまとめを行う．なお，本章で想定する環境は，2 章のものと同様である．

3.2　関連研究

文献 [16, 30, 60, 86] では，1 ホップクラスタリングによるルーティング手法を提案している．1 ホップクラスタリングは，クラスタの管理が容易であるため，モバイルアドホックネットワークに適している．そのため，CTR においても 1 ホップクラスタリングを行う．文献 [30] では，端末識別子を用いたクラスタリングアルゴリズムを提案している．この手法では，隣接端末中で，最も識別子の大きい端末がクラスタヘッドとなる．文献 [60] では，隣接端末の数に基づいたクラスタリング手法が提案されており，隣接端末の数が最大である端末がクラスタヘッドとなる．文献 [16] では，端末の移動速度からクラスタヘッドを決定する手法を提案している．この手法では，クラスタのメンテナンスの頻度を小さくするため，隣接端末と自身の移動速度を比較し，相対的な移動速度が最も小さい端末がクラスタヘッドとなる．また，これらのパラメータを重み付けした値からクラスタヘッドを決定する手法が，文献 [86] で提案されている．これらの手法では，ある端末へのデータ転送の効率化のためにクラスタリングを行っている．単純にこれらの手法を適用した場合にも，クエリメッセージのフラッディングは抑制できるが，端末が持つデータを考慮していないため，全てのクラスタヘッドにクエリメッセージを送信する必要があり，無駄な通信が発生してしまう．CTR では，上位となるデータを持つ端末をクラスタヘッドとすること，およびクラスタヘッドが上位データまでのホップ数を管理することにより，クラスタ内の端末，および隣接クラスタヘッドへの不要なクエリメッセージの転送を削減できる．

3.3 提案手法

本節では，CTR における Top-k 検索のためメッセージ処理方法について説明する．まず，CTR のクラスタリングについて説明し，その後，Top-k 検索およびスコア更新時におけるメッセージ処理方法について説明する．

3.3.1 クラスタの構築

2 章の拡張手法と同様の方法により，ネットワーク内で初めて Top-k 検索を行う端末 N_{init} は，順位表，および保持端末リストを作成する．N_{init} は，これらをフラッディングによりネットワーク全体に送信し，上位データの情報を全ての端末で共有する．順位表を受信した k_{max} 位以内のデータを持つ端末（データ所持端末と呼ぶ）は，データの順位が高いほど発火が早くなるようにタイマを設定する．タイマが発火した場合，クラスタヘッド（CH）となり，*ClusterHead-Announce*（以下 CHA）を送信する．これにより，上位データを保持する端末がクラスタヘッド（CH）となる．CHA を受信した端末は，自身のクラスタヘッドリスト ($List_{CH}$) に，CH の識別子を格納する．ここで，タイマを設定している端末が，タイマの発火前に CHA を受信した場合，タイマを破棄する．一定時間内に CHA を受信しなかったデータ所持端末以外の端末は，タイマを設定（端末識別子が小さいほど発火が早い）し，上記と同じ手順を行う．これにより，上位データを持つ端末が基本的に CH となるため，上位 k 個のデータを持つ端末間によるクエリメッセージのルーティングを実現できる．CH は，一定時間（t 秒）ごとに CHA を送信することにより，各端末は自身の所属するクラスタの CH を把握できる．また，$(\alpha \times t)$ 秒ごとに上記と同様の操作を行うことにより，クラスタの再構築を行い，ネットワークトポロジの変化に対応する．1 ホップクラスタでは，クラスタ内の端末は CH との無線リンクのみを把握すればよいため，文献 [4, 5] のように，多数の隣接端末とのリンクを管理する必要がなく，トポロジ変化に対して寛容になる．

各端末が，図 2.1 の順位表で表されるデータを保持している場合において，図 3.1 を用いてクラスタを構築する例を説明する．黒色の丸は CH を示し，灰色の丸はデータ所持端末を示す．白色の丸はそれ以外の端末を示す．また，点線の円は，

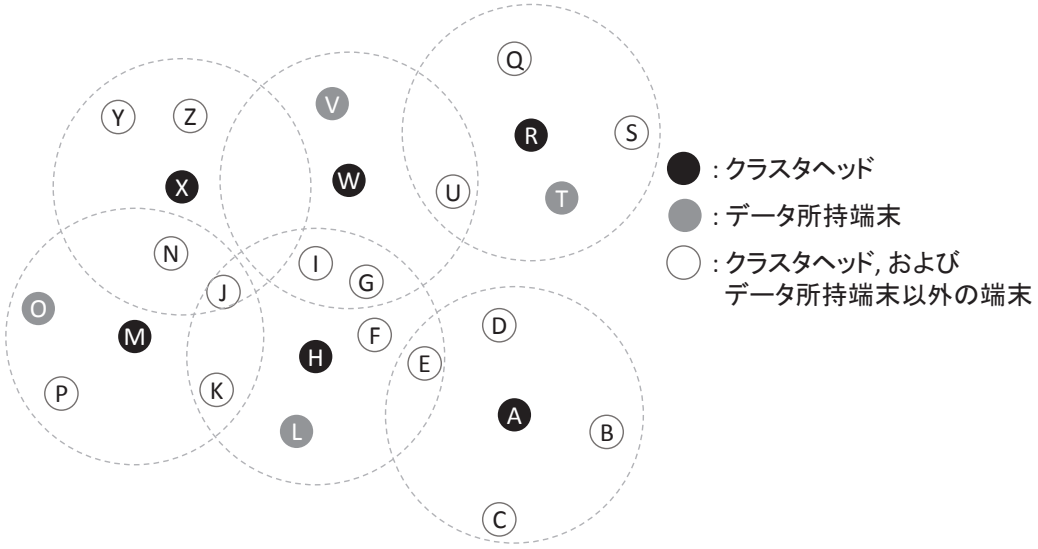


図 3.1: クラスタ構築の例

CH を中心とするクラスタの様子を示す．提案手法では，高順位のデータを持つ端末ほど早くタイマが発火する．そのため，まず 1 位のデータを持つ端末 A が CHA をブロードキャストし，これを受信した B，C，D，および E が $List_{CH}$ に A を加える．次に，2 位のデータを持つ端末 M が，その次に 3 位のデータを持つ端末 H が，A と同様の操作を行う．これにより，E，J，および K はゲートウェイノード（GW）となる．ここで，データ所持端末でない端末 R，S，T，および U は一定時間内に CHA を受信できないため，CHA の送信タイマを設定する．このとき,（辞書順において）識別子が最も小さい R が最短で CHA を送信し，CH となる．

3.3.2　Top-k 検索のためのメッセージ処理

本項では，CTR における Top-k 検索のためのメッセージ処理方法について説明する．

検索クエリの転送． CTR では，上位 k 個のデータを取得するために必要な端末間で検索クエリを処理するために，CH 間でクエリルーティングを行う．この時，複数のクラスタに属しているゲートウェイノード（GW）が CH の送信したクエリメッセージを中継する．CTR では，CH が上位データまでのホップ数をホップ数リスト

（$List_{hop}$）[2]を用いて保持する．クエリメッセージを初めて受信したCHは，受信したクエリメッセージに添付されている$List_{hop}$と，自身の$List_{hop}$を比較し，ホップ数が小さい順位のデータを検索する．これにより，クエリメッセージの不要な転送を抑制できる．ここで，クラスタ構築時において，CHが保持する$List_{hop}$における各データまでの初期ホップ数は∞とする．CHは，クエリメッセージの転送中にも，クエリメッセージに添付された情報，および順位表からデータまでのホップ数を把握することにより，$List_{hop}$の更新を行う．

クエリメッセージは，$q_i =< N_{org}, k, L_{route}, Dest_{CH}, List_{hop} >$で表される．$Dest_{CH}$は，クエリメッセージの送信先となるCHの識別子のリストである．（CHがクエリメッセージを送信する場合，そのクエリメッセージに含まれる$Dest_{CH}$は空である．）クエリ発行端末がCHである場合，クエリメッセージをブロードキャストする．また，GWである場合，自身が把握する全てのCHにクエリメッセージを送信する．クエリメッセージを受信した端末N_jがCH，およびGWの場合の処理をそれぞれアルゴリズム5，およびアルゴリズム6に示す．基本的に，CHはクエリメッセージをブロードキャストし，GWは，受信したクエリメッセージを，$Dest_{CH}$で指定したCHに転送する．

添付されている$Dest_{CH}$に，自身の識別子が含まれるクエリメッセージを初めて受信したCHは，クエリメッセージの送信元端末を親端末（N_j.parent）として記録し，クエリメッセージに添付されている経路リストを記録する（アルゴリズム5，1–4行）．その後，受信したクエリメッセージに含まれる$List_{hop}$と自身の$List_{hop}$を比較しながら検索データの絞り込みを行う（アルゴリズム5，6–11行）．各CHは，$N_j.SR$に自身が検索するデータの順位を格納し，クエリメッセージの送信の必要性を確認する（$N_j.SR = \emptyset$の場合，クエリメッセージを送信しない）ため，クエリメッセージの不要な送信を抑制できる．また，クラスタ構築後のTop-k検索では，CHが保持する$List_{hop}$における，各順位のデータに対するホップ数は基本的に∞である．そのため，受信したクエリメッセージに添付されている$List_{hop}$と自身の$List_{hop}$において，同じ順位のデータに対するホップ数が共に∞である場合，その順位を$N_j.SR$に追加する（アルゴリズム5，10–11行）．さらに，クエリメッセー

[2]特定のCHにクエリメッセージを転送するために必要なGW数をホップ数とする．つまり，図3.1において，AからH，B，およびXまでのホップ数は，それぞれ1，0，および2となる．

Algorithm 5: CH によるクエリ転送アルゴリズム

```
1  if ∃N_j ∈ q_i.Dest_CH then
2      if N_j receives for the first time then
3          N_j.parent ← q_i.L_route[|q_i.L_route| − 1]
4          N_j.L_route ← q_i.L_route
5          N_j.SR ← ∅
6          for x = 0 to |q_i.List_hop| − 1 do
7              y ← q_i.List_hop[x].rank
8              if N_j.List_hop[y − 1].hopCt < q_i.List_hop[x].hopCt and N_j ≠ yth rank data holder then
9                  N_j.SR ← N_j.SR ∪ {y}
10             if N_j.List_hop[y − 1].hopCt = ∞ and q_i.List_hop[x].hopCt = ∞ then
11                 N_j.SR ← N_j.SR ∪ {y}
12         if N_j.SR ≠ ∅ then
13             q_i.List_hop ← ∅
14             for x = 0 to |N_j.SR| − 1 do
15                 y ← N_j.SR[x]
16                 q_i.List_hop ← q_i.List_hop ∪ {< y, N_j.List_hop[y − 1].hopCt>}
17             q_i.L_route ← q_i.L_route ∪ {N_j}
18             q_i.Dest_CH ← ∅
19             Broadcast q_i
20         else
21             Execute Algorithm 7
22     else
23         if q_i.L_route[|q_i.L_route| − 2] = N_j then
24             N_j.children ← N_j.children ∪ q_i.Dest_CH
25         else
26             N_j.neighbor ← N_j.neighbor ∪ q_i.L_route[|q_i.L_route| − 1]
```

ジに含まれる経路リスト，および $Dest_{CH}$ から順位表に含まれる端末までのホップ数を把握でき，クエリメッセージ転送中に自身の $List_{hop}$ を更新できる．また，GW が転送したクエリメッセージが，自身が送信したものであった場合，その転送先の CH（$Dest_{CH}$）を子 CH として記録（アルゴリズム 5，23–26 行）し，これらの端末からのクエリ応答の返信を待つ．

CH からのクエリメッセージを初めて受信した GW は，CH と同様に，親端末と経路リストを記録する（アルゴリズム 6，2–4 行）．GW は基本的にクエリメッセージを CH に転送するのみであるが，検索する全てのデータまで残り 1 ホップ以下であり，このクエリメッセージの転送先となる CH の候補（$Dest_{CH}$.candidate）が，そのデータの保持端末でない場合を把握できる（アルゴリズム 6，5–9 行）．これ

Algorithm 6: GW によるクエリ転送アルゴリズム

```
1  case 検索クエリの受信
2      if N_j receives for the first time from a CH then
3          N_j.parent ← q_i.L_route[|q_i.L_route| − 1]
4          N_j.L_route ← q_i.L_route
5          if ∀x ∈ [0, |q_i.List_hop| − 1], q_i.List_hop[x].hopCt ≤ 1 and
6          N_j.RankingTable[q_i.List_hop[x].rank−1].node ∉ N_j.List_CH then
7              Dest_CH.candidate ← ∅
8              if N_j is a data holder then
9                  Execute Algorithm 7
10         else
11             Dest_CH.candidate ← N_j.List_CH − N_j.parent
12             Set query timer
13     else if N_j has already received from a CH then
14         Dest_CH.candidate ← Dest_CH.candidate −q_i.Dest_CH
15         if q_i.L_route[|q_i.L_route| − 2] = N_j then
16             N_j.children ← N_j.children ∪ q_i.L_route[|q_i.L_route| − 1]
17         else
18             N_j.neighbor ← N_j.neighbor ∪ q_i.L_route[|q_i.L_route| − 1]
19 case 検索クエリの送信
20     if query timer expired then
21         if Dest_CH.candidate ≠ ∅ then
22             q_i.L_route ← N_j.L_route ∪ {N_j}
23             q_i.Dest_CH ← Dest_CH.candidate
24             Send q_i
25         else
26             if N_j is a data holder then
27                 Execute Algorithm 7
```

により，クエリメッセージの不要な転送を抑制できる．また，クエリメッセージを転送する必要がなく，上位 k 個に含まれるデータを保持している場合，親端末にクエリ応答を返信する（アルゴリズム 6，9 および 27 行）．

GW および CH のいずれでもない端末が CH からクエリメッセージを受信した場合，クエリメッセージの転送は行わないが，自身が k 位以内のデータを保持していれば，送信元の CH を親として記録する．

ここで，モバイルアドホックネットワークでは端末の移動により，端末間のリンク切断が生じる場合がある．CH とのリンク切断が生じた場合，クエリメッセージの転送に失敗してしまう可能性がある．そのため，CH とのリンク切断を検出した

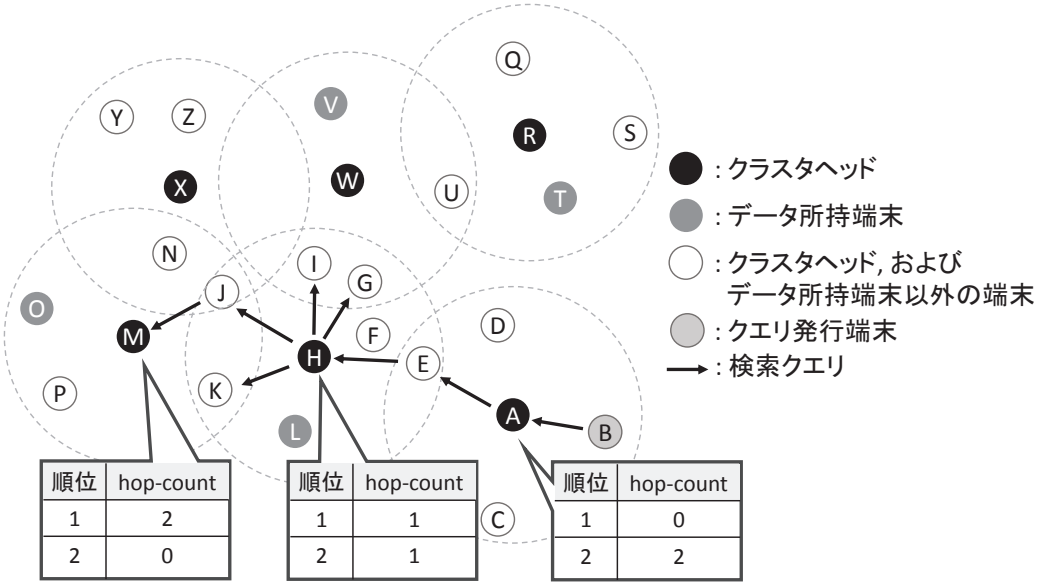


図 3.2: 検索クエリの転送例（$k = 2$）

端末は，クエリメッセージをブロードキャストする．このクエリメッセージに添付される $Dest_{CH}$ には，自身とのリンク切断が生じた CH の識別子が含まれる．このクエリメッセージを受信した端末は，$Dest_{CH}$ に含まれる CH のクラスタに属しており，その CH からのクエリメッセージを受信していない場合，その CH にクエリメッセージを送信する．また，リンク切断を検出した端末は，その CH を $List_{CH}$ から削除する．これにより，CH とのリンク切断が生じた場合においても，そのクラスタに属する端末がクエリメッセージを中継することにより，CH へクエリメッセージを転送できる．

図 3.2 を用いて，クエリメッセージの転送例を説明する．端末 B が $k = 2$ の Top-k 検索を行うものとし，図中の吹き出しは各 CH が保持する $List_{hop}$ を示す．B は自身が属するクラスタの CH である A を $Dest_{CH}$ としたクエリメッセージを送信し，これを受信した A はクエリメッセージをブロードキャストする．このクエリメッセージに含まれる $List_{hop}$ は（2 位，2 ホップ）で構成されている．これを受信した異なるクラスタにも属している E は，そのクラスタの CH である H を $Dest_{CH}$ としたクエリメッセージを送信する．このとき，A は，E の送信したクエリメッセージを傍受することにより，H を子 CH に追加する．H からのクエリメッセージを受信

したI，およびGはWが1ホップ先の2位のデータを保持している端末でないと判断できるため，クエリメッセージを転送しない．JはMが2位のデータを保持していることを把握できるため，Mにクエリメッセージを転送するが，Xには転送しない．また，Jのクエリメッセージを傍受したKも転送を停止する．以上の例により，CTRでは，上位k個のデータの取得に必要な端末のみによるTop-k検索を行えることがわかる．

クエリ応答の返信．クエリ応答は，クエリメッセージの転送経路を用いて返信する．さらに，リンク切断時には転送先を指定したクエリ応答をブロードキャストし，転送先を把握している端末が自律的にクエリ応答を転送する．

クエリ応答は，$r_i =< N_{src}, L_{data}, L_{CH}^{rep} >$で表される．$L_{data}$はデータ，およびそのデータの返信を開始したCHからのホップ数のリストであり，L_{CH}^{rep}は，これまでにクエリ応答を転送してきたCHの識別子のリストである．

クエリメッセージを送信する端末$N_{j'}$は，CHである端末，GWである端末，および，CHおよびGWのいずれでもないデータ所持端末である．クエリ応答の送信，および受信の処理は，それぞれアルゴリズム7，およびアルゴリズム8に従う．以下に，クエリ応答を送信する$N_{j'}$と，このクエリ応答を受信した端末の動作について説明する．

- CHおよびGWのいずれでもないデータ所持端末の場合
  順位表を参照し，自身が保持するk位以内のデータを添付したクエリ応答をクエリメッセージの送信元に送信する．

- CHまたはGWの場合

  1. 自身の子CHがいない端末$N_{j'}$は，クエリメッセージの送信元端末にクエリ応答を送信する．このとき，順位表を参照し，k位以内のデータをクエリ応答に添付する．$N_{j'}$がCHの場合，クラスタ内のデータ所持端末から受信したデータもクエリ応答に添付し，自身の識別子をL_{CH}^{rep}に追加する．
  2. 全ての子CHからクエリ応答を受信するか，クエリメッセージを受信してから一定時間経過した端末$N_{j'}$は，クエリ応答を作成し，クエリメッ

Algorithm 7: クエリ応答送信アルゴリズム

1 $r_i.L_{CH}^{rep} \leftarrow N_{j'}.L_{CH}^{rep}$ //クエリ応答を受信していない場合，$N_{j'}.L_{CH}^{rep} = \emptyset$
2 **if** $N_{j'}$ is a CH **then**
3 　$r_i.L_{CH}^{rep} \leftarrow r_i.L_{CH}^{rep} \cup \{N_{j'}\}$
4 $L_{data} \leftarrow \forall\{o, 0\}$ of its own data, where $score(o) \geq N_{j'}$.RankingTable[$k-1$].score
5 $r_i.L_{data} \leftarrow L_{data} \cup N_{j'}.L_{data}$ //クエリ応答を受信していない場合，$N_{j'}.L_{data} = \emptyset$
6 Send r_i to $N_{j'}$.parent

Algorithm 8: クエリ応答受信アルゴリズム

1 $N_{j'}.L_{CH}^{rep} \leftarrow N_{j'}.L_{CH}^{rep} \cup r_i.L_{CH}^{rep}$
2 Update each hop_count in $r_i.L_{data}$
3 $N_{j'}.L_{data} \leftarrow N_{j'}.L_{data} \cup r_i.L_{data}$
4 **if** $N_{j'}$ receives replies from its all children **or** predetermined time limit has elapsed **then**
5 　Execute Algorithm 7

セージの送信元端末にクエリ応答を送信する．このとき，クエリ応答には受信データ，および，自身が保持する k 位以内のデータを添付する．さらに $N_{j'}$ が CH の場合，L_{CH}^{rep} に自身の識別子を追加する．

以上の動作により，子 CH の存在しない端末からクエリ応答の返信を開始し，クエリ発行端末までクエリ応答を集約しながら返信できる．さらに，順位表を参照することにより，不要なデータの返信は生じない．また，受信したクエリ応答中の L_{data} から，データまでのホップ数を把握できるため，CH はこれを用いて自身の $List_{hop}$ を更新することができる

モバイルアドホックネットワークでは端末の移動に伴うリンク切断により，クエリメッセージの送信元端末にクエリ応答を返信できない場合がある．その場合，アルゴリズム 9 およびアルゴリズム 10 の手順に従ってクエリ応答を返信する．このときに送信されるクエリ応答の要素は，通常のクエリ応答の要素に加え，nextCH および nextGW を含む．nextCH は，このクエリ応答の転送先となる CH の識別子であり，nextGW は，このクエリ応答の転送先となる GW の識別子である．親端末とのリンク切断を検出した端末は，転送先となる端末の識別子（nextCH，および nextGW）を指定したクエリ応答をブロードキャストする．これを受信した端末は，指定された転送先の端末を把握している（例えば指定された CH のクラスタに属

Algorithm 9: クエリ応答のブロードキャスト

```
/* 端末 N_j がリンク切断を検出 */
case CH の場合
    if N_j.neighbor ≠ ∅ then
        N_j.parent ← ∃N_j' ∈ N_j.neighbor
        N_j.neighbor ← N_j.neighbor − N_j.parent
        Send r_i to parent
    else
        nextCH ← N_j.L_route[|N_j.L_route| − 2]
        nextGW ← original parent of N_j
        Broadcast r_i
case GW の場合
    nextCH ← N_j.parent
    nextGW ← ∅
    Broadcast r_i
```

Algorithm 10: ブロードキャストされたクエリ応答を受信

```
/* 端末 N_j' がブロードキャストされたクエリ応答を受信 */
if ∃r_i.nextCH ∈ N_j'.List_CH then
    Send r_i to r_i.nextCH
else
    if ∃r_i.nextGW ∈ N_j'.neighbor then
        Send r_i to r_i.nextGW
```

している）場合，その端末にクエリ応答を転送する．リンク切断が起きた場合のみ，クエリ応答をブロードキャストし，これを受信した端末が，指定された転送先の端末にクエリ応答を転送することにより，データの転送率を向上でき，検索精度を維持できる．

図 3.3 を用いて，リンク切断時のクエリ応答の返信例を説明する．図 3.2 の転送経路に従ってクエリメッセージを転送後，M-J 間でリンク切断が生じた場合を想定する．端末 M は子となる CH が存在しないため，クエリ応答の返信を開始する．M は 2 位以上のスコアをもつデータをクエリ応答に添付する．ここで，M は K を隣接端末として認識していないと仮定する（認識している場合，K にクエリ応答を返信する）．J とのリンク切断を検出した M は nextGW に J を，nextCH に H を格納したクエリ応答をブロードキャストする．これを受信した K は H のクラスタに属しているため，H にクエリ応答を転送する．H は子端末として J を記録せず，

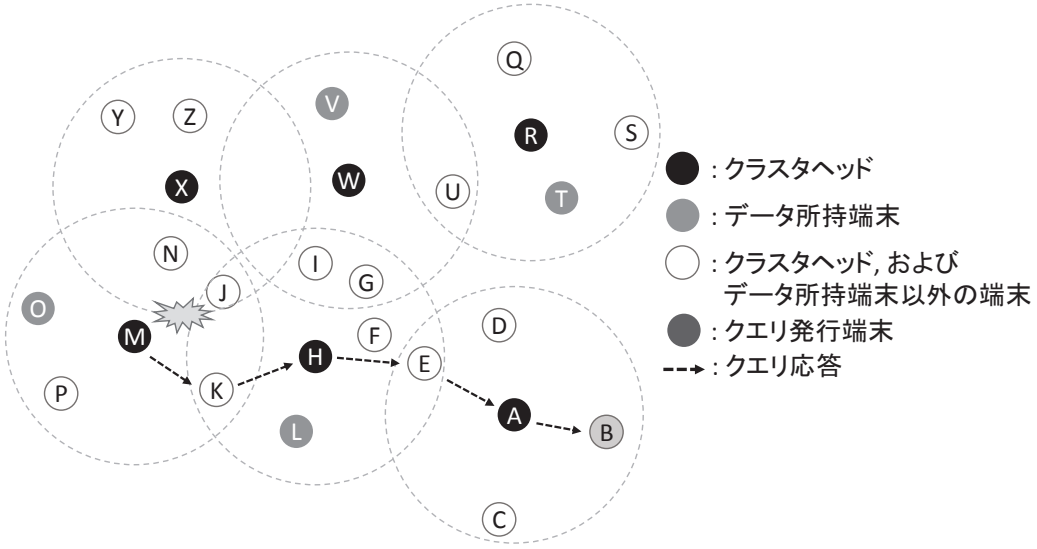


図 3.3: リンク切断時のクエリ応答の返信例

M を記録しているため，J からのクエリ応答を待たず，すぐにクエリ応答を送信できる．その後，クエリ応答は図 3.3 のように返信され，クエリ発行端末 B は，ネットワーク内の上位 2 個のデータを取得できる．

3.3.3　スコア更新への対応

CTR では，スコアが大きいデータを保持する端末ほど CH になりやすく，また，定期的にクラスタを再構築する．クラスタの構築では，ネットワーク内の端末間でデータの正確な順位を共有する必要があるため，スコア更新を全端末に通知する．単純に，スコアの更新をフラッディングを用いて全端末に送信することが考えられるが，メッセージの不要な転送が多く行われてしまい，実行中の Top-k 検索に干渉し，検索精度が低下することが考えられる．そこで CTR では，Top-k 検索におけるクエリメッセージと同様の方法でスコア更新メッセージを転送する．この際，$List_{hop}$ を用いず，CH はスコア更新メッセージをブロードキャストし，これを受信した端末は自身の順位表を修正する（CH は自身の $List_{hop}$ も更新する）．また，GW は，このメッセージを送信元以外の CH に転送する．これにより，メッセージの転送端末数を抑制しつつ，ネットワーク内の全ての端末がスコア更新メッセー

ジを受信できる．

3.4 性能評価

本節では，CTR 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す．本実験では，ネットワークシミュレータ Qualnet6.1[68] を用いた．

Top-k 検索手法として，CTR，CTR without $List_{hop}$，拡張手法 [4]，およびフラッディングによるデータ検索を行う単純手法を用いた．CTR without $List_{hop}$ は，基本的に CTR と同様のメッセージ処理を行うが，クエリメッセージ転送時に $List_{hop}$ を用いない．つまり，スコア更新メッセージと同様の手順でクエリメッセージが転送され，全ての CH がクエリメッセージをブロードキャストする．単純手法は，まず CTR と同様に順位表を作成する．クエリメッセージ，およびスコア更新メッセージはフラッディングを用いて転送し，親子関係を構築せず，クエリメッセージを受信した端末が k 位以内のデータを保持している場合，クエリメッセージの送信元端末にすぐにクエリ応答を送信する．各端末は，クエリ応答に初めて受信したデータが添付されている場合，クエリメッセージの送信元端末にクエリ応答を送信する．CTR において，クラスタリングの際に用いるパラメータ α および t は，予備実験から，基本値（表 3.1）における検索精度が最も高くなる値を用いた．また，拡張手法における，経路探索時の TTL を 2 とした．

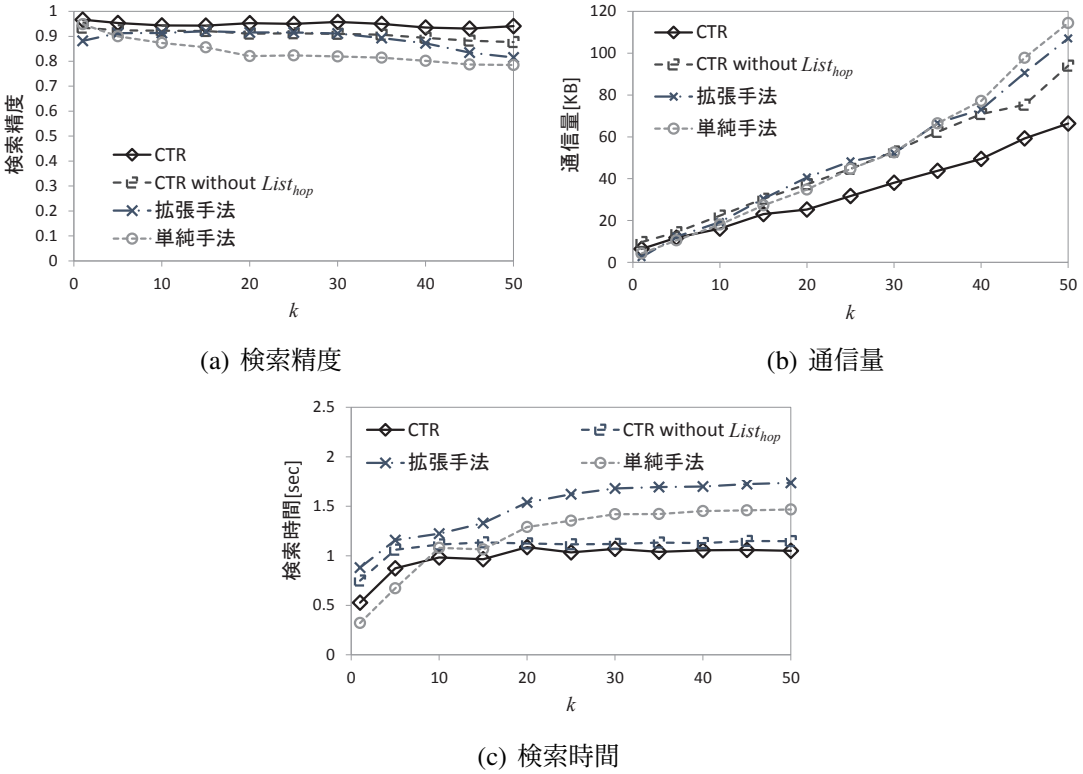
シミュレーション環境． 本実験におけるシミュレーション環境は，基本的に 2.6 節のものと同様であるが，端末数を n 台とし，I_q[sec] ごとにランダムに選ばれた端末が Top-k 検索を開始する．各端末の保持するデータのスコアは，300 秒から 600 秒の間で更新されるものとした．本実験におけるパラメータを表 3.1 に示す．評価指標は，2.6 節のものと同様である．

k の影響． 要求データ数 k を変化させた場合の各手法の性能を図 3.4 に示す．これらの図において，横軸は要求データ数 k を表し，縦軸は図 3.4(a) では検索精度，図 3.4(b) では通信量，および図 3.4(c) では検索時間を表す．

図 3.4(a) より，CTR は他の手法よりも高い検索精度を維持できていることが分かる．これは，クラスタリングによるネットワークトポロジ変化への対応，およ

表 3.1: パラメータ設定

パラメータ	値
k（要求データ数）	1, 5, 10, 15, 20, 25, **30**, 35, 40, 45, 50
v（端末の移動速度）	0, **0.5**, 1.0, 1.5, 2.0, 2.5, 3.0
I_q（クエリ発行周期）	2, **3**, 4, 5, 6, 7, 8, 9, 10
n（端末数）	**100**, 150, 200, 250, 300



(a) 検索精度

(b) 通信量

(c) 検索時間

図 3.4: 要求データ数 k の影響

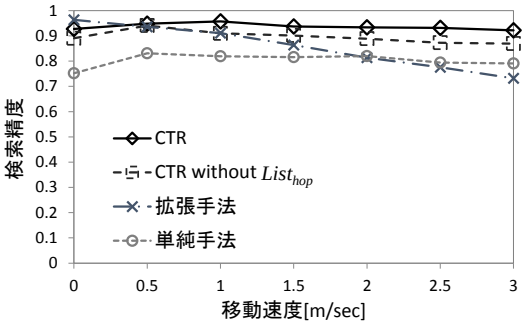
びリンク切断時におけるメッセージ処理により，クエリメッセージ，およびクエリ応答の転送率が他の手法よりも高いためである．特に k が大きい場合，拡張手法は検索精度が低下していることに対して，CTR，および CTR without $List_{hop}$ では検索精度の低下を抑止している．k が大きい場合，クエリ応答のメッセージサイズが大きくなり，パケットロスが発生しやすい．このとき，CTR，および CTR without $List_{hop}$ では，ブロードキャストを用いたクエリ応答の返信により，CH ま

でクエリ応答の返信を行うことで，上位 k 個に含まれるデータの転送率を向上している．単純手法は，クエリ応答を集約せずに，初めて受信したデータを返信する．そのため，クエリ応答の送信回数が増加し，パケットロスが発生しやすく，高い精度を維持できない．

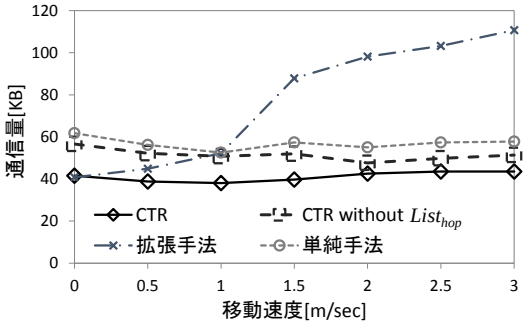
図 3.4(b) および図 3.4(c) より，CTR は小さい通信量，および短い検索時間で Top-k 検索を行っていることが分かる．CTR では，各 CH が $List_{hop}$ を用いて検索の不要なデータを判断することにより，クエリメッセージの転送範囲の拡大を抑制しているため，通信量を削減し，検索時間を抑制している．また，リンク切断の際も，経路探索を行わずにメッセージを転送できることから，遅延を抑制できる．CTR without $List_{hop}$ では，ネットワーク内の全ての CH にクエリメッセージが転送されるため，CTR よりも通信量，および検索時間が増加している．拡張手法は，k が大きい場合，検索時間が長い．これは，k が大きい場合，クエリメッセージの転送先端末が増加するため，リンク切断の機会が増加することから，経路探索を行う端末が増加し，クエリメッセージ，およびクエリ応答の転送遅延が増加するためである．単純手法は，親子関係を構築せず，k 位以内のデータはすぐにクエリメッセージの送信元端末に送信されるため，拡張手法よりも検索時間が短い．しかし，上述のように，クエリ応答の送信回数が増加するため，通信量が大きい．

ネットワークトポロジ変化の影響． 端末の移動速度 v およびクエリ発行間隔 I_q を変化させ，ネットワークトポロジの変化の影響を調べた．v および I_q を変えた場合の各手法の性能をそれぞれ図 3.5，および図 3.6 に示す．図 3.5 において，横軸は移動速度 v を表し，縦軸は図 3.5(a) では検索精度，図 3.5(b) では通信量，および図 3.5(c) では検索時間を表す．また，図 3.6 において，横軸はクエリ発行間隔 I_q を表し，縦軸は図 3.6(a) では検索精度，図 3.6(b) では通信量，および図 3.6(c) では検索時間を表す．

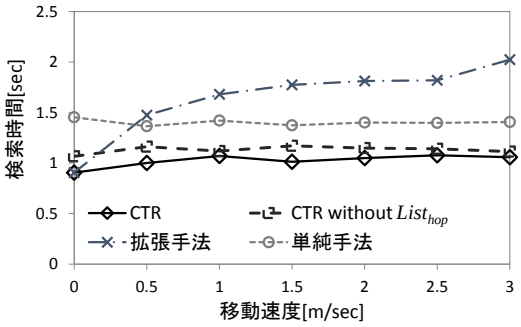
図 3.5(a) および図 3.6(a) から，CTR はネットワークトポロジ変化の影響をそれほど受けず，高い検索精度を維持できていることがわかる．CTR，および CTR without $List_{hop}$ では，CH が一定時間ごとに CHA を送信することにより，各端末は CH との無線リンクを確認することができ，定期的にクラスタを再構築している．そのため，拡張手法よりも，クエリメッセージ，およびクエリ応答の転送率が高い．さら



(a) 検索精度



(b) 通信量

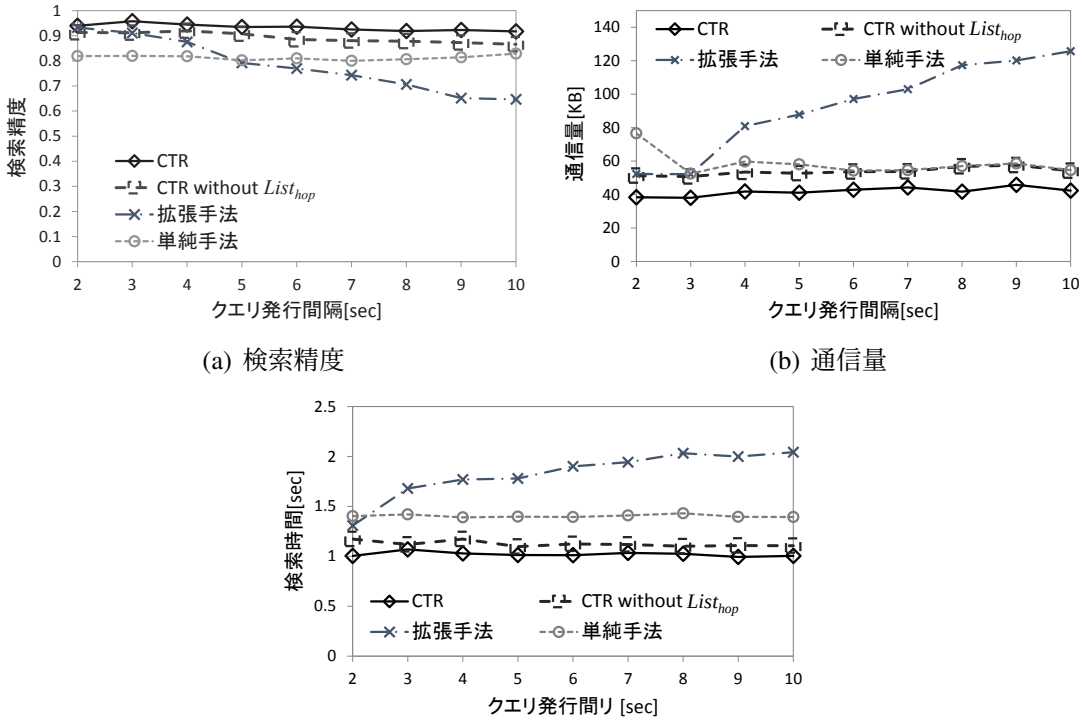


(c) 検索時間

図 3.5: 端末の移動速度 v の影響

に，リンク切断が起きた場合においても，リンク切断を検出した端末はメッセージをブロードキャストし，これを受信した端末が，受信メッセージの送信元端末とのリンク切断が起きた端末へ，自律的にそのメッセージを転送する．これにより，クエリメッセージ，およびクエリ応答のメッセージ転送失敗を防いでいる．拡張手法では，ネットワークトポロジ変化が小さい場合（v が小さい場合），リンク切断が起こる頻度が小さいため，安定したメッセージ転送を実現しており，検索精度が高い．しかし，ネットワークトポロジの変化が激しい場合（v が大きい場合），経路表における宛先端末，および Top-k 検索中の親端末とのリンク切断が頻繁に生じるため，経路探索に失敗する機会が増加し，検索精度が低下してしまう．同様に，I_q が大きい場合，経路表を更新する機会を失うため，経路表の精度を維持できず，検索精度が低下してしまう．

図 3.5(b)，図 3.5(c)，図 3.6(b)，および図 3.6(c) より，ネットワークトポロジの変



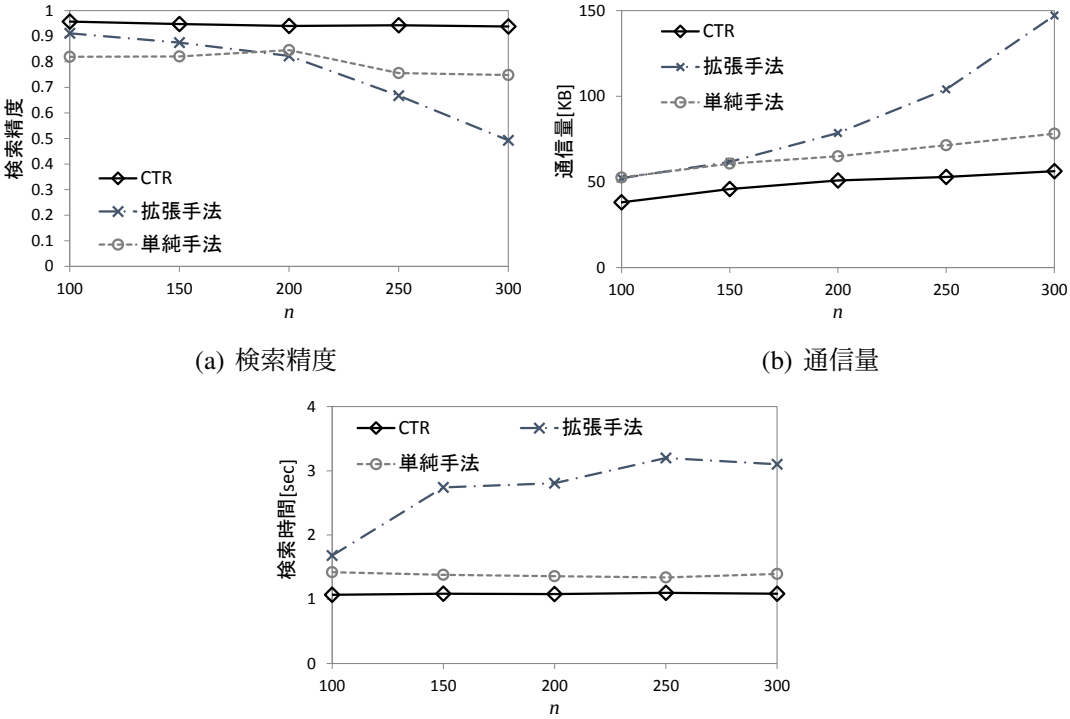
(a) 検索精度

(b) 通信量

(c) 検索時間

図 3.6: クエリ発行周期 I_q の影響

化が激しい場合においても，CTR は通信量が小さく，検索時間も短いことが分かる．これは，k の影響で述べた理由と同様である．CTR では，$List_{hop}$ によりクエリメッセージの不要な転送を抑制しているため，クエリメッセージの転送範囲の拡大を抑制し，CTR without $List_{hop}$ および拡張手法よりもネットワークトポロジ変化の影響を受けにくい．また，CTR，および CTR without $List_{hop}$ では，GW ではなく，CH を子端末として記録している．子である CH からのクエリ応答を受信できた場合，すぐにクエリ応答を返信するため，GW と CH 間のリンク切断の影響を受けにくく，遅延を抑制できる．拡張手法では，ネットワークトポロジの変化が激しい場合，多くの端末が経路探索を頻繁に実行するため，通信量および検索時間が増加する．また，リンク切断の影響により，子端末からクエリ応答を受信できない場合が多く，クエリメッセージの中継端末はクエリ応答をすぐに返信できず，遅延が増加してしまう．



(a) 検索精度

(b) 通信量

(c) 検索時間

図 3.7: 端末数 n の影響

n の影響． 端末数 n を変化させた場合の各手法の性能を図 3.7 に示す．これらの図において，横軸は端末数 n を表し，縦軸は図 3.7(a) では検索精度，図 3.7(b) では通信量，および図 3.7(c) では検索時間を表す．k の影響，およびネットワークトポロジ変化の影響から，CTR without $List_{hop}$ の性能は CTR 以下であることがわかるため，CTR without $List_{hop}$ の結果は割愛する．

図 3.7(a) より，CTR の検索精度は端末数に関わらず，安定して高い値であることがわかる．CTR では，検索に必要な端末のみにより Top-k 検索を行っているため，メッセージを送信する端末数はネットワーク全体の端末数に依らず，ほぼ一定である．そのため，検索時間もほぼ一定になる（図 3.7(c)）．一方，拡張手法の検索精度は，端末数が増加するとともに低下している．ネットワーク内の端末数が増加すると，上位 k 個のデータを取得するために必要な端末数の割合は低下する．拡張手法では，Top-k 検索中にのみ経路表を修正しているため，クエリメッセージ

を受信しない端末は自身の経路表を修正する機会を失い，その精度を保つことができない．また，端末数が多いとき，単純手法の検索精度も低下する．これは，クエリメッセージをフラッディングにより転送しているため，多くの端末がクエリ転送を行い，クエリ応答とのメッセージ衝突が頻繁に生じるためである．

図 3.7(b) より，CTR の通信量は，端末数が増加するとともにわずかに増加する．CTR では，CH とクエリメッセージを転送する GW とのリンク切断が生じた際，クラスタ内に存在する端末が受信したメッセージを CH に転送している．ネットワーク内の端末数が多い場合，クラスタに含まれる端末も増加するため，リンク切断の際に多くの端末がメッセージを転送し，通信量が増加する．拡張手法の通信量が増加するのは，多くの端末が自身の経路表を修正する機会を失うことに起因する．つまり，多くの端末が，自身がクエリメッセージを発行した際に経路探索メッセージを送信するため，通信量が増加し，検索時間が長くなる（図 3.7(c)）．単純手法の通信量が増加するのは，フラッディングによりクエリメッセージを転送しているためである．しかし，k 位以内のデータを保持する端末のみがクエリ応答の返信を開始することから，クエリ応答を返信する端末数は基本的に一定であり，検索時間が増加しない（図 3.7(c)）．

3.5 むすび

本章では，モバイルアドホックネットワークにおいて，上位 k 個のデータの取得に必要な端末のみによる Top-k 検索を実現する，クラスタを用いた Top-k 検索のためのルーティング手法として CTR を提案した．CTR では，高順位のデータを保持する端末をクラスタヘッドとするクラスタリングを行い，クラスタヘッド間でクエリルーティングを行うことにより，上位 k 個のデータを取得するために必要な端末のみで Top-k 検索を行う．また，クラスタヘッドによる定期的なメッセージ送信，およびクラスタの再構築によりネットワークトポロジの変化に対応し，ネットワークトポロジの変化が激しい場合でも高い検索精度を維持できる．リンク切断の際には，経路探索を用いず，クラスタ内の端末がメッセージをクラスタヘッドに中継することにより，転送遅延を抑制しつつ，メッセージの転送率を向上して

いる．シミュレーション実験の結果から，CTR はネットワークトポロジが頻繁に変化する環境や大規模なネットワーク環境においても高い検索精度を維持し，小さい通信量で高速に Top-k 検索を実行できることを確認した．

本章ではデータの複製を想定していないが，過去の Top-k 検索で取得したデータを複製することにより，クエリメッセージを送信することなく，または，クエリメッセージの送信範囲を抑制しつつ，上位 k 個のデータを取得できるものと考えられる．具体的には，CTR において，クラスタ内に存在する端末に高順位のデータの複製を配置することで，クエリメッセージの転送範囲の拡大を抑制でき，通信量および検索遅延をさらに抑制しつつ，高い検索精度を保証できるものと考えられる．このとき，データの更新が起きた場合においてもメンテナンスコストが小さくなるように複製を配置しなければならない．そのため，今後は，データの複製配置を考慮したメッセージ処理手法，およびメンテナンスコストが小さい複製配置手法について検討する予定である．

第4章　多次元データに対するTop-k検索のためのルーティング手法

4.1　まえがき

Top-k 検索では，与えられたスコアリング関数に基づいてデータをスコア付けし，それらのうち上位 k 個のものを検索する．アプリケーション，もしくはユーザ毎にスコアリング関数が異なる環境では，スコアリング関数によりデータのスコアが変化するため，k 位以内のデータ集合も異なる可能性がある．具体的には，データは複数の属性値をもっており，重要視される属性は，アプリケーションの目的やユーザの趣向などにより異なる．本章では，このような多次元データに対して，効率的に Top-k 検索を実行する問題に取り組む．アドホックネットワークでは，通信量の増加に伴う帯域の圧迫による，パケットロスや端末の消費電力の増大といった問題を防ぐため，Top-k 検索を行う際は，k 位以内に含まれないデータの返信の削減，およびメッセージ送信端末数の削減による通信量の抑制が重要である．そのため，上位 k 個のデータの取得に必要な端末のみで検索を行うことが望ましい．しかし，上位 k 個のデータはスコアリング関数に依存するため，検索に必要な端末もユーザの指定する条件によって異なる．

ここで，Top-k 検索を行う単純な方法として，次の二通りが考えられる．一つ目は，クエリメッセージをフラッディングにより送信し，全端末から上位 k 個のデータを取得するものである．しかし，このアプローチは無駄なデータの返信やクエリメッセージの転送が多く，通信量削減という目的を達成できない．もう一方は，検索条件ごとに効率的なメッセージの転送経路を作成するものである．しかし，Top-k 検索を行うユーザがそれぞれ異なる検索条件を指定する場合，それらの異なる条件の個々に対して経路を構築および管理する必要があり，そのオーバ

ヘッドは非常に大きい．また，ある検索条件において k 位以内となるデータを全ての端末に送信することにより，分散的な検索を避けるアプローチも考えられるが，送信すべきデータが非常に多い点，およびデータ更新が起きた際の処理で発生する通信量も大きくなる点で非効率である．そのため，検索条件ごとにクエリメッセージを転送すべき端末を動的に決定し，通信量を削減する方法が必要である．

そこで本章では，アドホックネットワークにおいて，ユーザごとに異なる検索条件が指定された場合においても，通信量を小さく，かつ短い検索時間で Top-k 検索を行う手法である ClusTo[1]を提案する．ClusTo は，スカイライン [17] を用いたクラスタリングにより，多くの条件で上位 k 個に含まれることが予想されるデータを持つ端末をクラスタヘッドとする．また，隣接端末の持つデータ集合の K-スカイバンド [59] を把握することにより，可能な限り，上位 k 個のデータの取得に必要な端末のみで Top-k 検索を行う．

以下では，4.2 節で想定環境を説明し，本章の問題を定義する．4.3 節で関連研究を紹介し，4.4 節で提案手法について述べる．4.5 節でシミュレーション実験の結果を示し，最後に 4.6 節で本章のまとめを行う．

4.2　想定環境と問題定義

4.2.1　想定環境

ネットワークモデル． ネットワーク内には，同等の性能をもつ n 台の端末（$N_1, N_2, ..., N_n$）が存在し，各々がデータ集合 O_i（$1 \leq i \leq n$）を保持している．各端末は何らかの無線通信方式を用いて，通信範囲内（c[m]）の端末と双方向で通信できるものとし，通信範囲内に存在する端末を隣接端末と呼ぶ．また，各端末は GPS などを用いて自身の位置を正確に把握できるものとする．さらに，本章では，固定端末による静的なネットワークを想定し，ネットワークトポロジの変化は起きないものとする．端末を持つユーザが移動するようなアプリケーションも考えられるが，ネットワークトポロジには影響を与えない程度の移動と考える．例えば，

[1] Cluster & K-skyband-based multi-dimensional Top-k query processing

協調作業を行うアプリケーションの一部では，ユーザは自分勝手な移動をすることなく，限られた範囲（担当領域など）を動くことが考えられる．

データモデル． ネットワーク内に存在するデータの集合を $O(=\cup O_i)$ とする．本章では，データ $o_i \in O$ は d 個の属性値をもつと想定し，これらの属性値は，ユークリッド空間上において多次元（d 次元）のポイントで表すことができる．つまり，$o_i.p =< p[1], p[2], ..., p[d] >$ であり，$p[j]$ は，d 次元空間における j 次元目の属性値である．また，各属性値は小さい方が好まれるものとし，[0, $value_{max}$] の範囲で正規化されているものとする．

4.2.2 問題定義

本章における Top-k クエリは，スコアリング関数 $f : \mathbb{R}^d \rightarrow \mathbb{R}$ を用いて，属性値からデータのスコアを計算する．$o_i \in O$ のスコアは，

$$score(o_i) = f(w, o_i) = \sum_{j=1}^{d} w_j \cdot o_i.p[j] \tag{4.1}$$

として計算される．また，$\sum_{j=1}^{d} w_j = 1$ である．Top-k 検索を行うユーザは，$w_j (1 \leq j \leq d)$ を入力とし，各次元に対する重要度を示す．例えば，図 4.1 は 2 次元（2 つの属性値をもつ）データの集合を示しており，d_1 次元に重みを置く場合，データ a が 1 位のデータとなり，逆に d_2 次元に重みを置く場合，h が 1 位のデータとなる．本章で想定するスコアリング関数は単調増加関数であり，以下の特徴をもつ．

$$\forall j \in [1, d], o.p[j] \leq o'.p[j] \Rightarrow f(o_i) \leq f(o_{i'}) \tag{4.2}$$

また，本章では検索範囲（qr[m]）を指定できるクエリを考える．ユーザは，k，w，および qr を指定し，自身から qr[m] 以内に存在する端末が保持するデータ集合において，スコアの最も小さい k 個のデータを検索する．

定義 2（Top-k クエリ）． k，w，および qr が与えられた時，Top-k クエリは，式 (4.1) からデータ $o_i \in O$ のスコアを計算する．また，クエリ発行端末から qr[m] 以

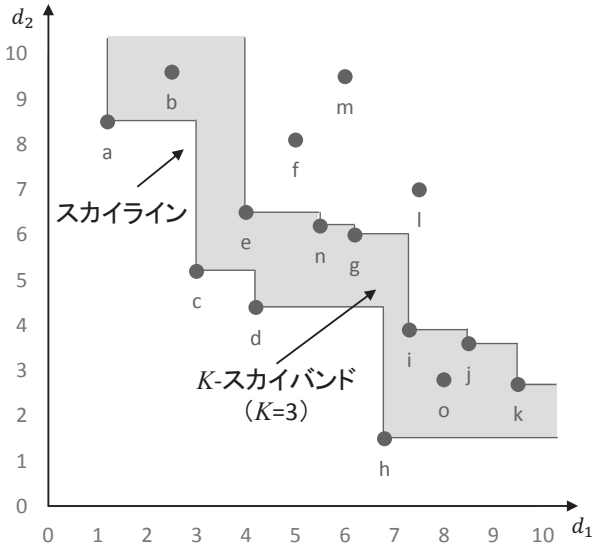


図 4.1: 2 次元データ集合におけるスカイラインおよび K-スカイバンドの例

内に存在する端末が保持するデータ集合を O' とする．この Top-k クエリは，O' においてスコアが最も小さい k 個のデータを検索する．つまり，正解集合を *TOPK* とし，$o_i \in$ *TOPK*，$o_j \in O' \backslash$ *TOPK* とすると，$score(o_i) \leq score(o_j)$ ある．

また，K を指定可能である最大の k とする．本章では，この Top-k クエリによるデータ検索が行われる環境において，通信量が小さく，かつ短い検索時間で上位 k 個のデータを取得する手法を提案する．

これまでに，単調増加関数をスコアリング関数とする Top-k クエリと密接に関係する概念が提案されており，ClusTo もこれらの概念を利用して Top-k 検索を行う．以下では，これらの概念を紹介する．

定義 3（支配）． あるデータ $o_i, o_{i'} \in O$ に対して，以下の条件をみたす場合，o_i は $o_{i'}$ を支配しているという．

$$(\forall j \in [1, d], o_i.p[j] \leq o_{i'}.p[j]) \wedge (\exists j \in [1, d], o_i.p[j] < o_{i'}.p[j]) \tag{4.3}$$

例えば，図 4.1 において，データ c は，データ e，f，g，l，m，および n を支配している．この支配という概念から，スカイライン [17] と呼ばれるデータ集合が導き出される．

定義 4（スカイライン）． あるデータ集合 O が与えられた時，スカイラインは O において，どのデータにも支配されていないデータの集合である．

例えば，図 4.1 におけるデータ集合において，スカイラインは {a, c, d, h} である．さらに，スカイラインを拡張した K-スカイバンド [59] を紹介する．

定義 5（K-スカイバンド）． あるデータ集合 O が与えられた時，K-スカイバンドは，最大 $K-1$ 個のデータに支配されているデータの集合である．

定義 5 からわかるように，スカイラインは $K=1$ の場合の K-スカイバンドである．この K-スカイバンドは，いかなる w に対しても上位 K 個のデータを含んでいるデータ集合である．

定理 1． スコアリング関数が単調増加関数である場合，上位 K 個のデータは，必ず K-スカイバンドに含まれる．

証明． K 位以内に含まれるあるデータ o_i が K-スカイバンドに含まれていないと仮定する．このとき，少なくとも K 個のデータが o_i を支配している．式 (4.2) および式 (4.3) から，少なくとも K 個のデータのスコアは，o_i のスコアよりも小さくなる．つまり o_i は K 位以内のデータとならず，これは仮定と矛盾する．したがって定理 1 は成り立つ． □

4.3 関連研究

単調増加関数をスコアリング関数とする Top-k 検索問題は，これまでに多くの研究で取り組まれており，集中管理データベースや分散ネットワークの分野において様々な手法が提案されている．本節では，これらの手法について紹介する．

4.3.1 集中管理データベースにおける多次元データに対する Top-k 検索

集中管理データベースの分野では，スコアリング関数とデータ集合が与えられたとき，Top-k となるデータを高速に検索するためのアルゴリズムが数多く提案さ

れている [13, 15, 22, 23, 28, 31, 35, 36, 37, 47, 74, 84, 91, 92]．これらのアルゴリズムは初期処理などにおいて，高速な検索を実現するためのデータ構造を作成する．このデータ構造は，レイヤ，リスト，およびビューの三つに分類することができる．

レイヤに基づくデータ構造は，凸包 [22, 23, 35] やスカイライン [47, 91, 92] を用いたインデックスであり，階層的に構築されている．あるデータは必ず 1 つのレイヤに属しており，上位 k 個のデータは k 番目までのレイヤに存在していることが保証されるという利点がある．リストに基づくデータ構造 [13, 15, 31, 37, 74] は，データの各属性値をそれぞれリストにより管理し，値の小さい（または大きい）順にソートしておく．これらのリストは，ラウンドロビンを用いたアプローチによりアクセスされ，あるデータは，そのスコアの下界値（または上界値）を計算されることにより，k 位以内のデータとなる可能性があるか判断される．ビュー [28, 36, 84] は，過去の Top-k 検索の結果をキャッシュしたものであり，新たに入力されたスコアリング関数に類似したスコアリング関数における Top-k 検索の結果を用いて検索時間を短縮する．

4.3.2 分散ネットワークにおける多次元 Top-k 検索

P2P ネットワークや無線センサネットワークにおいても，集中管理データベースで提案されたアイデアを利用し，効率的に Top-k 検索を実行するアルゴリズムが提案されている．文献 [20] では，上位 k 個に含まれないデータの返信を削減する TPUT と呼ばれるアルゴリズムが提案されている．文献 [54] では，TPUT を拡張した手法である KLEE が提案されている．これらの手法では，リストを用いて複数フェーズによる Top-k 検索を行う．複数フェーズによるアプローチは検索遅延が増加するため，高速に上位 k 個のデータを取得できない．また，これらの手法では，垂直分散型のデータベースを想定しており，水平分散型のデータベースを想定する本章と異なる．

文献 [62, 78, 79, 80, 89] では，水平分散型のデータベースを想定した手法が提案されている．文献 [62, 89] では，ビューのアイデアを用いており，過去の Top-k 検索で取得したデータをキャッシュし，新たな Top-k 検索で生じる通信量および遅延

を削減している．具体的には，検索クエリを発行した端末はまず，入力されたスコアリング関数における上位 k 個のデータを保持しているか確認し，上位 k 個の全てのデータを保持しているならば検索を終了する．上位 k 個のデータが揃っていない場合には，欠けているデータの検索要求を他の端末に行い，正確に Top-k 検索を行う．データをキャッシュするアプローチは，転送するデータの数を削減できるため，通信量の観点から有効であるが，データの更新が起きる場合，キャッシュが無効化される．そのため，キャッシュに依存したアルゴリズムは適用範囲が限定的であり，スケーラブルでない．文献 [78, 79, 80] では，スカイラインをクエリルーティングのためのインデックスに用いた手法を提案している．あるスコアリング関数が指定された場合，上位 k 個に含まれる可能性があるデータを保持している端末にのみクエリメッセージを送信し，通信量および検索遅延を削減している．しかし，任意の端末に 1 ホップでアクセスできる環境を想定しており，アドホックネットワークに直接適用することは困難である．

文献 [40, 41] では，無線センサネットワークにおける多次元データに対する Top-k 検索手法を提案している．この手法では，文献 [91, 92] で提案されたデータ構造を用いてフィルタを作成する．基地局端末は，収集したセンサデータから各端末に対応するフィルタを作成し，無駄なデータの送信を抑制している．しかし，この手法では，想定される全てのスコアリング関数を考慮しており，システムに入力されない条件に対しても検索が行われている．そのため，不要に通信量が大きくなるという問題が発生する．

提案手法の ClusTo は，レイヤに属するデータ構造である K-スカイバンドを利用して Top-k 検索を行う．さらに，新たなアプローチとして，スカイラインを利用したクラスタリングを提案する．これは，高順位となりやすいデータを持つ端末をクラスタヘッドにするものである．ClusTo では，k 番目のスコアである閾値を計算するため，クエリメッセージにフィルタを添付し，クラスタヘッド間でクエリルーティングを行うことにより，クエリメッセージの無駄な転送を抑制し，不要なデータの返信を削減する．

4.4 提案手法

本章では，ClusTo におけるクラスタリング方法，および Top-k 検索のためのメッセージ処理方法について説明する．

4.4.1 クラスタの構築

フラッディングは，上位 k 個に含まれるデータを持つ端末へクエリメッセージを送信するための単純なアプローチである [64]．しかし，フラッディングによる転送では，検索に必要がない多くの端末にクエリメッセージが送信されてしまい，無駄な通信量が発生してしまう．ClusTo は，クラスタを用いてクエリメッセージをルーティングすることにより，この問題を解決する．クラスタを用いたルーティングでは，他のクラスタに属する端末と隣接する端末（ゲートウェイノード）を介し，クラスタヘッド間でクエリメッセージの送信が行われる．このとき，クラスタヘッドの持つデータが，クラスタ内の端末の持つデータを支配している場合，これらの端末にクエリメッセージを送信することなく Top-k 検索を行える．そこで，より少ない端末数での Top-k 検索の実現を目的として，多くのデータを支配すると予想されるデータを持つ端末をクラスタヘッドとする．具体的には，各端末は自身の持つデータにおけるスカイラインを計算し，以下の式からクラスタヘッドとなるためのタイマを設定する．タイマは以下の式を用いて計算する．

$$Timer = \frac{\alpha \cdot SkyDist_{avg} + (1-\alpha) \cdot SkyDist_{min}}{\beta}. \tag{4.4}$$

$\alpha \in [0,1]$，および $\beta(>0)$ は，システムパラメータであり，β はタイマの値が不要に大きくなることを防ぐために用いる．ここで，S_i を O_i におけるスカイラインとする．また，$o \in S_i$ とすると，

$$SkyDist_{avg} = \frac{\sum_{|S_i|}(\sqrt{\sum_{j=1}^{m} o.p[j]^2})}{|S_i|}. \tag{4.5}$$

$$SkyDist_{min} = \min_{|S_i|}(\sqrt{\sum_{j=1}^{m} o.p[j]^2}). \tag{4.6}$$

Algorithm 11: クラスタ構築メッセージの処理

```
1  /* Expire the timer of node N_p */
2  cf ←< N_p, N_p, N_p.pos, KSKY_p >
3  Broadcast cf
4  /* Node N_q receives a cf from node N_p' */
5  if cf.N_CH ≠ N_q then
6      if cf.N_CH = cf.N_src then
7          Cancel its own timer
8          RoutingTable.entry ← RoutingTable.entry ∪{N_p', N_p'.pos, N_p'}
9          if N_q received cf for the first time then
10             cf ←< N_p', N_q, N_q.pos, KSKY_q >
11             Broadcast cf
12     else
13         RoutingTable.entry ← RoutingTable.enry ∪ {cf.N_CH, ∅, N_p'}
14 N_q.neighbor ← N_q.neighbor ∪ {N_p', N_p'.pos, KSKY_p'}
```

Algorithm 12: メンバ通知メッセージの処理

```
1  /* Procedure of non-cluster-head, N_q */
2  CHL ← {∀{N_p, N_p.pos} ∈ N_q.RoutingTable}
3  mn ←< N_q, CHL >
4  Send mn to the nearest cluster-head from N_q
5  /* Procedure of cluster head N_p which receives mn from N_q */
6  CM ← CM ∪ {N_q}
7  for ∀ mn.CHL.entry do
8      if ∃i, j, RoutingTable.entry_i.CH = mn.CHL.entry_j.CH then
9          RoutingTable.entry_i.CH_position ← mn.CHL.entry_j.position
10         if Dist(RoutingTable.entry_i.CH, N_q) < Dist(RoutingTable.entry_i.CH,
           RoutingTable.entry_i.next_node) then
11             RoutingTable.entry_i.next_node ← N_q
12     else
13         RoutingTable.entry ← RoutingTable.entry ∪ {mn.CHL.entry, N_q}
```

であり，$SkyDist_{avg}$ および $SkyDist_{min}$ は，端末 N_i が保持するデータ集合におけるスカイラインに含まれるデータ（のポイント）と原点 (0, 0, ..., 0) との平均距離，および最短距離をそれぞれ示す．定理 1 から，スカイラインは何らかの検索条件に対して最上位となるデータの集合であるため，これらの値が小さいほど，多くの検索条件において上位 k 個に含まれやすいデータを保持していることを示す．

ClusTo のクラスタリングは，アルゴリズム 11 およびアルゴリズム 12 に従って行われる．タイマが発火した端末 N_p はクラスタヘッドとなり，クラスタ構築メッ

セージ（*cf*）をブロードキャストする．*cf* は，$cf =< N_{CH}, N_{src}, N_{src}.pos, KSKY_{src} >$ で表される．N_{CH} は，*cf* を発行したクラスタヘッドの識別子であり，端末 N_p が発行した場合，$N_{CH} = N_p$ である．N_{src} は，*cf* を送信する端末の識別子であり，$N_{src}.pos$ は，*cf* を送信する端末の位置情報である．$KSKY_{src}$ は，O_{src} における K-スカイバンドに含まれるデータのポイントの集合である．クラスタヘッドからクラスタ構築メッセージを受信した端末はタイマをキャンセルし，以下の処理を行う．(i) 経路表を作成する（アルゴリズム 11，8 行および 13 行）．経路表のそれぞれのエントリにはクラスタヘッドの識別子（CH），その位置情報（CH_position），およびそのクラスタヘッドにクエリメッセージを送信するための隣接端末（next_node）の識別子[2]が格納される（図 4.2）．さらに，(ii) 隣接端末の位置情報，および *KSKY* を記録する（アルゴリズム 11，14 行）．データそのものではなく，K-スカイバンドに含まれるデータのポイントを送信するのは，データのスコアを計算するためには，これらの情報のみ把握すればよく，データそのものを送信すると通信量が大きくなるためである．定理 1 から，上位 $k(\leq K)$ 個のデータは，検索範囲に存在する端末が保持する K-スカイバンド内に存在するため，隣接端末の K-スカイバンドを把握することにより，いかなる検索条件が指定された場合においても，効果的なフィルタを作成できる（4.4.2 項）．さらに，これまでにクラスタ構築メッセージを送信していない場合，(iii) クラスタ構築メッセージをブロードキャストする．このとき，クラスタヘッドの識別子は，受信したメッセージのクラスタヘッドの識別子とする（アルゴリズム 11，9–11 行）．

一定時間経過後，クラスタヘッド以外の端末は，自身が把握する最短距離のクラスタヘッドに，メンバ通知メッセージ（*mn*）を送信する．*mn* は，$mn =< N_{src}, CHL >$ で表され，N_{src} は，*mn* を送信する端末の識別子である．また，*CHL* は，クラスタヘッドの識別子および位置情報のリストである．これにより，各クラスタヘッドは，隣接するクラスタのクラスタヘッド，next_node，および自身のクラスタに所属するメンバ端末（*CM*）を把握できる．このとき，next_node は送信先となるクラスタヘッドに最も近い端末とする（アルゴリズム 12，10–11 行）．これは，送信先となるクラスタヘッドに最も近い端末は，そのクラスタヘッドと隣接している

[2]クラスタヘッドと隣接している場合，next_node=CH となる

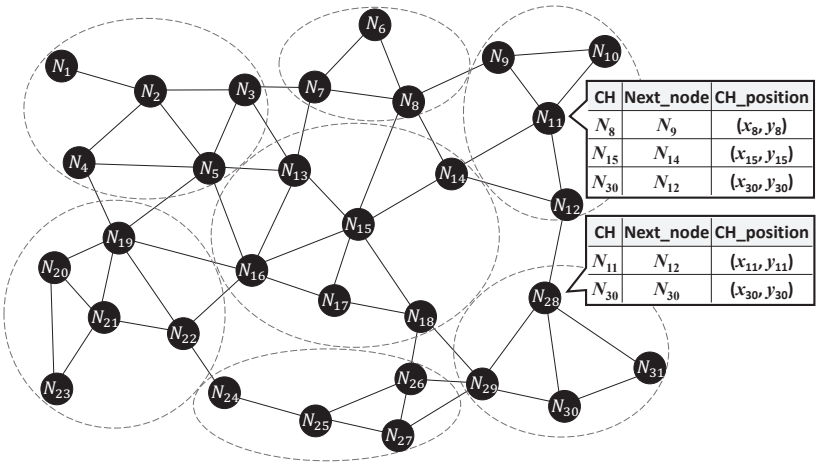


図 4.2: クラスタと経路表の例

可能性が高いため，送信先クラスタのゲートウェイノードにクエリメッセージを転送する必要がなく，クエリメッセージのホップ数を削減できるためである．

以上の動作により，効率的な Top-k 検索のためのクラスタを構築し，クエリメッセージを送信する端末数を抑制する．また，各端末は，自身の隣接端末の位置情報，および K-スカイバンドを把握できる．さらに，データの更新が起きた場合においても，各端末は隣接端末の情報のみを保持しているため，更新された *KSKY* を隣接端末にブロードキャストするだけでよい．

4.4.2 フィルタの作成

ClusTo では，経路表を用いてクエリルーティングを行い，ユーザが指定する検索範囲までクエリメッセージを転送する．その後，各クラスタにおいてクエリメッセージが送信されていない端末にクエリメッセージを送信し，上位 k 個に含まれるデータを取得する．通信量を削減するためには，検索に必要ない端末へはクエリメッセージを送信すべきではない．そのために，各端末が，検索範囲内の上位 k 番目のデータのスコア（閾値）を見積もることにより，k 位以内に含まれるデータを持たない隣接端末への不要なクエリメッセージの転送を削減することが有効である．これを実現する直感的な手法として，各端末がクエリメッセージにフィルタとして，自身が把握する上位 k 個のデータのスコアとその識別子を添付し，それ

を受信した端末が，この情報と自身が持つデータのスコアから，閾値を見積もることが考えられる．しかし，k が大きい場合，フィルタの送信にかかるオーバヘッドが大きくなる問題がある．そこで，ClusTo では，フィルタによるオーバヘッドを固定かつ小さく抑えるため，k 個分のデータではなく，各端末が把握する，上位 1 番目，$\lfloor k/2 \rfloor + 1$ 番目，および k 番目となる 3 つのデータのスコアとその識別子の組合せ（それぞれ lb，*me*，および *ub*）をフィルタとする．さらに，閾値を正確な値に近づけるため，各端末は，受信したクエリメッセージに含まれるフィルタ，および自身と自身の隣接端末が持つ *KSKY* からフィルタを更新する．

フィルタ作成のアルゴリズムを，アルゴリズム 13 に示す．各端末は，基本的に，受信したフィルタと自身と自身の隣接端末が持つデータにおける上位 k 個のデータのスコア情報から，フィルタを作成する．ここで，クエリメッセージに添付するフィルタは，3 つのスコアと識別子の組合せであるため，$k > 3$ の場合，クエリ発行端末からクエリの送信元端末までの経路に含まれる端末，およびそれらの隣接端末（以後，上流端末）の持つデータにおいて，上位 k 個となったデータのスコアと識別子の全ては把握できない．このとき，何らかの方法により，把握できなかったデータのスコアを推定する必要があるが，閾値が正確な値よりも小さくなってしまうと，検索範囲内の k 位以内のデータを持つ端末にクエリメッセージを送信しない可能性がある．一方，k 位以内のデータを持つ端末へ確実にクエリメッセージを転送するために，把握できなかったデータのスコアを大きめ（セーフティ）に推定しすぎると，クエリメッセージの不要な転送，および上位 k 個に含まれないデータの返信が増加してしまう．そのため，k 位以内のデータを持つ端末へのクエリメッセージの転送を保証しつつ，閾値が不要に大きくならないようにすることが重要である．

そこで，ClusTo では，上流端末の持つデータにおける上位 k 個のデータのうち，フィルタとして添付されなかったデータ[3]のスコア（および識別子）を，各端末が把握できるデータのスコアを基に推定する．具体的には，各端末はまず，クエリ送信元端末と自身との共通の通信範囲内に存在する端末の持つデータのスコアを計算する．このスコアと受信したフィルタから，クエリ送信元端末が把握してい

[3]クエリ送信元端末において，lb から me，および me から ub の間には，それぞれ $(\lfloor k/2 \rfloor - 1)$ 個，および $(k - \lfloor k/2 \rfloor - 2)$ 個のデータが存在する．

Algorithm 13: フィルタの計算

Input: lb, me, ub
Output: lb', me', ub'

1 /* Node, N_p, constructs a filter from a query sent by N_q */
2 $lm_data \leftarrow mu_data \leftarrow lm_data_{com} \leftarrow mu_data_{com} \leftarrow \emptyset$
3 **for** $\forall N_{q'} \in N_p$.neighbor **do**
4 **if** $N_{q'}$ is within the search range **then**
5 **if** $Dist(N_q, N_{q'}) \leq c$ **then**
6 $lm_data_{com} \leftarrow lm_data_{com} \cup \{\{\forall score(o_j), j\} \mid o_j.p \in KSKY_{q'}, score(o_j) \in [lb.score, me.score]\}$
7 $mu_data_{com} \leftarrow mu_data_{com} \cup \{\{\forall score(o_j), j\} \mid o_j.p \in KSKY_{q'}, score(o_j) \in (me.score, ub.score]\}$
8 Sort lm_data_{com} and mu_data_{com} by ascending order of score
9 $lm_data \leftarrow lb \cup lm_data_{com}$
10 **if** $\mid lm_data \mid < \lfloor k/2 \rfloor + 1$ **then**
11 **for** $i = 0$ to $\lfloor k/2 \rfloor + 1 - \mid lm_data \mid$ **do**
12 $lm_data \leftarrow lm_data \cup \{me.score, -(i+1)\}$
13 $mu_data \leftarrow mu_data_{com}$
14 **if** $\mid mu_data \mid < k - \lfloor k/2 \rfloor - 1$ **then**
15 **for** $i = 0$ to $k - \lfloor k/2 \rfloor - 1 - \mid mu_data \mid$ **do**
16 $mu_data \leftarrow mu_data \cup \{ub.score, -(i+k)\}$
17 $LTOPK \leftarrow$ top-k data in its own dataset and K-skyband of neighbors existing within the search range
18 $LTOPK \leftarrow LTOPK \cup lm_data \cup me \cup mu_data \cup ub$
19 Sort $LTOPK$ by ascending order of score
20 $lb' \leftarrow LTOPK[0]$
21 $me' \leftarrow LTOPK[\lfloor k/2 \rfloor]$
22 $j \leftarrow \lfloor k/2 \rfloor$
23 **while** $me'.id < 0$ **do**
24 $j \leftarrow j + 1$
25 $me' \leftarrow LTOPK[j]$
26 $ub' \leftarrow LTOPK[k-1]$
27 $j \leftarrow k - 1$
28 **while** $ub'.id < 0$ **do**
29 $j \leftarrow j + 1$
30 $ub' \leftarrow LTOPK[j]$

る上位 k 個のデータを推定する（3–8 行）．また，共通の通信範囲内の端末が持つデータのうち，スコアがフィルタの範囲内となるものの個数が，lb から me，および me から ub に存在するデータの個数に満たなかった場合，不足分を，[lb, me] 間は me，[me, ub] 間は ub と見なす（10–12 行，および 14–16 行）．このように，把握していないデータのスコアを [lb, me] 間，および [me，ub] 間における最大スコ

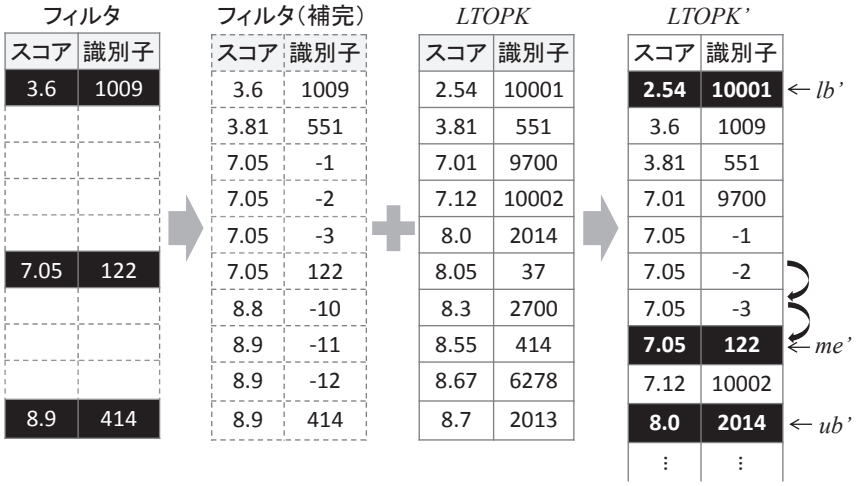


図 4.3: フィルタ構築の例（$k = 10$）

アとして補うことにより，クエリ送信元端末が把握している上位 k 個のデータのスコアを小さく見積もることはない．その結果，k 位以内のデータを持つ端末へのクエリメッセージの転送を保証できる．その後，受信したフィルタ，補完した { スコア，識別子 }，および自身と隣接端末が持つデータから上位 k 個となるものを計算し，自身のフィルタを作成する（17–30 行）．

図 4.3 を用いて，$k = 10$ の場合におけるフィルタ構築の例を説明する．ある端末 N_p が，図の左のフィルタが添付されたクエリメッセージを，端末 N_q から受信したとする．これらの端末の共通の隣接端末として，$N_{p'}$ が存在し，$o_{122}, o_{551} \in O_{p'}$ であり，$f(o_{122}) = 7.05$，$f(o_{551}) = 3.81$ とする．N_p は，$KSKY_{p'}$ により，これらの情報を取得でき，me を $\{7.05, 122\}$，ub を $\{8.9, 414\}$ とできる．また，自身が把握するデータ集合における上位 k 個のデータ（*LTOPK*）を計算し，フィルタとして lb'，me'，および ub' を得る．

4.4.3　Top-k 検索

本項では，ClusTo における Top-k 検索のメッセージ処理方法について説明する．

ClusTo では，2 種類の検索クエリ，クラスタ間クエリ（*CtoC query*）およびデータ収集クエリ（*data acquisition query*）を用いる．クラスタ間クエリにより，ユー

Algorithm 14: 検索クエリ q_i の処理

```
1  if N_p receives q_i for the first time then
2      N_p.parent ← q_i.N_src
3      Set F by Algorithm 13
4      N_p.ρ ← q_i.F[2].score //ub.score
5      if qc_i then
6          CL ← q_i.CL
7          for ∀ q_i.AT where q_i.AT.next_node ≠ N_p do
8              CL ← CL ∪ q_i.AT.CH
9          AT ← ∅
10         for ∀t ← RoutingTable.entry do
11             if t.CH is not included in CL and t.next_node is within qr from N_org then
12                 AT ← AT ∪ {t.CH, t.next_node}
13         if AT ≠ ∅ then
14             qc_i ←< N_org, N_org.pos, N_p, k, qr, w, F, AT, CL >
15             Send qc_i to nodes included in AT as next_node
16         else
17             N_p.LeafFlag ← 1
18             Execute Algorithm 15
19     if qd_i then
20         Reply data with scores not more than the threshold to N_p.parent
```

ザが指定した検索範囲を保証し，その後，データ収集クエリにより，上位 k 個に含まれる可能性があるデータを取得する．

まず，クラスタ間クエリ qc_i は，$qc_i =< N_{org}, N_{org}.pos, N_{src}, k, qr, w, F, AT, CL >$ で表される．i は検索クエリの識別子，N_{org} はクエリ発行端末の識別子，$N_{org}.pos$ は，クエリ発行端末の位置情報，および N_{src} はクエリ送信端末の識別子である．クエリ発行端末 N_{org} は，要求データ数 k，検索距離 qr，および $w =< w[1], w[2], \cdots, w[d] >$ を指定する．また，データのスコアを計算し，フィルタ F を作成する．その後，自身の経路表における全てのクラスタヘッドの識別子とその next_node のペアを，*AT*(=*AddressTuple*) に格納する．このとき，next_node が検索範囲内に含まれない場合は，*AT* に格納しない．また，*CL*(=*ClusterList*) は，クエリメッセージが送信されたクラスタヘッドの識別子を格納するリストであり，N_{org} が送信するクエリメッセージでは $\emptyset$ である（N_{org} がクラスタヘッドならば，*CL*= $\{N_{org}\}$）．クエリメッセージを作成後，N_{org} は，*AT* に含まれる next_node にクエリメッセージを送信する．これを受信した端末 N_p におけるメッセージ処理をアルゴリズム 14，およびア

Algorithm 15: 葉端末，およびクラスタヘッド N_p の処理

```
1  A ← ∅
2  if N_p.LeafFlag = 1 then
3      Put in ∀N_j, which are neighors and satisfy all the following conditions, to A
4      ·N_j is within qr from N_org
5      ·N_j has data with scores not more than N_p.ρ
6      ·N_j's cluster head is not within qr from N_org
7  if N_p is a cluster head then
8      Put in ∀N_j, which are cluster member and satisfy all the following conditions, to A
9      ·N_j is within qr from N_org
10     ·N_j has data with score not more than N_p.ρ
11     ·N_j ≠ N_p.parent
12 if A ≠ ∅ then
13     qd_i ←< N_org, N_org.pos, N_p, k, qr, w, F, A >
14     Send qd_i to nodes included in A
15 else
16     Reply data with scores not larger than N_p.ρ to N_p.parent
```

ルゴリズム 15 に示す．また，クエリメッセージを送信後，クエリ応答を受信した場合の処理をアルゴリズム 16 に示す．N_p は，すでにクエリメッセージが送信されているクラスタヘッドを除いたクラスタヘッドにクエリメッセージを送信する（アルゴリズム 14，6–12 行）．このとき，クラスタヘッドに送信するための next_node が検索範囲外であるならば，そのクラスタヘッドにはクエリメッセージを送信しない（アルゴリズム 14，9–12 行）．また，*AT* が $\emptyset$ となった場合は自身を葉端末とし，クラスタ間クエリの転送を中止する[4]．葉端末は，閾値以下のスコアをもつデータを保持している端末にデータ収集クエリを送信する（アルゴリズム 15）．

データ収集クエリ qd_i は，$qd_i =< N_{org}, N_{org}.pos, N_{src}, k, qr, w, \boldsymbol{F}, \boldsymbol{A} >$ で表される．*A*(=*Address*) は，データ収集クエリの送信先端末の識別子を格納するリストである．自身の *AT* に含まれる全ての next_node からクエリ応答を受信したクラスタヘッドも同様にデータ収集クエリを送信する．クエリメッセージを受信した各端末は，フィルタから閾値を設定する．さらに，葉端末までクエリメッセージを転送後，クエリ応答により閾値を更新し（アルゴリズム 16，2–3 行），その後，データ収集クエリを送信する．これにより，可能な限り閾値を小さくし，上位 k 個のデー

[4]クラスタ間クエリの転送経路は，クエリ発行端末を根とする木構造として表すことができるため，転送を中止する端末は葉と見なせる．

Algorithm 16: クエリ応答 r_i の処理

```
1  /* N_p receives r_i */
2  if N_p.ρ > r_i.ρ then
3      N_p.ρ ← r_i.ρ
4  N_p.L_data ← N_p.L_data ∪ r_i.L_data
5  if N_p receives r_i from all next_nodes inlcluded in AT for the fist time then
6      if N_p is a cluster head then
7          Execute Algorithm 15
8      else
9          L_data ← {∀o ∈ N_p.L_data | score(o) ≤ N_p.ρ}
10         r_i ←< N_p, L_data, N_p.ρ >
11         Send r_i to N_p.parent
12 else
13     if N_p receives r_i from all next_nodes inlcluded in A then
14         L_data ← {∀o ∈ N_p.L_data | score(o) ≤ N_p.ρ}
15         r_i ←< N_p, L_data, N_p.ρ >
16         Send r_i to N_p.parent
```

タの取得に不要な端末へのクエリメッセージ送信を抑制できる．ここで，検索範囲の境界付近では，クラスタヘッドは検索範囲外であるが，そのクラスタに所属する端末は検索範囲内に存在する場合がある．その端末は，クラスタヘッドによるデータ収集クエリの対象とならないため，上位 k 個に含まれるデータを持っている場合でも，クラスタヘッドからデータ収集クエリが送信されない．そのため，葉端末が上記の条件を満たす端末にデータ収集クエリを送信する必要がある．このとき，葉端末におけるデータ収集クエリを送信する条件の 3 つ目（アルゴリズム 15，6 行）により，検索範囲内の端末へのクエリメッセージの送信を保証する．

クエリ応答 r_i は，$r_i =< N_{src}, L_{data}, \rho >$ で表される．L_{data} には，閾値 ρ 以下のスコアをもつデータが含まれる．データ収集クエリを送信する必要のない葉端末（アルゴリズム 15，14–16 行），およびデータ収集クエリを受信した端末は，閾値以下のスコアをもつデータを親端末に返信する．

クエリメッセージ，およびクエリ応答を受信した端末は，受信確認のために ACK を送信元端末に返信する．クエリメッセージ，またはクエリ応答の送信端末は，一定時間内に ACK を受信できなかった場合，再送を行う．

以上の動作により，ClusTo は，いかなる検索条件が指定された場合においても，クエリメッセージを送信する端末数を抑制しつつ，k 位以内のデータを取得でき

る．また，ClusTo では，上位に含まれることが多いと考えられるデータを持つ端末をクラスタヘッドとしている．各端末は，クラスタヘッドにクエリメッセージを送信していくため，クラスタヘッドがクエリメッセージを受信した際，フィルタを更新し，閾値を小さく設定できるという利点がある．これにより，データ収集クエリの送信先となる端末数を抑制でき，通信量を削減できる．

4.5 性能評価

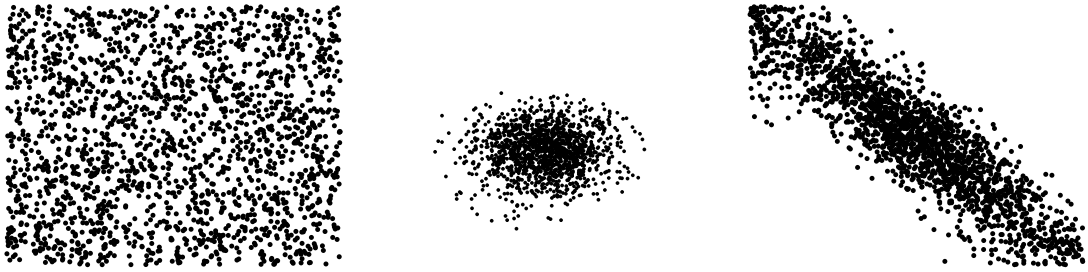
本節では，提案手法の性能評価のために行ったシミュレーション実験の結果を示す．本実験は，ネットワークシミュレータ Qualnet6.1[68] を用いた．

データセット． 本実験は，実データおよび人工データを用いた．

- 実データ
  本実験では，Intel Berkeley Research Lab[5]が無線センサネットワークにおいて収集（2004 年 2 月 29 日から 3 月 6 日）したデータを用いた．このデータ集合は，54 台のセンサ端末が観測したセンサ値により構成されている．また，欠損データやエラー値も存在するため，“temperature”，“humidity”，および “light” のセンサ値が完全に揃っているデータ集合を用いた．これらのセンサ値を [0, 250] の範囲で正規化し，属性値のバイトサイズおよびデータのバイトサイズを，それぞれ 4[bytes] および 128[bytes] と想定した．

- 人工データ
  実データに加え，3 つの人工データを用いた．人工データは，d 次元のデータ集合であり，その分布は，一様，クラスタ，および負相関に従う．一様分布に従うデータ集合におけるデータの各属性は，[0, 250] の範囲でランダムな値を取る．クラスタ分布に従うデータ集合では，平均を 125，分散を 30 とする正規分布によりデータの各属性値を決定する．負相関分布に従うデータは，ある属性値が小さい場合，他の属性値は大きいという特徴を持つ．図 4.4

[5]http://db.csail.mit.edu/labdata/labdata.html



(a) 一様 (b) クラスタ (c) 負相関

図 4.4: データ分布（$d = 2$）

は，2 次元のデータ集合の例を示している．また，データの属性を 4[bytes] とし，データを s[bytes] とした．

比較手法． ClusTo の比較手法として，ClusTo(OPT)，2 フェーズ検索手法 [64]，および単純手法を用いた．ClustTo(OPT) は，検索範囲内のデータ集合における k 番目に小さいスコアを添付したクエリメッセージを送信する．そのため，上位 k 個に含まれないデータが返信されることはない．なお，この手法を実現することは不可能であるが，この手法との比較により ClusTo の性能が最適なものにどれだけ近いかを確認できる．2 フェーズ検索手法は，2 章で紹介したものである．単純手法では，各端末は，自身が把握するデータにおける上位 k 個のもののスコアを添付したクエリメッセージをブロードキャストする．データを返信する際は，ClusTo と同様に閾値を設定し，閾値以下のスコアをもつデータのみを返信する．この手法は，文献 [33] で提案されている手法に対して，k 個の基準値を用いた場合と同等のものである．

シミュレーション環境． 800[m]×800[m] の 2 次元平面状の領域に n 台の端末をランダムに配置する．各端末は，それぞれ 50 個のデータを保持しており，IEEE802.11b を使用し，伝送速度 11[Mbps]，通信伝搬距離が 100[m] 程度となる送信電力でデータを送信する．データは時間経過により更新し，シミュレーション中に x[%] のデータが更新する．

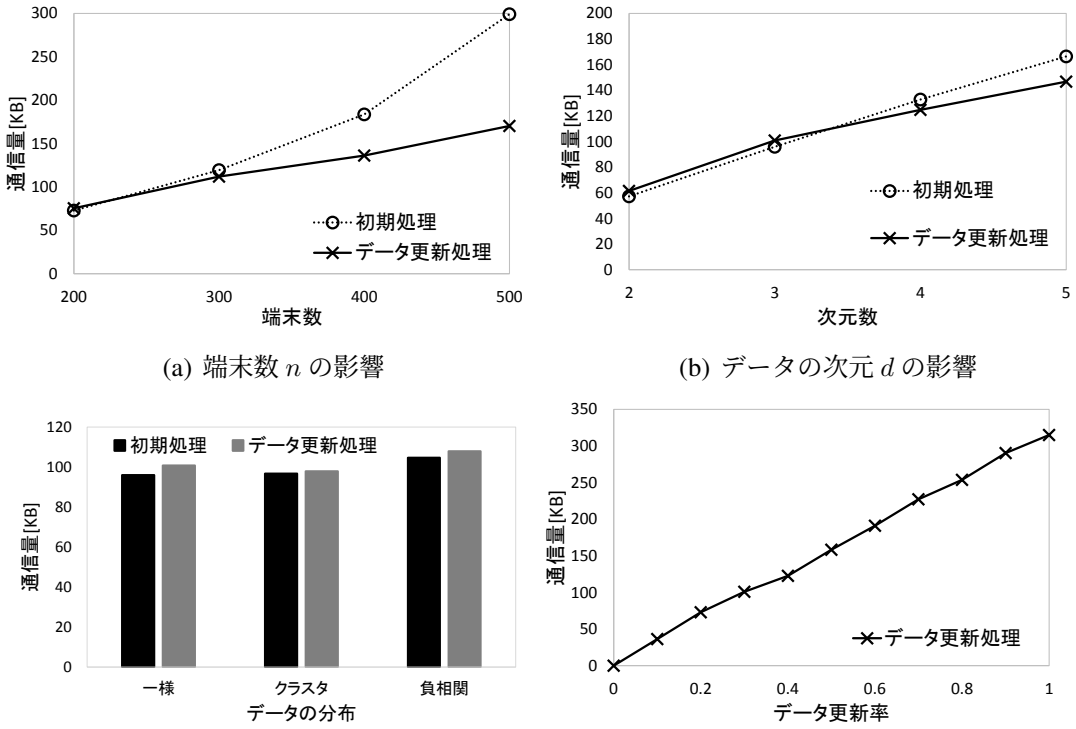
本実験では，$K = 15$ とし，クエリ応答の再送回数を 10 とした．これにより，限

表 4.1: パラメータ設定

パラメータ	値
n（端末数）	200, **300**, 400, 500
k（要求データ数）	1, 3, 5, 7, 10, 15
s（データサイズ）	32, 64, **128**, 256
m（データの次元）	2, **3**, 4, 5
データの分布	**一様**, クラスタ, 負相関
データ更新率	0, 0.1, 0.2, **0.3**, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0

定された再送回数における検索精度を把握できる．また，式 (4.4) における α および β は，予備実験によりそれぞれ 0.35 および 100 とした．各端末は，[60, 600][sec] の範囲内におけるランダムな間隔で検索クエリを発行する．このとき，w はランダムに決定し，k は，平均および分散が 5 の正規分布に従い，[1, K] の範囲内から決定される（k の影響を調べる場合を除く）．検索範囲 qr は，$qr = 50 \cdot j$[m] であり，j は，平均 5，分散 20 の正規分布に従い，[2, 8] の範囲内から決定される．表 4.1 は，本実験のパラメータを示しており，太字が標準値である．シミュレーション開始から 900[sec] 経過した際の以下の評価値を調べた．

- 初期処理コスト
  クラスタリングを実行した際に送信された全てのメッセージの総バイト．
- データ更新処理コスト
  データ更新が起きた際に送信された全てのメッセージの総バイト．
- 通信量
  Top-k 検索のために送信された全てのメッセージの総バイト数を Top-k 検索回数で割った値．
- アクセス端末数
  クエリメッセージが送信された端末数の平均値．
- 検索精度
  第 2 章および第 3 章のものと同様の方法で計算される．



(a) 端末数 n の影響

(b) データの次元 d の影響

(c) データ分布の影響

(d) データ更新率の影響

図 4.5: 初期処理コストおよびデータ更新処理コスト

- 検索時間

  Top-k 検索に要した時間の平均値.

4.5.1 初期処理コストおよびデータ更新処理コスト

図 4.5 は，各パラメータに対する初期処理コストおよびデータ更新処理コストの結果を示している．また，図 4.5(a) では，実データを用いて端末数の影響を調べた結果を示しており，図 4.5(b)，4.5(c)，および 4.5(d) は，人工データを用いてデータの次元数，データの分布，およびデータの更新率の影響を調べた結果をそれぞれ示している．また，一様分布以外の分布における，データの次元数およびデータの更新率の影響は，一様分布のものの傾向と同様の結果であったため，本論文では割愛する．

図 4.5(a) から，端末数が増加すると，初期処理コストおよびデータ更新処理コストも増加することがわかる．これは，クラスタ構築メッセージおよびメンバ通知メッセージを送信する端末の数が増えることが理由である．また，端末数が増加するとネットワーク内のデータ数も増えるため，*KSKY* が更新された際に生じる通信量も増加する．

データの次元数が増加すると，初期処理コストおよびデータ更新処理コストも増加することが図 4.5(b) からわかる．d が大きい場合，あるデータは他のデータを支配していない場合が多いため，K-スカイバンドに含まれるデータ数が多くなる．そのため *KSKY* のサイズが増加し，クラスタ構築メッセージのサイズや，データ更新の際に送信されるメッセージのサイズも大きくなる．

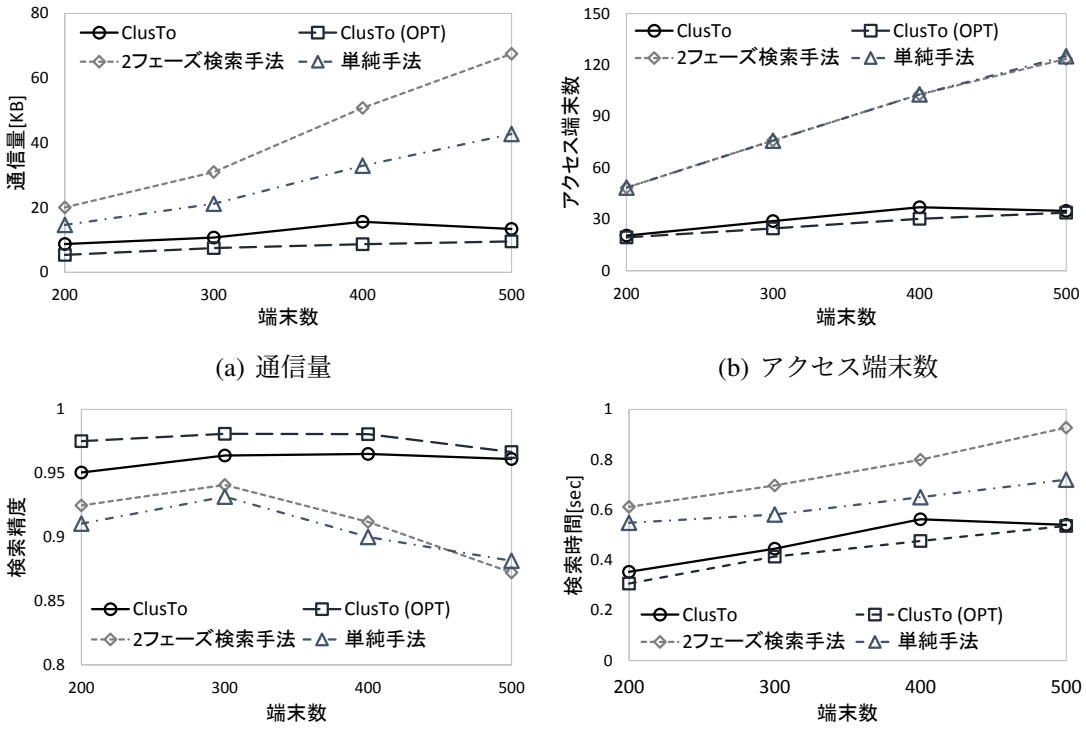
図 4.5(c) より，負相関に従うデータ分布の場合，初期処理コストおよびデータ更新処理コストが大きくなることがわかる．これは，他のデータ分布に比べて支配されているデータ数が少なく，*KSKY* のサイズが大きくなるためである．

図 4.5(d) より，データ更新率が増加すると，データ更新処理コストも大きくなることがわかる．これは直感的な結果であり，データの更新が多い場合，*KSKY* が更新される場合も多くなるためである．

4.5.2 Top-k 検索

n の影響． 図 4.6 は端末数を変えた場合の結果を示しており，図 4.6(a)，4.6(b)，4.6(c)，および 4.6(d) は，それぞれ通信量，アクセス端末数，検索精度，および検索時間の結果を示している．また，これらは実データによる実験の結果である．

図 4.6(a) より，ClusTo は小さい通信量で Top-k 検索を行っていることがわかる．2 フェーズ検索手法および単純手法では，フラッディングによりクエリメッセージの転送を行っているため，端末数が増えると通信量が増加する．一方 ClusTo は，検索範囲内でクエリルーティングを行い，データ収集クエリにより上位 k 個のデータを取得する．そのため，アクセス端末数は不要に増加しない．その結果が図 4.6(b) により示されている．ClusTo は，2 フェーズ検索手法および単純手法に比べてアクセス端末数を大幅に削減している．



(a) 通信量

(b) アクセス端末数

(c) 検索精度

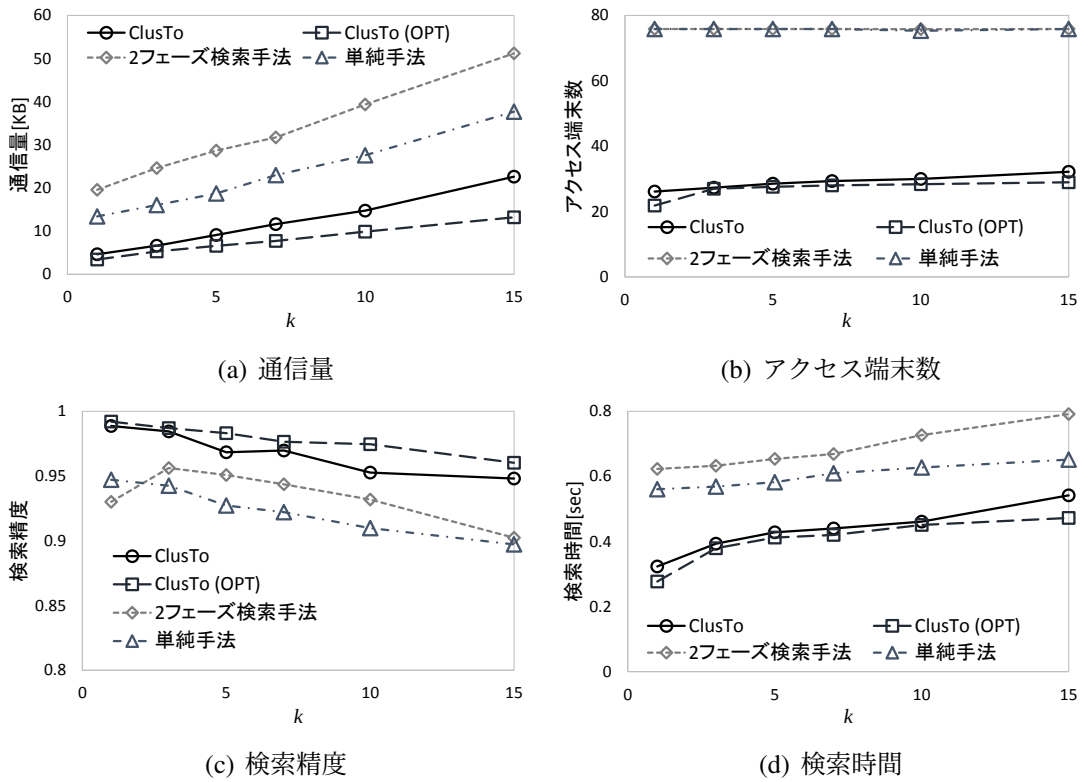
(d) 検索時間

図 4.6: 端末数 n の影響（実データ）

図 4.6(c) より，ClusTo は他の手法よりも効率的にデータを取得していることがわかる．ClusTo は，クエリルーティングにより通信量を削減しており，パケットロスの影響を抑制しているため，データの取得精度が高い．2 フェーズ検索手法および単純手法は，端末数が多い場合にデータの取得精度が小さい．これは，フラッディングによるクエリメッセージの転送により，メッセージ衝突によるパケットロスが頻繁に起きやすく，上位 k 個のデータが取得できないためである．

ClusTo は，比較手法よりも高速にデータを取得できることが図 4.6(d) からわかる．これは，クエリルーティングによりアクセス端末数および通信量を削減し，クエリ応答の再送回数も削減しているためである．そのため，ClusTo(OPT) はさらに検索時間が短い．2 フェーズ検索手法の検索時間が単純手法よりも長いのは，クエリメッセージのフラッディングを 2 度行うことにより生じる遅延が一因である．

k の影響． 図 4.7 は k を変えた場合の結果を示しており，図 4.7(a)，4.7(b)，4.7(c),



(a) 通信量　(b) アクセス端末数

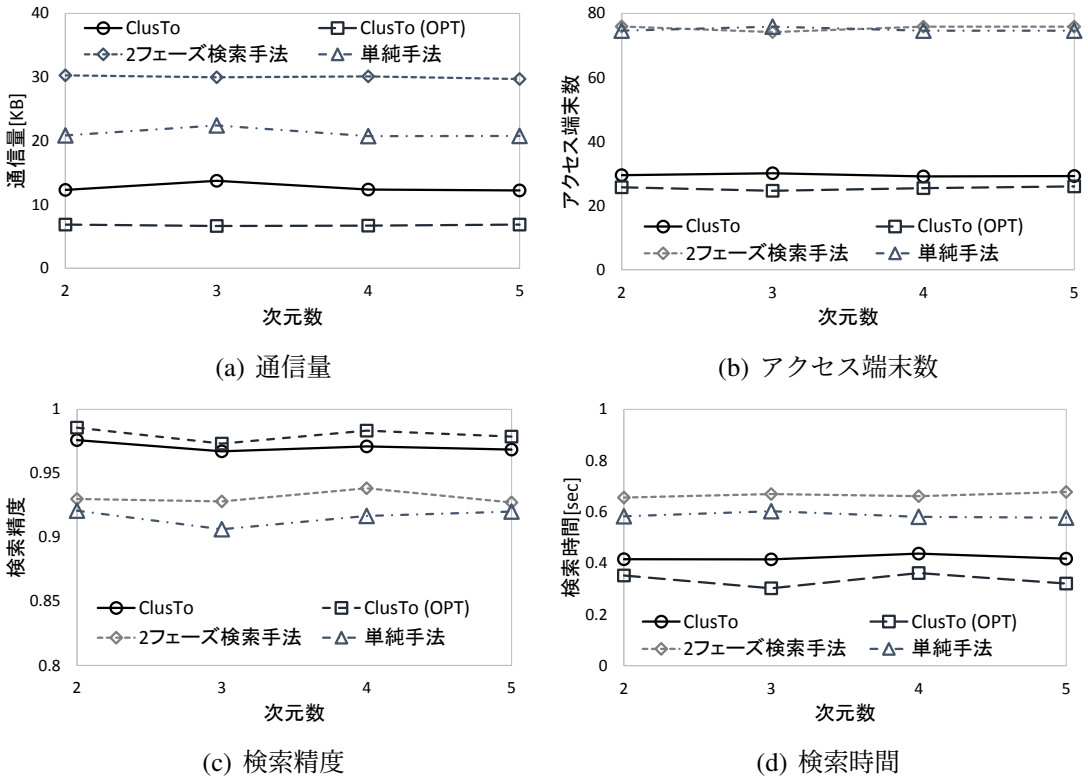
(c) 検索精度　(d) 検索時間

図 4.7: 要求データ数 k の影響（実データ）

および 4.7(d) は，それぞれ通信量，アクセス端末数，検索精度，および検索時間の結果を示している．また，これらも実データによる実験の結果である．

図 4.7(a) より，k が大きくなると通信量が増加することがわかる．これは，返信するデータの数が増えるためである．k が大きい場合，閾値も大きくなるため，データ収集クエリの送信先となる端末も増加する（図 4.7(b)）．また，通信量の増加に伴いパケットロスの影響が大きくなるため，k が大きくなると検索精度が低下することが図 4.7(c) からわかる．しかし，図 4.7(d) からもわかるように，ClusTo は比較手法よりも精度が高く，検索時間も短い．n および k の影響の結果により，フラッディングによるクエリメッセージの転送は通信量が大きく，アドホックネットワークでは非効率であることがわかる．

d の影響． 次に，データの次元数 d の影響を調べた．図 4.8 は，その結果を示している．また，図 4.8(a)，4.8(b)，4.8(c)，および 4.8(d) は，それぞれ通信量，アクセ



(a) 通信量

(b) アクセス端末数

(c) 検索精度

(d) 検索時間

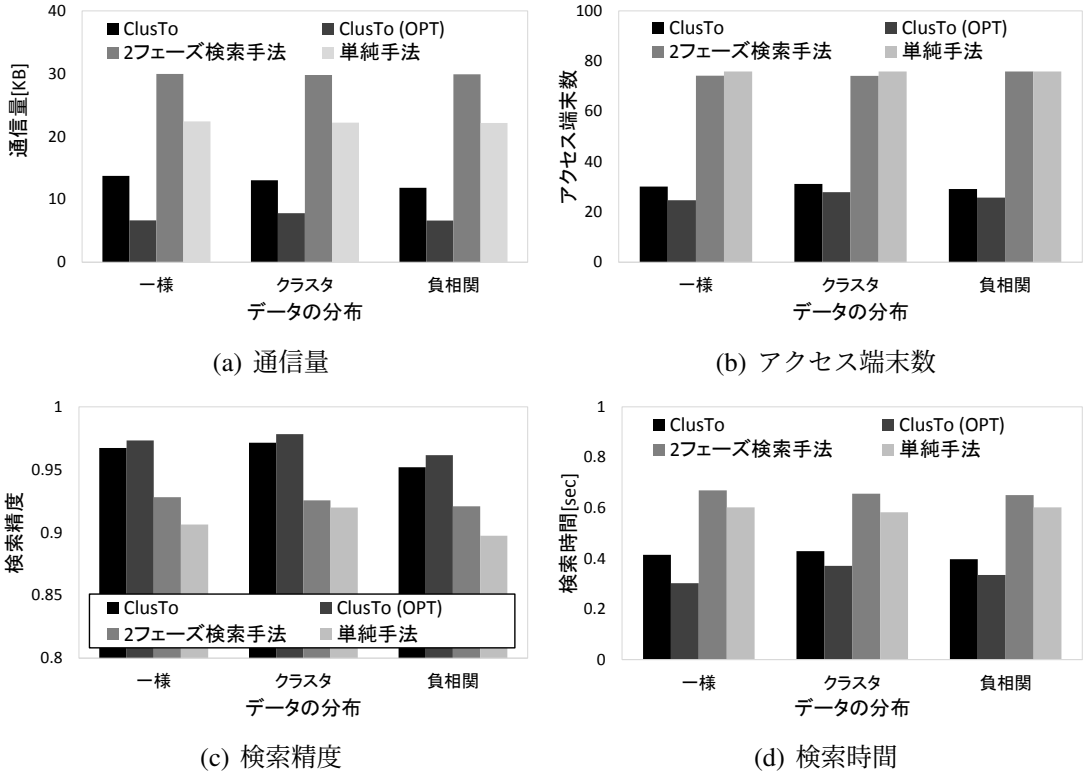
図 4.8: データの次元数 d の影響（人工データ）

ス端末数，検索精度，および検索時間の結果を示している．これ以降では，人工データを用いた実験結果について議論する．

図 4.8(a)–4.8(d) からわかるように，ClusTo の性能は，d に依存していない．ClusTo は 1 次元の値である閾値を用いてクエリメッセージを送信する端末を決定しているため，多次元データ間での比較などのオーバヘッドの大きい処理を必要としない．そのため，初期処理コストは大きくなるが，高次元データに対しても有用であるといった利点がある．

データ分布の影響． 図 4.9 は，データ分布を変えた場合の結果を示しており，図 4.9(a)，4.9(b)，4.9(c)，および 4.9(d) は，それぞれ通信量，アクセス端末数，検索精度，および検索時間の結果を示している．

ClusTo は，これら 3 つのどのデータ分布に対しても高い性能を示していることが図 4.9(a)–4.9(d) からわかる．ClusTo では，上位 k 個に含まれやすいデータを持



(a) 通信量　(b) アクセス端末数

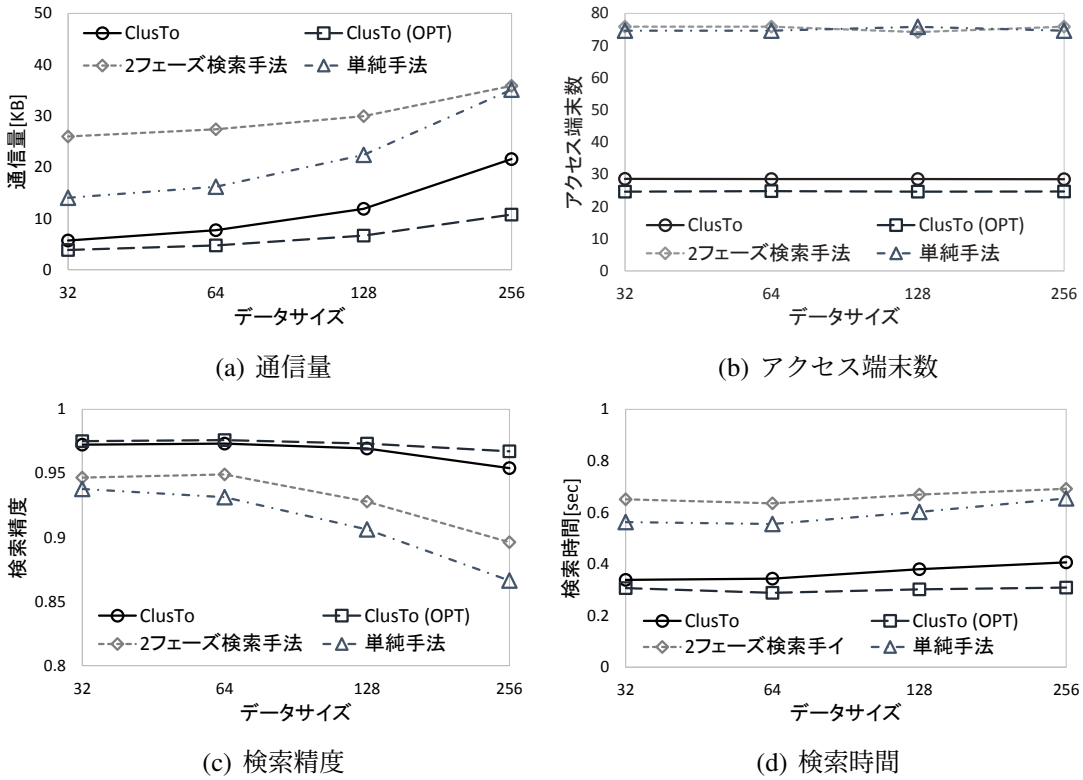
(c) 検索精度　(d) 検索時間

図 4.9: データの分布の影響（人工データ）

つ端末をクラスタヘッドとしているため，データ収集クエリの送信先となる端末の数を抑制できる（図 4.9(b)）．これにより，通信量および検索時間を削減でき（図 4.9(a) および 4.9(d)），スカイラインと原点との距離を用いたクラスタリング（式 (4.4)）が有用であることがわかる．

s の影響． データサイズ s の影響について調べた結果を図 4.10 に示す．図 4.10(a), 4.10(b), 4.10(c), および 4.10(d) は，それぞれ通信量，アクセス端末数，検索精度，および検索時間の結果を示している．

図 4.10(a) により，データサイズが大きくなると通信量が増加することがわかる．これはクエリ応答のメッセージサイズが増加するためである．データサイズが大きい場合には，通信量の増加によりパケットロスが頻繁に生じるため，クエリ応答の再送回数も増加し，通信量がさらに大きくなる．ClusTo では，クエリルーティングおよびクエリメッセージに添付するフィルタによりアクセス端末数を削減し



(a) 通信量

(b) アクセス端末数

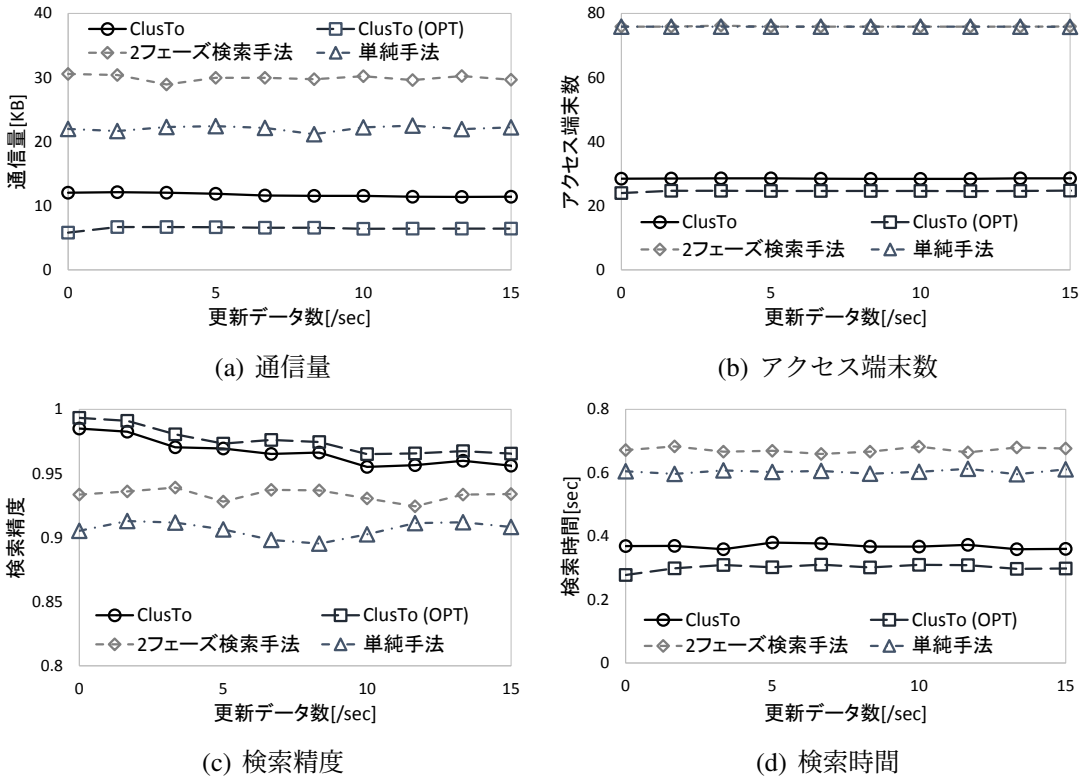
(c) 検索精度

(d) 検索時間

図 4.10: データサイズ s の影響（人工データ）

ているため（図 4.10(b)），パケット衝突を抑制し，再送回数を削減すると共に通信量を削減している．図 4.10(c) から，ClusTo および ClusTo(OPT) は，データサイズが大きい場合においても高い精度を維持していることがわかる．一方，2 フェーズ検索手法および単純手法はデータサイズが大きい場合に検索精度が低下している．これらの手法も無駄なデータの返信は抑制しているが，クエリメッセージのフラッディングによりメッセージ衝突が起こりやすく，パケットロスの影響が非常に大きくなる問題がある．また，アクセス端末数および再送回数の削減などの要因により，ClusTo は比較手法よりも検索時間が短いことが図 4.10(d) からわかる．

データ更新率の影響． 最後に，データ更新に対するスケーラビリティを調べるため，データ更新率を変えて実験を行った．図 4.11 はその結果を示しており，図 4.11(a), 4.11(b), 4.11(c), および 4.11(d) は，それぞれ通信量，アクセス端末数，検索精度，および検索時間の結果を示している．また，これらの図の横軸は 1[sec] あたりに



(a) 通信量

(b) アクセス端末数

(c) 検索精度

(d) 検索時間

図 4.11: データ更新率の影響（人工データ）

更新されたデータの数を示している．

図 4.11(a)–4.11(d) からわかるように，2 フェーズ検索手法および単純手法はデータ更新の影響を受けない．これらの手法では，隣接端末が保持するデータの情報などを利用しないため，データ更新が Top-k 検索処理に影響しない．一方，ClusTo は隣接端末が保持するデータの情報を用いて Top-k 検索を行う．データが更新された場合においても正確な検索を保証するため，各端末は更新された *KSKY* を隣接端末に送信する．クラスタ間クエリは経路表に基づいて送信されるため，このクエリを送信する端末はデータ更新が起きた場合においても基本的に変わらない．また，データ収集クエリは閾値に基づいて送信されるが，データ更新が起きた場合には *KSKY* も更新されるため，検索の正確性は失わない．そのため，通信量，アクセス端末数，および検索時間がデータ更新率に影響しない．

図 4.11(c) により，データ更新率が大きくなると，ClusTo および ClusTo(OPT) の

検索精度がわずかに低下していることがわかる．データ更新率が大きくなると，*KSKY* を送信する機会も増加する．このメッセージが頻繁に送信されると，Top-k 検索のためのメッセージとの衝突が生じ，パケットロスが起きやすくなるため，必要なデータが取得できなくなる．しかし，データ更新率が大きい場合においても，ClusTo は非常に高い精度を示している．

4.6 むすび

本章では，静的なアドホックネットワークにおける多次元データに対する Top-k 検索のためのルーティング手法として，ClusTo を提案した．ユーザは，スコアリング関数と検索範囲を指定し，自身にとって望ましいデータを検索できる．ClusTo では，スカイラインを用いたクラスタリングにより，効率的なクエリルーティングおよびフィルタの更新が可能である．さらに隣接端末の K-スカイバンドを考慮するため，Top-k 検索をより効率的に実行できる．シミュレーション実験の結果から，提案手法は通信量が小さく，検索時間の短い Top-k 検索を実現していることを確認した．また，限られた再送回数においても高い検索精度を維持できていることも確認した．

本章では，ネットワークトポロジが変化しない静的なアドホックネットワークを想定した．しかし，アプリケーションによっては，ユーザは局所的な移動のみではなく，自由に移動し，無線リンクの切断が起きる場合が考えられる．ClusTo は隣接端末の情報を利用して Top-k 検索を行うため，動的なネットワークトポロジの変化への対応が課題である．今後は，このような環境に適応するために，手法の拡張を行う予定である．

第5章　結論

5.1　本論文のまとめ

本論文では，アドホックネットワークにおける Top-k 検索のためのルーティング手法について議論した．まず，第 1 章では，アドホックネットワークにおいて，ネットワークトポロジの動的な変化，また，多様な検索条件へ対応しつつ，通信量を削減する重要性について述べた．

第 2 章では，経路表を用いることにより，Top-k 検索に必要がない端末へのクエリメッセージの転送を抑制する手法を提案した．通信量を削減するためには，上位 k 個のデータの取得に必要ない端末へのクエリメッセージの送信を削減し，必要なデータのみを返信することが有効である．本研究では，まず Top-k 検索のための経路表を作成し，各端末は自身の経路表に従ってクエリメッセージを転送する．検索クエリの発行端末は，検索を要求するデータの順位を添付したクエリメッセージを，各順位に対応する隣接端末にそれぞれ送信する．クエリメッセージを受信した端末は，要求された順位に対応する隣接端末を経路表により確認し，クエリメッセージを転送する．また，要求された全ての順位のデータを保持している端末は，それらのデータのみを返信することにより，クエリ発行端末は上位 k 個のデータを得る．ネットワークトポロジの変化により隣接端末との無線リンク切断が起きた場合には，経路探索を行うことにより経路表を修正し，正確な Top-k 検索を継続する．しかし，提案手法は，各順位のデータを検索するクエリメッセージは，基本的にそれぞれ単一の経路により転送されるため，ネットワークトポロジの変化にロバストでない．そのため，提案手法を拡張し，複数経路によりクエリメッセージを転送する手法を提案した．拡張手法では，各端末が経路表における k 位までに管理されている隣接端末にクエリメッセージを送信することにより

複数経路での転送を実現している．その際，クエリメッセージの受信および送信状況により，クエリメッセージの転送が必要ない隣接端末を確認し，無駄な転送は削減している．これらのアプローチにより，提案手法および拡張手法では，通信量を削減しつつ，高い検索精度を維持することが可能な Top-k 検索を実現した．また，拡張手法は，提案手法よりもクエリメッセージの転送率を向上できるため，さらに高い検索精度を達成した．

第 3 章では，ネットワークトポロジの変化が頻繁に起こる環境や大規模なネットワークに対応することを目指し，クラスタを用いた Top-k 検索のためのルーティング手法を提案した．提案手法では，高順位となるデータを持つ端末をクラスタヘッドとし，これらの端末間でクエリのルーティングを行う．また，クラスタヘッドのメッセージの送信，およびクラスタの再構築を定期的に行うことにより，ネットワークトポロジの変化に対応する．Top-k 検索において，クエリメッセージは，複数のクラスタに属している端末が転送を中継することにより，クラスタヘッドに転送される．また，クラスタヘッドは，各データまでのホップ数を確認することにより，検索すべきデータを自律的に判断し，クエリメッセージの無駄な送信を削減する．クラスタヘッドとの無線リンク切断が起きた場合にのみメッセージをブロードキャストし，これを受信した端末がクラスタヘッドにメッセージを転送することにより，経路探索を行うことなく Top-k 検索を継続できる．そのため，ネットワークトポロジの変化の影響を受けた場合においても，遅延を抑制できる．提案手法では，高順位のデータを持つ端末間でクエリルーティングを行うことにより，ネットワーク内の端末数が多い場合においても，クエリメッセージを送信する端末の数を増やすことはない．これにより，通信量が小さく，検索時間が短い Top-k 検索を実現した．

第 4 章では，多次元データに対する Top-k 検索のためのルーティング手法を提案した．多次元データに対する Top-k 検索では，ユーザが指定する検索条件によって上位 k 個となるデータが異なるため，k 位以内のデータを検索するために必要な隣接端末も検索条件によって異なる．アドホックネットワークでは通信量の削減が重要であるため，ユーザごとに検索条件が異なる場合においても，上位 k 個のデータを取得するために必要な端末のみで Top-k 検索を行うことが望ましい．そ

こで，提案手法では，上位となりやすいデータを持つ端末がクラスタヘッドとなるようにクラスタを構築しつつ，隣接端末が保持する上位 k 個に含まれる可能性があるデータの情報を取得する．Top-k 検索では，クラスタおよび取得したデータの情報を利用することにより，クエリメッセージを送信する．このクエリメッセージには，転送経路中に存在するデータの中で上位のもののスコアが添付されており，各端末はクラスタヘッドに向かってクエリメッセージを送信する．また，クラスタヘッドは上位となりやすいデータを保持しているため，クエリメッセージに添付されるデータの情報も上位 k 個に近いものとなる．これにより，上位 k 個に含まれないデータの返信を削減できると共に，上位 k 個に含まれるデータを保持していない隣接端末へのクエリメッセージの転送も削減できる．提案手法は，通信量および検索時間の観点において，既存手法を上回る高い性能を達成した．

5.2 検討課題

本節では，アドホックネットワークにおける Top-k 検索において，今後解決すべき問題について議論する．

本研究の目的は，アドホックネットワークにおいて，検索条件やネットワークトポロジの変化に影響せず，効率的に上位 k 個のデータを検索するルーティング手法を実現することである．2 章から 4 章では，検索条件またはネットワーク環境に制約を置き，その想定環境における効率的な Top-k 検索のためのルーティング手法を提案した．そして，本論文の提案手法により，クエリルーティングを用いたアプローチは，全端末にクエリメッセージを送信するアプローチよりも検索精度を高く，かつ通信量および検索時間を短くできることを確認した．今後は，モバイルアドホックネットワーク，および Top-k 検索を開始するユーザがそれぞれ検索条件を指定する環境において，効率的に上位 k 個のデータを検索するルーティング手法について考える必要がある．本論文の提案手法により，クエリルーティングを用いたアプローチは，全端末にクエリメッセージを送信するアプローチよりも検索精度を高く，かつ通信量および検索時間を短くできることを確認した．そのため，上の問題に対しても，クエリルーティングを用いたアプローチが有効で

あると考えられる．ここで，2 章で提案したアプローチを単純に用いる方法は，発行されたクエリごとに経路表が必要であるため効率的でない．一方，3 章および 4 章の提案手法のように，高順位となるデータを保持する端末をクラスタヘッドとするアプローチは有効である．しかし，3 章の想定環境では，データの順位が一意であるため，端末が個々に検索条件を指定する環境に直接は適用できない．また，4 章で提案した，スカイラインを用いたクラスタリングは，モバイル環境においても適用可能であるが，モバイルアドホックネットワークでは隣接端末が動的に変化するため，隣接端末の持つデータに基づいてクエリメッセージを送信するアプローチの適用は困難である．上記の問題に対して，通信量および検索時間を削減する効率的なクエリルーティングを実現するには，本論文で提案したクラスタリングのアイデアに加え，隣接端末の動的な変化に対応しつつ，検索に必要がない隣接端末を判断することが課題となる．

また，3 章で述べたように，CTR を拡張し，クラスタおよびデータの複製配置を組み合わせた手法を実現することが考えられる．この方法は，CTR に比べて，クエリメッセージの転送範囲をさらに削減できることが見込まれるため，通信量および取得時間を削減できると予想される．このとき，データの更新が起きた場合に通信量が大きくならないようにデータの複製を配置することが重要である．

本研究では，検索クエリを発行した時点における上位 k 個のデータを効率的に検索する手法を提案した．これらは，あるタイミングでデータを取得することが要求されるアプリケーションにとって有用である．一方，検索クエリを発行後，条件を満たすデータを継続的にモニタリングすることが要求されるアプリケーションも想定される [12, 27, 29]．例えば，4 章で想定した Top-k クエリのように，検索範囲内に存在するデータにおける上位 k 個のものを常にモニタリングする場合，クエリ発行端末の移動により検索範囲も変わるため，上位 k 個となるデータが変わることが考えられる．また，データの更新が頻繁に発生する環境では，上位 k 個のデータも頻繁に変わることが考えられる．このような，Top-k データのモニタリング [56, 85, 88] は多くのアプリケーションにとって有用であるが，アドホックネットワークにおいて，Top-k データを効率的に収集し，モニタリングする手法はこれまでに提案されていない．検索クエリ発行時における上位 k 個のデータの収集は，

本研究で提案した手法により実現可能である．そのため，その後の効率的なモニタリングを実現することが，今後の取り組むべき課題である．

謝辞

本研究全般に関して，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 西尾章治郎教授に謹んで御礼申し上げます.

本研究を推進するにあたり，直接の御指導，御助言，御討論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 原隆浩准教授に衷心より感謝申し上げます.

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，前川卓也准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　藤原融教授，薦田憲久教授，下條真司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，御討論を頂いた名古屋大学大学院情報科学研究科 佐々木勇和氏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島根大学大学院総合理工学研究科　神崎映光准教授，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 白川真澄特任助教，授独立行政法人情報通信研究機構 寺西裕一氏，大阪大学大学院工学研究科 春本要准教授に深謝致します.

筆者の所属する研究グループにおいて，有益な御助言，御討論を頂い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　後藤啓介氏，駒井友香氏，Kamalas Udomlamlert 氏，横山正浩氏，津田琢士氏，大阪大学工学部電子情報工学科 中村義邦氏，中山侑紀氏に深く御礼申し上げます.

本研究を進める上で惜しみない御助言，御協力，研究活動を進めるにあたっての多大なる御支援を頂いた SCSK 株式会社 藤井俊充氏に感謝の意を表します.

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 西尾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いた両親をはじめとする家族に心から感謝致します.

参考文献

[1] Akbarinia, R., Pacitti, E., and Valduriez, P.: Reducing network traffic in unstructured P2P systems using top-k queries, *Distributed and Parallel Databases, Springer*, Vol. 19, No. 2, pp. 67–86 (2006).

[2] 天方大地, 佐々木勇和, 原隆浩, 西尾章治郎：アドホックネットワーク上の効率的な Top-k 検索のためのルーティング手法, 分散, 協調とモバイル (DICOMO 2012) シンポジウム論文集, 第 2012 巻, pp. 1236–1244 (2012).

[3] 天方大地, 佐々木勇和, 原隆浩, 西尾章治郎：アドホックネットワーク上の効率的な Top-k 検索のためのルーティング手法, 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 (DPSWS2012), 第 2012 巻, pp. 9–16 (2012).

[4] Amagata, D., Sasaki, Y., Hara, T., and Nishio, S.: A robust routing method for top-k queries in mobile ad hoc networks, in *Proc. Int. Conf. on Mobile Data Management (MDM)*, pp. 251–256 (2013).

[5] Amagata, D., Sasaki, Y., Hara, T., and Nishio, S.: A routing method for top-k query processing in mobile ad hoc networks, in *Proc. Int. Conf. on Advanced Information Networking and Applications (AINA)*, pp. 42–49 (2013).

[6] 天方大地, 佐々木勇和, 原隆浩, 西尾章治郎：アドホックネットワークにおける Top-k 検索のためのルーティング手法, 情報処理学会論文誌, Vol. 54, No. 8, pp. 2036–2047 (2013).

[7] 天方大地, 佐々木勇和, 原隆浩, 西尾章治郎：アドホックネットワークにおける多次元データ Top-k 検索手法, 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 (DPSWS2013), 第 2013 巻, pp. 212–219 (2013).

[8] 天方大地, 佐々木勇和, 原隆浩, 西尾章治郎：モバイルアドホックネットワークにおけるクラスタを用いた Top-k 検索のためのルーティング手法, 分散, 協調とモバイル (DICOMO 2013) シンポジウム論文集, 第 2013 巻, pp. 561–568 (2013).

[9] Amagata, D., Sasaki, Y., Hara, T., and Nishio, S.: CTR: An Efficient Cluster-based Top-k Query Routing in MANETs, in *Proc. Int. Conf. on Advanced in Mobile Computing & Multimedia (MoMM)*, pp. 225–234 (2014).

[10] 天方大地, 佐々木勇和, 原隆浩, 西尾章治郎：モバイルアドホックネットワークにおけるクラスタを用いた Top-k クエリルーティング手法, 情報処理学会論文誌, Vol. 55, No. 5, pp. 2036–2047 (2014).

[11] Amagata, D., Sasaki, Y., Hara, T., and Nishio, S.: Efficient Multi-dimensional Top-k Query Processing in Wireless Multi-hop Networks, *Mobile Information Systems* (To appear).

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

[13] Balke, W.-T., and Kießling, W.: Optimizing multi-feature queries for image databases, in *Proc. Int. Conf. on Very Large Data Bases (VLDB)*, pp. 10–14 (2000).

[14] Balke, W.-T., Nejdl, W., Siberski, W., and Thaden, U.: Progressive distributed top-k retrieval in peer-to-peer networks, in *Proc. Int. Conf. on Data Engineering (ICDE)*, pp. 174–185 (2005).

[15] Bast, H., Majumdar, D., Schenkel, R., Theobald, M., and Weikum, G.: Io-top-k: Index-access optimized top-k query processing, in *Proc. Int. Conf. on Very Large Data Bases (VLDB)*, pp. 475–486 (2006).

[16] Basu, P., Khan, N., and Little, T. D.: A mobility based metric for clustering in mobile ad hoc networks, in *Proc. Int. Conf. on Distributed Computing Systems (ICDCS) workshop*, pp. 413–418 (2001).

[17] Börzsönyi, S., Kossmann, D., and Stocker, K.: The skyline operator, in *Proc. Int. Conf. on Data Engineering (ICDE)*, pp. 421–430 (2001).

[18] Buckley, C., and Voorhees, E. M.: Evaluating evaluation measure stability, in *Proc. ACM SIGIR Conf. on Research and Development in Information Retrieval (SIGIR)*, pp. 33–40 (2000).

[19] Camp, T., Boleng, J., and Davies, V.: A survey of mobility models for ad hoc network research, *Wireless communications and mobile computing*, Vol. 2, No. 5, pp. 483–502 (2002).

[20] Cao, P., and Wang, Z.: Efficient top-k query calculation in distributed networks, in *Proc. ACM symposium on Principles of Distributed Computing (PODC)*, pp. 206–215 (2004).

[21] Chang, F., Dean, J., Ghemawat, S., Hsieh, W. C., Wallach, D. A., Burrows, M., Chandra, T., Fikes, A., and Gruber, R. E.: Bigtable: A distributed storage system for structured data, *ACM Trans. on Computer Systems (TOCS)*, Vol. 26, No. 2, p. 4 (2008).

[22] Chang, Y.-C., Bergman, L., Castelli, V., Li, C.-S., Lo, M.-L., and Smith, J. R.: The onion technique: indexing for linear optimization queries, in *Proc. ACM SIGMOD Int. Conf. on Management of Data (SIGMOD)*, pp. 391–402 (2000).

[23] Cheema, M. A., Shen, Z., Lin, X., and Zhang, W.: A Unified Framework for Efficiently Processing Ranking Related Queries., in *Proc. Int. Conf. on Extending Database Technology (EDBT)*, pp. 427–438 (2014).

[24] Chen, B., Liang, W., and Min, G.: Top-k query evaluation in sensor networks with the guaranteed accuracy of query results, in *Proc. Int. Conf. on Database and Expert Systems Applications (DEXA)*, pp. 156–171 (2011).

[25] Chen, B., Liang, W., Zhou, R., and Yu, J. X.: Energy-efficient top-k query processing in wireless sensor networks, in *Proc. Int. Conf. on Information and Knowledge Management (CIKM)*, pp. 115–122 (2010).

[26] Chinara, S., and Rath, S. K.: A survey on one-hop clustering algorithms in mobile ad hoc networks, *Journal of Network and Systems Management*, Vol. 17, No. 1–2, pp. 183–207 (2009).

[27] Chow, C.-Y., Mokbel, M. F., and Leong, H. V.: On efficient and scalable support of continuous queries in mobile peer-to-peer environments, *IEEE Trans. on Mobile Computing (TMC)*, Vol. 10, No. 10, pp. 1473–1487 (2011).

[28] Das, G., Gunopulos, D., Koudas, N., and Tsirogiannis, D.: Answering top-k queries using views, in *Proc. Int. Conf. on Very Large Data Bases (VLDB)*, pp. 451–462 (2006).

[29] Do, T. T., Hua, K. A., and Lin, C.-S.: Extrange: Continuous moving range queries in mobile peer-to-peer networks, in *Proc. Int. Conf. on Mobile Data Management (MDM)*, pp. 317–322 (2009).

[30] Ephremides, A., Wieselthier, J. E., and Baker, D. J.: A design concept for reliable mobile radio networks with frequency hopping signaling, *Proc. the IEEE*, Vol. 75, No. 1, pp. 56–73 (1987).

[31] Fagin, R., Lotem, A., and Naor, M.: Optimal aggregation algorithms for middleware, *Journal of Computer and System Sciences, Elsevier*, Vol. 66, No. 4, pp. 614–656 (2003).

[32] Haas, Z. J., and Pearlman, M. R.: The performance of query control schemes for the zone routing protocol, *IEEE/ACM Trans. Networking*, Vol. 9, No. 4, pp. 427–438 (2001).

[33] Hagihara, R., Shinohara, M., Hara, T., and Nishio, S.: A message processing method for top-k query for traffic reduction in ad hoc networks, in *Proc. Int. Conf. on Mobile Data Management (MDM)*, pp. 11–20 (2009).

[34] Hara, T., Hagihara, R., and Nishio, S.: Data replication for top-k query processing in mobile wireless sensor networks, in *Proc. Int. Conf. on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC)*, pp. 115–122 (2010).

[35] Heo, J.-S., Cho, J., and Whang, K.-Y.: The hybrid-layer index: A synergic approach to answering top-k queries in arbitrary subspaces, in *Proc. Int. Conf. on Data Engineering (ICDE)*, pp. 445–448 (2010).

[36] Hristidis, V., Koudas, N., and Papakonstantinou, Y.: Prefer: A system for the efficient execution of multi-parametric ranked queries, in *Proc. ACM SIGMOD Int. Conf. on Management of Data (SIGMOD)*, pp. 11–20 (2001).

[37] Hwang, S.-w., and Chang, K. C.-c.: Optimizing top-k queries for middleware access: A unified cost-based approach, *ACM Trans. on Database Systems (TODS)*, Vol. 32, No. 1, p. 5 (2007).

[38] IEEE 802.11, The Working Group for Wireless LANs
: *http://grouper.ieee.org/groups/802/11/.*

[39] Ilyas, I. F., Beskales, G., and Soliman, M. A.: A survey of top-k query processing techniques in relational database systems, *ACM Computing Surveys (CSUR)*, Vol. 40, No. 4, p. 11 (2008).

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

[41] Jiang, H., Cheng, J., Wang, D., Wang, C., and Tan, G.: A general framework for efficient continuous multidimensional top-k query processing in sensor net-

works, *IEEE Trans. on Parallel and Distributed Systems (TPDS)*, Vol. 23, No. 9, pp. 1668–1680 (2012).

[42] Johnson, D. B., and Maltz, D. A.: Dynamic source routing in ad hoc wireless networks, *Kluwer International Series in Engineering and Computer Science*, Vol. 353, pp. 153–179 (1996).

[43] Junhai, L., Danxia, Y., Liu, X., and Mingyu, F.: A survey of multicast routing protocols for mobile ad-hoc networks, *Communications Surveys & Tutorials*, Vol. 11, No. 1, pp. 78–91 (2009).

[44] Junhai, L., Liu, X., and Danxia, Y.: Research on multicast routing protocols for mobile ad-hoc networks, *Computer Networks*, Vol. 52, No. 5, pp. 988–997 (2008).

[45] Kießling, W.: Foundations of preferences in database systems, in *Proc. Int. Conf. on Very Large Data Bases (VLDB)*, pp. 311–322 (2002).

[46] Komai, Y., Sasaki, Y., Hara, T., and Nishio, S.: kNN Query Processing Methods in Mobile Ad Hoc Networks, *IEEE Trans. on Mobile Computing (TMC)*, Vol. 13, No. 5, pp. 1090–1103 (2014).

[47] Lee, J., Cho, H., and Hwang, S.-w.: Efficient dual-resolution layer indexing for top-k queries, in *Proc. Int. Conf. on Data Engineering (ICDE)*, pp. 1084–1095 (2012).

[48] Liu, X., Xu, J., and Lee, W. C.: A cross pruning framework for top-k data collection in wireless sensor network, in *Proc. Int. Conf. on Mobile Data Management (MDM)*, pp. 157–166 (2010).

[49] Madden, S., Franklin, M. J., Hellerstein, J. M., and Hong, W.: TAG: A tiny aggregation service for ad-hoc sensor networks, *ACM SIGOPS Operating Systems Review*, Vol. 36, No. SI, pp. 131–146 (2005).

[50] 松波秀和, 寺田努, 西尾章治郎：P2P 型コンテンツ検索システムのための効率的な Top-k 検索処理手法, 情報処理学会論文誌, Vol. 47, No. 9, pp. 2850–2859 (2006).

[51] Menchaca-Mendez, R., and Garcia-Luna-Aceves, J.: An interest-driven approach to integrated unicast and multicast routing in MANETs, in *Proc. Int. Conf. on Network Protocol (ICNP)*, pp. 248–257 (2008).

[52] Menchaca-Mendez, R., and Garcia-Luna-Aceves, J.: Hydra: Efficient multicast routing in MANETs using sender-initiated multicast meshes, *Pervasive and Mobile Computing*, Vol. 1, No. 1, pp. 144–157 (2010).

[53] Menchaca-Mendez, R., and Garcia-Luna-Aceves, J.: Robust and scalable integrated routing in MANETs using context-aware ordered meshes, in *Proc. Int. Conf. on Computer Communications (INFOCOM)*, pp. 1–9 (2010).

[54] Michel, S., Peter, T., and Weikum, G.: KLEE: A framework for distributed top-k query algorithms, in *Proc. Int. Conf. on Very Large Data Bases (VLDB)*, pp. 637–648 (2005).

[55] Mnaouer, A. B., Chen, L., Foh, C. H., and Tantra, J. W.: OPHMR: an optimized polymorphic hybrid multicast routing protocol for MANET, *IEEE Trans. Mobile Computing (TMC)*, Vol. 6, No. 5, pp. 551–562 (2007).

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

[57] Niedermayer, J., Nascimento, M. A., Renz, M., krøger, P., and Kriegel, H. P.: Exploiting local node cache in top-k queries within wireless sensor networks, in *Proc. Int. Conf. on Advanced in Geographical Information Systems (ACM SIGSPATIAL GIS)*, pp. 434–437 (2010).

[58] Padhariya, N., Mondal, A., Goyal, V., Shankar, R., and Madria, S.: EcoTop: an economic model for dynamic processing of top-k queries in mobile-P2P networks, in *Proc. Int. Conf. on Database Systems for Advanceed Applications (DASFAA)*, pp. 251–265 (2011).

[59] Papadias, D., Tao, Y., Fu, G., and Seeger, B.: Progressive skyline computation in database systems, *ACM Trans. on Database Systems (TODS)*, Vol. 30, No. 1, pp. 41–82 (2005).

[60] Parekh, A. K.: Selecting routers in ad-hoc wireless networks, in *Proc. Int. Telecommunications Symposium*, pp. 420–424 (1994).

[61] Perkins, C. E., and Royer, E. M.: Ad-hoc on-demand distance vector routing, in *Proc. Int. Conf. Mobile Computing Systems and Applications*, pp. 90–100 (1999).

[62] Ryeng, N., Vlachou, A., Doulkeridis, C., and Nørvåg, K.: Efficient distributed top-k query processing with caching, in *Proc. Int. Conf. on Database Systems for Advanceed Applications (DASFAA)*, pp. 280–295 (2011).

[63] Sasaki, Y., Hagihara, R., Hara, T., Shinohara, M., and Nishio, S.: A top-k query method by estimating score distribution in mobile ad hoc networks, in *Proc. Int. Conf. on Advanced Information Networking and Applications (AINA) Workshops*, pp. 944–949 (2010).

[64] Sasaki, Y., Hara, T., and Nishio, S.: Two-phase top-k query processing in mobile ad hoc networks, in *Proc. Int. Conf. on Network-Based Information Systems (NBiS)*, pp. 42–49 (2011).

[65] 佐々木勇和, 原隆浩, 西尾章治郎：モバイルアドホックネットワーク上の Top-k 検索のための複製配置およびメッセージ処理手法, 日本データベース学会論文誌, Vol. 12, No. 1, pp. 115–120 (2013).

[66] Sasaki, Y., Hara, T., and Nishio, S.: Top-k query processing for replicated data in mobile peer to peer networks, *Journal of Systems and Software, Elsevier*, Vol. 92, pp. 45–58 (2014).

[67] Sasaki, Y., Lee, W.-C., Hara, T., and Nishio, S.: On Alleviating Beacon Overhead in Routing Protocols for Urban VANETs, in *Proc. Int. Conf. Mobile Data Management (MDM)*, Vol. 1, pp. 66–76 (2013).

[68] Scalable Network Technologoes: Creaters of QualNet Network Simulator Software: *http://www.scalable-networks.com/*.

[69] Silberstein, A., 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)*, pp. 68–78 (2006).

[70] Silberstein, A., Munagala, K., and Yang, J.: Energy-efficient monitoring of extreme values in sensor networks, in *Proc. ACM SIGMOD Int. Conf. on Management of Data (SIGMOD)*, pp. 169–180 (2006).

[71] Tang, S., Mao, X., and Li, X.: Efficient and fast distributed top-k query protocol in wireless sensor networks, in *Proc. Int. Conf. on Network Protocol (ICNP)*, pp. 99–108 (2011).

[72] 寺田努, 松波秀和, 西尾章治郎：P2P 型コンテンツ検索システムにおけるコンテンツ分布を考慮した Top-k 検索処理手法, 情報処理学会論文誌, Vol. 48, No. 14, pp. 1–10 (2007).

[73] The Official Bluetooth Wireless Info Site: *http://www.bluetooth.com/*.

[74] Theobald, M., Weikum, G., and Schenkel, R.: Top-k query evaluation with probabilistic guarantees, in *Proc. Int. Conf. on Very Large Data Bases (VLDB)*, pp. 648–659 (2004).

[75] Tseng, Y.-C., Ni, S.-Y., Chen, Y.-S., and Sheu, J.-P.: The broadcast storm problem in a mobile ad hoc network, *Wireless networks*, Vol. 8, No. 2–3, pp. 183–207 (2002).

[76] Vaishampayan, R., and Garcia-Luna-Aceves, J. J.: Efficient and robust multicast routing in mobile ad hoc networks, in *Proc. Int. Conf. Mobile Ad-Hoc and Sensor Systems (MASS)*, pp. 304–313 (2004).

[77] Vaishampayan, R., Garcia-Luna-Aceves, J., and Obraczka, K.: Routing with adaptive path and limited flooding for mobile ad hoc networks, *Computers & electrical engineering*, Vol. 36, No. 2, pp. 280–290 (2010).

[78] Vlachou, A., Doulkeridis, C., and Nørvåg, K.: Distributed top-k query processing by exploiting skyline summaries, *Distributed and Parallel Databases, Springer*, Vol. 30, No. 3-4, pp. 239–271 (2012).

[79] Vlachou, A., Doulkeridis, C., Nørvåg, K., and Vazirgiannis, M.: On efficient top-k query processing in highly distributed environments, in *Proc. ACM SIGMOD Int. Conf. on Management of Data (SIGMOD)*, pp. 753–764 (2008).

[80] Vlachou, A., Doulkeridis, C., Nørvåg, K., and Vazirgiannis, M.: Skyline-based peer-to-peer top-k query processing, in *Proc. Int. Conf. on Data Engineering (ICDE)*, pp. 1421–1423 (2008).

[81] Wi-Fi Peer-to-Peer: *http://developer.android.com/guide/topics/connectivity/wifip2p.html*.

[82] Wu, M., Jianliang Xu, J., Xueyan Tang, X., and Wang-Chien Lee, W.-C.: Top-k Monitoring in Wireless Sensor Networks, *IEEE Trans. on Knowledge and Data Engineering (TKDE)*, No. 7, pp. 962–976 (2007).

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

[84] Xie, M., Lakshmanan, L. V., and Wood, P. T.: Efficient top-k query answering using cached views, in *Proc. Int. Conf. on Extending Database Technology (EDBT)*, pp. 489–500 (2013).

[85] Yang, D., Shastri, A., Rundensteiner, E. A., and Ward, M. O.: An optimal strategy for monitoring top-k queries in streaming windows, in *Proc. Int. Conf. on Extending Database Technology (EDBT)*, pp. 57–68 (2011).

[86] Yang, W.-d., and Zhang, G.-z.: A weight-based clustering algorithm for mobile ad hoc network, in *Proc. Int. Conf. Wireless and Mobile Communications*, p. 3 (2007).

[87] Yang, W., Yang, X., Yang, S., and Yang, D.: A greedy-based stable multi-path routing protocol in mobile ad hoc networks, *Ad Hoc Networks*, Vol. 9, No. 4, pp. 662–674 (2011).

[88] Yu, A., Agarwal, P., and Yang, J.: Processing a large number of continuous preference top-k queries, in *Proc. ACM SIGMOD Int. Conf. on Management of Data (SIGMOD)*, pp. 397–408 (2012).

[89] Zhao, K., Tao, Y., and Zhou, S.: Efficient top-k processing in large-scaled distributed environments, *Data & Knowledge Engineering (DKE), Elsevier*, Vol. 63, No. 2, pp. 315–335 (2007).

[90] ZigBee Alliance: *http://www.zigbee.org/*.

[91] Zou, L., and Chen, L.: Dominant Graph: An Efficient Indexing Structure to Answer Top-K Queries, in *Proc. Int. Conf. on Data Engineering (ICDE)*, pp. 536–545 (2008).

[92] Zou, L., and Chen, L.: Pareto-based dominant graph: An efficient indexing structure to answer top-k queries, *IEEE Trans. on Knowledge and Data Engineering (TKDE)*, Vol. 23, No. 5, pp. 727–741 (2011).