位置・キーワードに基づく kNN データモニタリングの分散処理のためのクエリ割り当てアルゴリズム

鶴岡　翔平† 　西尾　俊哉†† 　天方　大地†† 　原　隆浩††

† 大阪大学工学部電子情報工学科　〒 565-0871 大阪府吹田市山田丘 1-5
†† 大阪大学大学院情報科学研究科　〒 565-0871 大阪府吹田市山田丘 1-5
E-mail: †{tsuruoka.shohei,nishio.syunya,amagata.daichi,hara}@ist.osaka-u.ac.jp

あらまし　近年，パブリッシュ/サブスクライブ（Pub/Sub）モデルに基づいてデータを配信するアプリケーションが多く存在し，ユーザは自身の興味のあるデータのみを取得する．また，スマートフォンや SNS の普及により，データやクエリの数が増加しており，単一のサーバで全ての処理を行うとリアルタイム性を保持することが困難になっている．本稿では，Pub/Sub モデルにおいて，複数のサーバにクエリを分散して処理を分担することにより，各クエリに対して，クエリのキーワードを含むデータの中でクエリに最も近い k 個のデータ（kNN データ）を効率的にモニタリングする問題に取り組む．単純にクエリを分散させると，各サーバの処理量に偏りが生まれ，その結果，新たなデータが生成されたとき，処理量の大きいサーバの処理時間が長くなり，全体の処理時間も長くなる．この問題を解決するため，各サーバの処理時間が均一かつ全体の処理時間が短くなるようにクエリを分散するアルゴリズムを提案する．実データを用いた実験により，提案手法の有効性を示す．

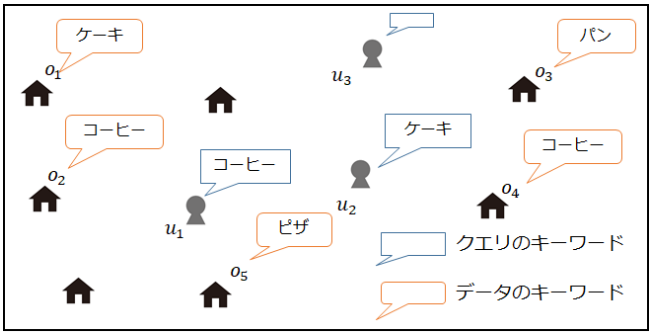
キーワード　Pub/Sub，kNN クエリ，分散処理

1　はじめに

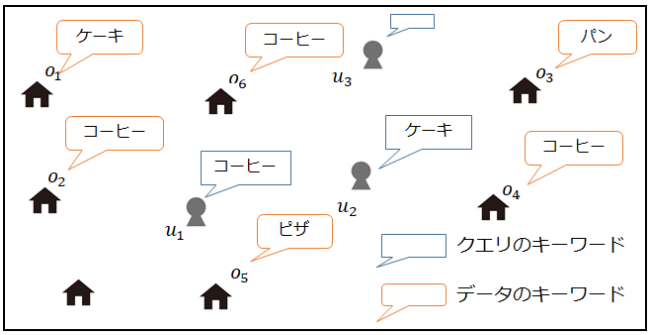
近年，GPS 機能付きのモバイル端末の普及により，ジオタグ付きのツイートなどの位置情報とキーワードを持つデータが大量に生成されている．それらのデータの中から，ユーザが自身の興味のあるデータを得るための手段としてパブリッシュ/サブスクライブ（Pub/Sub）モデルに基づくアプリケーションが多く存在する [7], [9]．Pub/Sub モデルでは，ユーザは事前にサーバへクエリを登録し，サーバは登録されたクエリに基づいてユーザにデータを配信する．大量にデータが存在する環境では，クエリのキーワードを含むデータの中で，クエリに最も近い k 個のデータ (kNN データ) を検索する kNN クエリが有用である [5]．さらに，Pub/Sub モデルでは頻繁にデータが生成されるため，各クエリの kNN データは頻繁に変化する．

例 1. 図 1 は，Pub/Sub モデルにおいて，3 人のユーザが自身の興味のあるデータをモニタリングする例を示している．各ユーザはクエリ（位置情報およびキーワード）を登録しており，クエリに合う k 個のデータをモニタリングしている．$k = 2$ と仮定すると，時刻 t において，ユーザ u_1 の kNN データは，「コーヒー」を含み，かつ u_1 に最も近いデータ o_2 および o_4 である．時刻 $t+1$ において，o_6 が生成されると，各クエリの kNN データが更新される．o_6 は，「コーヒー」を含み，o_4 よりも u_1 に近いため，u_1 の kNN データは，o_2 および o_6 に変化する．

ここで，スマートフォンや SNS の普及により，生成されるデータやサーバに登録されるクエリの数は増加している．データやクエリの数が増加すると，新たに生成されるデータに対するサーバの処理量が増加するため，単一のサーバで全ての処理を行う場合にリアルタイム性を保証することが難しくなる．そこで，本稿では，複数のサーバにクエリを分散して処理を分担することにより，各クエリの kNN データを効率的にモニタリ



(a) 時刻 t



(b) 時刻 $t+1$

図 1: Pub/Sub モデルにおける kNN データのモニタリングの例

ングする問題に取り組む．

課題． 複数のサーバにクエリを分散して各クエリの kNN データを効率的にモニタリングするためには，クエリをどのようにサーバへ分散するかが重要である．単純な方法では，同じキーワードを持つクエリを一つのサーバで管理する方法が考えられる．しかし，あるキーワードが多くのデータやクエリに含まれる場合，そのキーワードを持つクエリを管理するサーバの処理量が大きくなってしまう．また，もう一つの方法として，クエリが存在する領域をサーバの数に等分割し，各領域に含まれるクエリを一つのサーバで管理する方法が考えられる．しかし，ある領域にデータやクエリが多く存在する場合，その領域に含まれるクエリを管理するサーバの処理量が大きくなってしまう．このように，単純にクエリを分散させると，各サーバの処理量に偏りが生まれる．その結果，新たなデータが生成されたとき，処理量の大きいサーバの処理時間が長くなり，全体の処理時間も長くなる．

提案アルゴリズムの概要． この問題を解決するため，まず，あるデータが生成されたとき，各サーバにおける処理時間を予測するため，サーバの負荷を定義する，あるサーバの負荷は，そのサーバで管理するクエリの負荷の合計と定義し，あるクエリの負荷はデータやクエリの位置情報とキーワードに基づいて計算される．そして，各サーバの負荷を均一にしつつ，全体の負荷を小さくするようにクエリを分散するアルゴリズムを提案する．具体的には，全てのクエリの集合 Q を，集合の数が一定数以上になるまで，部分集合 Q_i に分割する．そして，各サーバの負荷が均一になるように，Q_i に含まれるクエリを管理するサーバを決定する．これにより，各サーバの負荷を均一にしつつ，全体の負荷をより小さくするようにクエリを分散する．

貢献． 以下に本研究の貢献を示す．

- 複数のサーバでクエリを分散管理して処理を分担することにより，各クエリの kNN データを効率的にモニタリングする問題に取り組む．著者らの知る限り，この問題はこれまでに取り組まれていない．
- 各サーバの負荷を均一にしつつ，全体の負荷を小さくするようにクエリを分散するアルゴリズムを提案する．
- 実データを用いた実験により，提案アルゴリズムの有効性を確認する．

本稿の構成． 2 章で関連研究について説明し，3 章で本稿の問題を定義する．4 章で提案アルゴリズムについて説明し，5 章で実データを用いた実験の結果を示す．最後に 6 章で本稿をまとめる．

2 関 連 研 究

本章では，クエリの分散処理の関連研究を紹介する．

Pub/Sub モデルにおけるクエリの分散． 様々な研究が，Pub/Sub モデルにおいて，複数のサーバでクエリを分散して管理し，クエリの解をモニタリングする問題に取り組んでいる [2], [4], [10]．文献 [4] では，位置情報およびキーワードに基づいてクエリを分散するアルゴリズムが提案されている．しかし，この研究はレンジクエリを対象としており，kNN クエリを対象とした本研究とは問題が異なる．また，文献 [2], [10] では，Pub/Sub モデルにおいて，キーワードに基づいてクエリを分散するアルゴリズムが提案されている．例えば，文献 [2] では，データのキーワードの出現頻度に基づいて，各サーバが担当するキーワードを決定し，担当するキーワードを持つクエリを管理する．しかし，これらの研究では，データおよびクエリの位置情報を考慮しておらず，位置情報を考慮した場合，提案アルゴリズムは効果的ではない．

位置情報を用いたクエリの分散処理． 文献 [1], [6] では，位置情報に基づいてデータを分散するアルゴリズムが提案されている．例えば，文献 [6] では，データが含まれる領域をサーバの数に分割し，1 つの領域に含まれるデータを 1 つのサーバで管理する．そして，ある kNN クエリの解を計算するとき，まず，クエリの位置を含む領域の中で kNN データを検索する．その後，クエリと k 番目のデータの距離の範囲が，他の領域に重複する場合，その領域に含まれるデータも含めて kNN データを検索する．しかし，これらの研究では，データおよびクエリのキーワードを考慮していない．また，MapReduce 環境を想定しており，ストリームデータに適用できない．

3 問 題 定 義

3.1 定　　義

これまでに生成されたデータの集合を O，発行されたクエリの集合を Q とする．

データモデル． あるデータ $o \in O$ は位置（$o.loc$），およびキーワード集合 ($o.key$) で構成される（$o = \langle loc, key \rangle$）．$o.loc$ は，緯度と経度によって表される 2 次元平面上の点である．

クエリモデル． ある kNN クエリ $q \in Q$ は，位置（$q.loc$），キーワード集合 ($q.key$)，および何個のデータを受信するかを指定する変数 ($q.k$) で構成される（$q = \langle loc, key, k \rangle$）．$q.key$ は，key_{max} 個以下のキーワードを指定する．O が与えられたとき，q の kNN データ A は次の条件を満たす．(i) $|A| = k$, (ii)$\forall o \in A, \forall o' \in O_q - A, dist(o.loc, q.loc) \leqq dist(o'.loc, q.loc)$ ここで，O_q は O の中で $q.key$ に含まれるキーワードを一つ以上含むデータ集合，$dist(\cdot, \cdot)$ はデータとクエリとのユークリッド距離である．

問題定義． O および Q が与えられたとき，Q に含まれる全てのクエリの kNN データをモニタリングする．

3.2 最適なクエリの分散

本稿では，複数のサーバでクエリを分散して管理することで，各クエリの kNN データを効率的にモニタリングする．複数のサーバでクエリを管理する際，各サーバの負荷を均一にしつつ全体の負荷を小さくすることが望ましい．まず，サーバの負荷を定義する．

定義 1（サーバの負荷）. あるサーバ s_i の負荷 $L(s_i)$ を以下で定義する.

$$L(s_i) = \sum_{q \in Q_{s_i}} L(q)$$

ここで，Q_{s_i} はサーバ s_i で管理されているクエリの集合である．また，$L(q)$ はクエリ q の負荷であり，データおよびクエリの位置情報とキーワードに基づいて計算される.

次に，サーバの負荷に基づいて，各サーバの負荷を均一にしつつ，全体の負荷を最小にするクエリの分散を決定するクエリ分散問題を定義する.

定義 2(クエリ分散問題). O，Q およびサーバの数 m が与えられたとき，クエリ分散問題は，Q を以下の条件を満たす m 個の集合 Q_1，Q_2，$\cdots$，Q_m に分割する.（i）$\sum_{i=1}^{m} L(s_i)$ が最小（ii）$\forall i \neq j$ に対して，$L(s_i) \geqq L(s_j)$，$L(s_i)/L(s_j) \leqq \sigma$．ここで，$\sigma \simeq 1$ である.
このとき，以下の定理が成り立つ.

定理 1. クエリ分散問題は NP 困難である.

証明. n 個の整数 $a_1, a_2, \cdots, a_n$ を 2 つの集合に分けるとき，各集合に含まれる整数の和が等しくなる集合に分ける問題を分割問題と呼び，この問題は NP 完全である．文献 [8] では，分割問題を拡張し，n 個の整数 $a_1, a_2, \cdots, a_n$ を，k 個の集合に分けるとき，各集合に含まれる整数の和の中で，最大のものと最小のものの差が最小となる集合に分ける問題に取り組んでおり，この問題は NP 困難である．ここで，$m = k$ および $\sigma = 1$ であるとき，クエリ分散問題は，

$$\sum_{q_1 \in Q_1} L(q_1) = \sum_{q_2 \in Q_2} L(q_2) = \cdots = \sum_{q_k \in Q_k} L(q_k)$$

を満たすような k 個の集合 $Q_1, Q_2, \cdots, Q_k$ を見つけることである．これは，文献 [8] で取り組まれている問題を解くことと同じである．よって，定理が成り立つ. □

3.3 ベースラインアルゴリズム

クエリ分散問題は NP 困難であるため，貪欲法を用いてクエリの分散を決定する．まず，ベースラインとなるアルゴリズムを紹介する.

3.3.1 キーワードに基づくクエリの分散

同じキーワードを持つクエリを 1 つのサーバで管理する方法のように，単純にクエリを分散すると，各サーバの負荷に偏りが生まれる．そこで，データに頻出するキーワードを持つクエリから順に，サーバに分散することで，各サーバの負荷を均一にする.

まず，データのキーワードが時間により変化しない，かつ各キーワードの出現確率が独立であると仮定すると，新たなデータ o が生成されたとき，$o.key$ にキーワード w が含まれる確率 $P_t(w)$ は以下の式で計算される.

$$P_t(w) = \frac{|O_w|}{|O|}$$

ここで，O_w は O の中で w を含むデータの集合である．この

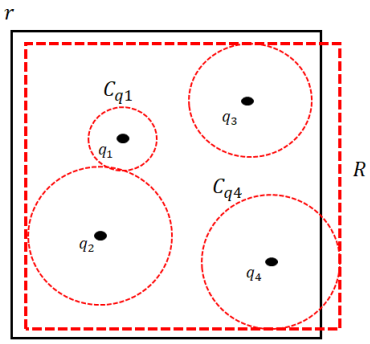


図 2: kNN の更新の範囲

とき，あるクエリ q の kNN データとなりえるデータは，$q.key$ に含まれるキーワードを 1 つ以上含むデータであるため，$L(q)$ は $P_t(w)$ を用いて，以下のように計算される.

$$L(q) = \sum_{w \in q.key} P_t(w)$$

ここから，あるクエリを管理するサーバを決定するアルゴリズムを紹介する．まず，全てのクエリ $q \in Q$ に対して $L(q)$ を計算する．そして，$L(q)$ の大きいクエリから順に，クエリを管理するサーバを決定する．あるクエリ q を管理するサーバは，その時点でサーバの負荷が最も小さいサーバ s_i とする．この操作を，全てのクエリに対してそれらを管理するサーバを決定するまで繰り返すことで，クエリの分散を決定する.

新たなデータ o が生成されたとき，各サーバで管理するクエリの kNN データを更新するアルゴリズムを紹介する．各サーバ s は，自身の管理するクエリに対して，キーワードごとにクエリへのポインタを管理する転置ファイル $s.I$ を保持する．$s.I$ を用いて，o のキーワードを含むクエリの集合 Q_c を取得し，$q \in Q_c$ の kNN データを更新する.

3.3.2 位置情報に基づくクエリの分散

クエリが存在する領域をサーバの数に等分割し，各領域に含まれるクエリを 1 つのサーバで管理する方法のように，単純にクエリを分散すると，各サーバの負荷に偏りが生まれる．そこで，各領域に含まれるクエリ集合の負荷を考え，負荷の大きい領域を小さい領域に分割する．そして，各サーバで複数の領域に含まれるクエリ集合を管理することで，各サーバの負荷を均一にする.

まず，ある領域 r に含まれるクエリ集合を Q_r とし，Q_r の負荷 $L(Q_r)$ を考える．$L(Q_r)$ は，新たに生成されたデータにより Q_r 内に含まれるクエリの kNN データが変化する可能性がある確率と考えることができる．ここで，クエリ q の k 番目のデータを o_k とすると，新たに生成されたデータ o が q の kNN データとなるには，$dist(o.loc, q.loc) < dist(o_k.loc, q.loc)$ が必要である．つまり，$q.loc$ から $dist(o_k.loc, q.loc)$ の範囲を C_q とすると，C_q 内に生成されたデータが q の kNN データとなりえる．したがって，Q_r に含まれる全てのクエリの C_q を包含する領域を R とすると，R 内で生成されるデータが，Q_r 内に含まれるクエリの kNN データとなりえる．例えば，図 2 において，

ある領域 r 内に 4 つのクエリが存在している．このとき，それぞれのクエリの C_q を包含する領域が R である．また，データの位置の分布が時間により変化しないと仮定すると，R 内に新たなデータが生成される確率 $P_s(R)$ は以下の式で計算される．

$$P_s(R) = \frac{|O_R|}{|O|}$$

ここで，O_R は R に含まれるデータの集合である．よって，$L(Q_r)$ は，$P_s(R)$ を用いて，以下のように計算される．

$$L(Q_r) = P_s(R) \times |Q_r|$$

さらに，各サーバは，複数の領域に含まれるクエリ集合を管理するため，$L(s_i)$ は，$L(Q_r)$ を用いて，以下のように計算される．

$$L(s_i) = \sum_{Q_i \in s_i.\mathbb{Q}} L(Q_r)$$

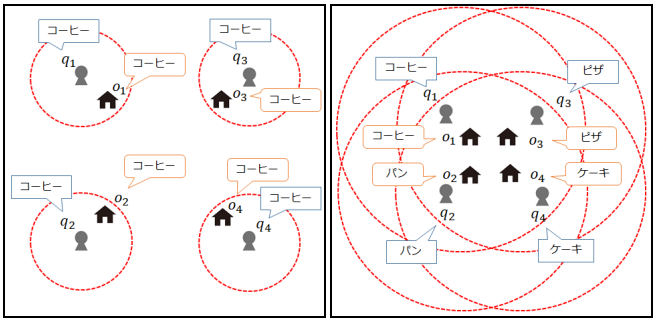
ここで，$s_i.\mathbb{Q}$ は，s_i が管理するクエリ集合の集合である．

ここから，クエリの分散を決定するアルゴリズムを紹介する．まず，クエリおよびデータが存在し得る領域を $\mathbb{R}^2$ とする．$\mathbb{R}^2$ を 4 つの領域に等分割し，各領域 r に含まれるクエリ集合 Q_r に対して，$L(Q_r)$ を計算する．そして，全ての領域の中で $L(Q_r)$ が最も大きい領域を再帰的に分割する．この処理を，分割された領域の数が，サーバの数の α 倍になるまで行う．その後，$L(Q_r)$ の大きいクエリ集合から順に，その集合を管理するサーバを決定する．あるクエリ集合 Q_r を管理するサーバは，その時点で管理するクエリ集合の負荷の合計が最も小さいサーバ s_i とする．この操作を，全てのクエリ集合を管理するサーバを決定するまで繰り返す．

新たなデータ o が生成されたとき，各サーバで管理するクエリの kNN データを更新するアルゴリズムを紹介する．まず，サーバ s が管理するクエリ集合 $Q_i \in s.\mathbb{Q}$ の中で，$o.loc \in Q_i.R$ である集合を $\mathbb{Q}_c$ とする．各サーバ s は，各クエリ集合 $Q_i \in s.\mathbb{Q}$ に対して，キーワードごとにクエリ $q \in Q_i$ へのポインタを管理する転置ファイル $Q_i.I$ を保持している．よって，$Q_i \in \mathbb{Q}_c$ に対して，$Q_i.I$ を用いて，o のキーワードを含むクエリの集合 Q_c を取得し，$q \in Q_c$ の kNN データを更新する．

4 提案アルゴリズム

ベースラインアルゴリズムでは，位置情報またはキーワードのいずれかに基づいて，クエリ集合を分散した．しかし，クエリおよびデータの位置およびキーワードの分布によって，より良い分散の方法は異なる．図 3 に，クエリおよびデータの位置およびキーワードの分布の一例を示す．図 3a の場合，各クエリの C_q は小さいため，位置情報に基づいてクエリを分散したとき，全体の負荷が小さくなる．一方，全てのデータおよびクエリが同じキーワードが含んでいるため，キーワードに基づいてクエリを分散したとき，全体の負荷が大きくなる．また，図 3b の場合，各クエリの C_q が大きく，C_q が重複しているため，位置情報に基づいてクエリを分散したとき，全体の負荷が大きくなる．一方，データおよびクエリのキーワードがそれぞれ異



(a) (b)

図 3: 2 種類のデータとクエリ

なるため，キーワードに基づいてクエリを分散したとき，全体の負荷が小さくなる．よって，図 3a の場合，位置情報に基づいたクエリの分散，図 3b の場合，キーワードに基づいたクエリの分散が適している．

そこで，提案アルゴリズムでは，あるクエリの集合を分割するとき，クエリおよびデータの位置およびキーワードの分布に基づき，位置に基づく分割およびキーワードに基づく分割のうち，負荷の小さくなる分割を選択することで，全体の負荷がより小さくなるようにクエリを分割する．その後，各サーバの負荷が均一になるように，分割された各クエリ集合に含まれるクエリを管理するサーバを決定する．

4.1 位置，キーワードに基づくサーバの負荷

はじめに，位置およびキーワードに基づく負荷を定義する．まず，あるクエリ集合 Q_i に含まれるクエリ q の負荷を考える．データの位置およびキーワードの分布が，時間により変化しない，かつ各キーワードの出現確率が独立であると仮定すると，新たなデータ o が生成されたとき，$o.loc$ が R 内に含まれ，かつ $o.key$ にあるキーワード w が含まれる確率 $P(R, w)$ は，以下の式で計算される．

$$P(R, w) = \frac{|O_{R,w}|}{|O|}$$

ここで，$O_{R,w}$ は R 内に含まれ，かつ w を含むデータの集合である．このとき，Q_i に含まれるクエリ q の kNN データとなりえるデータは，R 内に含まれ，かつ $q.key$ に含まれるキーワードを一つ以上含むデータであるため，$L(q)$ は $P(R, w)$ を用いて，以下のように計算される．

$$L(q) = \sum_{w \in q.key} P(R, w)$$

また，Q_i の負荷 $L(Q_i)$ は，Q_i に含まれるクエリの負荷の合計と考えることができる．よって，$L(Q_i)$ は $L(q)$ を用いて，以下のように計算される．

$$L(Q_i) = \sum_{q \in Q_i} L(q)$$

さらに，各サーバは複数のクエリ集合を管理するため，$L(s_i)$ は，$L(Q_i)$ を用いて，以下のように計算される．

$$L(s_i) = \sum_{Q_i \in s_i.\mathbb{Q}} L(Q_i)$$

Algorithm 1 Query Set Partition

Input: Q_i,β
Output: $\mathbb{Q}$

```
1: if |Q_i| > β then
2:     Q_s ← SpacePartition(Q_i)
3:     Q ← Q ∪ Q_s
4: else
5:     Q_s ← SpacePartition(Q_i)
6:     Q_t ← TextPartition(Q_i)
7:     Calculate C_t and C_s
8:     if C_s < C_t then
9:         Q ← Q ∪ Q_s
10:    else
11:        Q ← Q ∪ Q_t
```

Algorithm 2 Server Decision

Input: $\mathbb{Q}$

```
1: while |Q| > 0 do
2:     Pop Q_i from Q
3:     while |Q_i| > 0 do
4:         Pop q from Q_i
5:         s ← arg min_{s∈S} L(s)
6:         s.Q ← s.Q ∪ {q}
```

4.2 分散のアルゴリズム

ここから，クエリの分散を決定するアルゴリズムを説明する．

4.2.1 クエリ集合の分割

あるクエリの集合 Q_i を複数の集合に分割するとき，位置情報に基づく分割およびキーワードに基づく分割のうち，より負荷の小さくなる方法で分割を行う．位置情報に基づく分割およびキーワードに基づく分割は，それぞれ以下の方法で行う．

位置情報に基づく分割． Q_i がある領域 r に含まれるとき，r を 4 つの領域 $r_i(1 \leqq i \leqq 4)$ に等分割し，それぞれの領域に含まれるクエリの集合 Q_{r_i} に分割する．

キーワードに基づく分割． Q_i に含まれるクエリ q を $L(q)$ に基づいて，4 つの集合 $Q_{t_i}(1 \leqq i \leqq 4)$ に分割する．具体的には，$L(q)$ が大きいクエリから順に，各集合に分配する．あるクエリを分配するとき，その時点で分配されたクエリの負荷の合計が最も小さい集合に，クエリを分配する．この処理を Q_i に含まれる全てのクエリに対して行う．

位置情報に基づく分割およびキーワードに基づく分割のうち，負荷の小さくなる分割方法を決定するため，それぞれの分割により得られた集合の負荷の合計を比較する．具体的には，位置情報に基づく分割によって得られた集合の負荷の合計を $C_s = \sum_{1 \leqq i \leqq 4} L(Q_{r_i})$，キーワードに基づく分割により得られた集合の負荷の合計を $C_t = \sum_{1 \leqq i \leqq 4} L(Q_{t_i})$ とする．C_s が C_t より小さければ，位置情報に基づく分割を選択し，C_t が C_s より小さければ，キーワードに基づく分割を選択する．

アルゴリズム 1 は，クエリの集合 Q_i を分割するアルゴリズムを示している．まず，キーワードに基づく分割は，Q_i に含まれるクエリを 1 つずつチェックするため，Q_i に多くのクエリが含まれるとき，分割に多大な時間がかかる．そのため，$|Q_i| > \beta$ である場合，常に SpacePartition(Q_i) を実行する（2–3 行）．ここで，SpacePartition(Q_i) は，Q_i を位置情報に基づいて分割した集合の組を返す関数である．$|Q_i| \leqq \beta$ である場合，位置情報に基づく分割およびキーワードに基づく分割を行い，それぞれによって得られた集合の負荷の合計 C_s および C_t を計算する（5–7 行）．ここで，TextPartition(Q_i) は，Q_i をキーワードに基づいて分割した集合の組を返す関数である．$C_s < C_t$ ならば，位置情報に基づく分割を選択し，$C_s \geqq C_t$ ならば，キーワードに基づく分割を選択する（8–11 行）．分割された全ての集合を $\mathbb{Q}$ とすると，$|\mathbb{Q}|$ が αm 以上になるまで，負荷の最も大きい集合 Q_i を再帰的に分割する．

4.2.2 クエリ集合を管理するサーバの決定

集合の分割後，各クエリ集合に含まれるクエリを管理するサーバを決定する．同じクエリ集合に含まれるクエリは，位置が近く，同じような出現頻度のキーワードを持つクエリが多いため，同じデータによってクエリの更新を行う可能性が高い．そのため，1 つのクエリ集合に含まれるクエリを 1 つのサーバで管理するのではなく，複数のサーバに分散させることで，各サーバの負荷が均一になるようにクエリを分散する．

アルゴリズム 2 は各クエリ集合に含まれるクエリを管理するサーバを決定するアルゴリズムを示している．$\mathbb{Q}$ は分割された全てのクエリ集合であり，負荷の降順でソートされているとし，負荷の大きい集合 Q_i から順に，管理するサーバを決定する（2 行）．Q_i に含まれるクエリが負荷の降順にソートされているとし，負荷の大きいクエリから順に管理するサーバを決定する．(4 行）．あるクエリを管理するサーバは，その時点で負荷が最も負荷が小さいサーバ s とする (5–6 行)．Q_i に含まれる全てのクエリを管理するサーバを決定するまで，この操作を行う．この処理を，全てのクエリ集合に対して行う．

4.3 kNN データの更新アルゴリズム

新たなデータ o が生成されたとき，各サーバで管理するクエリの kNN データを更新するアルゴリズムを紹介する．まず，クエリおよびデータが存在し得る領域 $\mathbb{R}^2$ を $n \times n$ の領域に等分割する．各サーバは，それぞれの領域 r_i に対して，サーバで管理するクエリ $q \in s.Q$ の C_q が r_i と重複するクエリへのポインタをキーワードごとに管理する転置ファイル $r_i.I$ を保持する．新たなデータ o が生成されたとき，$o.loc \in r_i$ となる領域の $r_i.I$ を用いて，o のキーワードを含むクエリの集合 Q_c を取得し，$q \in Q_c$ の kNN データを更新する．

5 評 価 実 験

本章では，提案アルゴリズムの性能を評価するため，以下のアルゴリズムとの比較を行った．

- ベースライン 1：キーワードに基づいてクエリの分散を行う．

表 1: データセットの詳細

ツイートの数	20,000,000
キーワードの数	2,225,654
1 つのツイートに含まれるキーワードの数の平均	5.51

表 2: パラメータの設定

パラメータ	値
サーバ数	2, 4, **8**, 16
$q.k$ の最大値，k_{max}	5, **10**, 15, 20

- ベースライン 2：位置情報に基づいてクエリの分散を行う．

5.1 セッティング

本実験は，Windows 10 Pro，3.00GHz Intel Xeon Gold 6154，および 512GB RAM を搭載した計算機で行い，全てのアルゴリズムは C++で実装した．

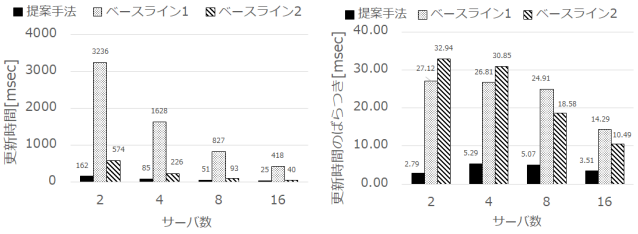
データセット．本実験では，生成されるデータとして Twitter [3] のツイートを用いた．表 1 にデータセットの詳細を示す．各クエリは，1 つのツイート中に現れる単語の中から，5 個以下の単語をランダムに選び，クエリのキーワード集合とする．

パラメータ．表 2 に，本実験で用いたパラメータを示す．太字で表されている値はデフォルトの値である．クエリの数を 1,000,000 とし，さらに，k は 1 から k_{max} の間の一様乱数とした．また，$\alpha = 20$ および $\beta = 100,000$ とした．さらに，1 度に 1,000 個のデータが発生するとする．実験は，データが 1,000,000 個発生した時点で，クエリを分散し，新たに 1,000,000 個のデータが発生するまで行う．

5.2 評 価 結 果

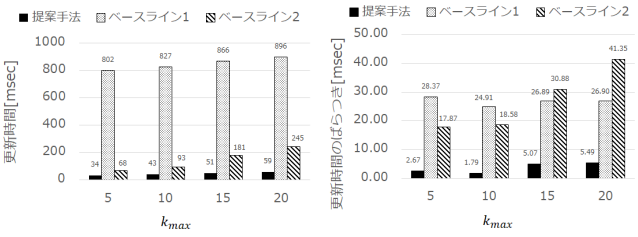
本節では，各アルゴリズムにおける平均更新時間 (1 度に発生する全てのデータに対して，kNN データの更新が完了するまでの平均時間 [msec]) およびサーバの更新時間のばらつき (各サーバで管理しているクエリの kNN データの更新が完了するまでの時間の中で最大の時間と最小の時間の差 [msec]) の結果を示す．

サーバ数の影響．図 4 にサーバ数を変えたときの結果を示す．サーバ数が増加すると，各サーバで管理するクエリの数が少なくなるため，各アルゴリズムにおいて，平均更新時間は小さくなる．また，提案アルゴリズムは，サーバ数によらず，常に平均更新時間が最も小さい．ベースライン 1 では，発生したデータのキーワードを含む全てのクエリの kNN データの更新を行うため，平均更新時間は大きくなる．ベースライン 2 では，クエリ集合の R と転置ファイルに基づいて，kNN データの更新を行うクエリを決定する．これにより，ベースライン 1 に比べて，更新を行うクエリの数が小さくなるため，平均更新時間は小さくなる．提案アルゴリズムでは，位置とキーワードに基づく負荷を考慮することにより，ベースライン 2 よりも，より正確にクエリ集合に含まれるクエリの kNN データの更新にかかる時間を見積もることができる．また，位置に基づく分割およびキーワードに基づく分割のうち，より全体の負荷が小さくなるようにクエリ集合を分割することで，クエリの更新にかかる時間が短くなる．さらに，同じクエリ集合に含まれるクエリは，位置が近く，同じような出現頻度のキーワードを持つクエリが多いため，同じデータによってクエリの更新を行う可能性が高い．そのため，1 つのクエリ集合に含まれるクエリを 1 つのサーバで管理するのではなく，複数のサーバに分散させることで，各サーバの更新時間のばらつきが小さくなり，平均更新時間も小さくなる．



(a) 平均更新時間　　**(b)** 更新時間のばらつき

図 4: サーバ数の影響



(a) 平均更新時間　　**(b)** 更新時間のばらつき

図 5: k_{max} の影響

各アルゴリズムにおいて，サーバ数が増加すると，各サーバで更新を行うクエリの数が小さくなるため，更新時間のばらつきは小さくなる．提案アルゴリズムは，各ベースラインに比べて，更新を行うクエリの数がより小さくなり，各サーバで更新を行うクエリの数の差が小さくなるため，更新時間のばらつきがより小さくなる．

k_{max} の影響．図 5 に k_{max} を変えたときの結果を示す．k_{max} が増加すると，kNN データが変化するクエリの数が増加するため，各アルゴリズムにおいて，平均更新時間が大きくなる．提案アルゴリズムは，k_{max} によらず，常に平均更新時間が最も小さい．ベースライン 2 は，k_{max} が増加すると，クエリ集合の R の範囲が大きくなり，新たに発生したデータが kNN データとなる可能性があるクエリの数が大きくなるため，平均更新時間が大きくなる．提案アルゴリズムは，位置とキーワードに基づいた負荷がより小さくなるようにクエリ集合を分割し，1 つのクエリ集合に含まれるクエリを複数のサーバに分散することで，各サーバの更新時間のばらつきが小さくなり，平均更新時間も小さくなる．

提案アルゴリズムは，k_{max} によらず，常に更新時間のばらつきが小さい．ベースライン 1 では，k_{max} によってクエリの

負荷が変化することはなく，各サーバで管理されるクエリは変化しない．したがって，各サーバで更新を行うクエリの数は変化しないため，更新時間のばらつきがほぼ一定である．ベースライン 2 では，k_{max} が増加すると，各クエリ集合の R の範囲が大きくなり，各サーバで管理されるクエリ集合の R の範囲が重複する可能性が大きくなる．したがって，あるサーバの R の範囲が重複する位置にデータが発生した場合，そのサーバで更新を行うクエリの数が大きくなるため，更新時間のばらつきが大きくなる．提案アルゴリズムでは，同じクエリ集合に含まれるクエリは，位置が近く，同じような出現頻度のキーワードを持つクエリが多いため，同じデータによってクエリの更新を行う可能性が高い．したがって，ベースライン 2 のように，1 つのクエリ集合に含まれるクエリを 1 つのサーバで管理するのではなく，複数のサーバに分散するため，更新時間のばらつきが小さくなる．

6 ま と め

近年，パブリッシュ/サブスクライブモデルに基づくアプリケーションが多く存在する．また，スマートフォンや SNS の普及により，データやクエリの数が増加しており，単一のサーバで全ての処理を行うとリアルタイム性を保持することが困難になっている．本稿では，パブリッシュ/サブスクライブモデルにおいて，複数のサーバでクエリを分散管理して処理を分担することにより，各クエリの kNN データを効率的にモニタリングする問題に取り組んだ．データおよびクエリの位置情報およびキーワードに基づいてサーバの負荷を定義し，クエリ集合を分割する際，位置情報に基づく分割およびキーワードに基づく分割のうち，全体の負荷が小さくなる分割を選択することで，各サーバの負荷を均一にしつつ，全体の負荷を小さくするようにクエリを分散するアルゴリズムを提案した．実データを用いた実験により，提案アルゴリズムの有効性を確認した．

今後の課題．データの生成によって，クエリの kNN データが更新されると，クエリの C_q の大きさが変化し，クエリの負荷が変化する．その結果，サーバの負荷が変化し，各サーバの負荷が偏ってしまう可能性がある．データの生成によって，各サーバの負荷が均一でなくなったとき，サーバが管理するクエリ集合を動的に変更し，各サーバの負荷を均一に保つアルゴリズムを提案することが今後の課題としてあげられる．

謝辞．本研究の一部は，文部科学省科学研究費補助金・基盤研究 (A)(JP26240013)，(A)(JP18H04095)，および基盤研究 (B)(T17KT0082a) の研究助成によるものである．ここに記して謝意を表す．

文　　献

[1] Aly, A.M., Mahmood, A.R., Hassan, M.S., Aref, W.G., Ouzzani, M., Elmeleegy, H., Qadah, T.: Aqwa: adaptive query workload aware partitioning of big spatial data. PVLDB 8(13), 2062–2073 (2015)

[2] Basık, F., Gedik, B., Ferhatosmanoğlu, H., Kalender, M.E.: s^3-tm : scalable streaming short text matching. The VLDB Journal 24(6), 849–866 (2015)

[3] Chen, L., Cong, G., Cao, X., Tan, K.L.: Temporal spatial-keyword top-k publish/subscribe. In: ICDE. pp. 255–266 (2015)

[4] Chen, Z., Cong, G., Zhang, Z., Fuz, T.Z., Chen, L.: Distributed publish/subscribe query processing on the spatio-textual data stream. In: ICDE. pp. 1095–1106 (2017)

[5] Choi, D.W., Pei, J., Lin, X.: Finding the minimum spatial keyword cover. In: ICDE. pp. 685–696 (2016)

[6] Eldawy, A., Mokbel, M.F.: Spatialhadoop: A mapreduce framework for spatial data. In: ICDE. pp. 1352–1363 (2015)

[7] Hu, H., Liu, Y., Li, G., Feng, J., Tan, K.L.: A location-aware publish/subscribe framework for parameterized spatio-textual subscriptions. In: ICDE. pp. 711–722 (2015)

[8] Korf, R.E.: Multi-way number partitioning. In: IJCAI. pp. 538–543 (2009)

[9] Li, G., Wang, Y., Wang, T., Feng, J.: Location-aware publish/subscribe. In: SIGKDD. pp. 802–810 (2013)

[10] Yu, A., Agarwal, P.K., Yang, J.: Subscriber assignment for wide-area content-based publish/subscribe. TKDE 24(10), 1833–1847 (2012)