EFTG: Efficient and Flexible Top-K Geo-textual Publish/Subscribe

Hong zhu[1], Hongbo Li[1, 2], Zongmin Cui[*, 3], Zhongsheng Cao[1], Meiyi Xie[1]
[1]School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, Hubei, China
[2]School of Computer Science and Information Technology, Daqing Normal University, Daqing, Heilongjiang, China
[3]School of Information Science and Technology, Jiujiang University, Jiujiang, Jiangxi, China
[e-mail: cuizm01@gmail.com]
*Corresponding author: Zongmin Cui

Received July 17, 2017; revised April 28, 2018; accepted August 8, 2018; published December 31, 2018

Abstract

With the popularity of mobile networks and smartphones, geo-textual publish/subscribe messaging has attracted wide attention. Different from the traditional publish/subscribe format, geo-textual data is published and subscribed in the form of dynamic data flow in the mobile network. The difference creates more requirements for efficiency and flexibility. However, most of the existing Top-k geo-textual publish/subscribe schemes have the following deficiencies: (1) All publications have to be scored for each subscription, which is not efficient enough. (2) A user should take time to set a threshold for each subscription, which is not flexible enough. Therefore, we propose an efficient and flexible Top-k geo-textual publish/subscribe scheme. First, our scheme groups publish and subscribe based on text classification. Thus, only a few parts of related publications should be scored for each subscription, which significantly enhances efficiency. Second, our scheme proposes an adaptive publish/subscribe matching algorithm. The algorithm does not require the user to set a threshold. It can adaptively return Top-k results to the user for each subscription, which significantly enhances flexibility. Finally, theoretical analysis and experimental evaluation verify the efficiency and effectiveness of our scheme.

Keywords: Top-k; geo-textual, publish/subscribe, efficient and flexible, text classification

This research was supported by the National Natural Science Foundation of China [Nos. 61772215 and 61762055]; and the Jiangxi Provincial Natural Science Foundation of China [Nos. 20161BAB202036 and 20181BAB202014]. Meawhhile, we gratefully acknowledge the reviewers' comments and helpful suggestions.

1. Introduction

The popularity of intelligent mobile devices equipped with GPS receivers [1] has caused widespread interest in geo-textual publish/subscribe systems [2], [3], [4], [5]. Thus geo-textual publish/subscribe is used in many mobile network scenarios, such as mobile social media [6], geo-based product recommendations [7], etc.

In a geo-textual publish/subscribe system, subscribers (i.e., users, data visitors, buyers, etc.) register their interests as geo-textual subscriptions (for example, a "clothing promotion" near the subscriber's location "Beijing Wangfujing Street") [8]. Then the system quickly and accurately sends information (such as " Nike shoes 20% discount in Wangfujing Street No. 36 clothing store") to the relevant subscribers [9]. The information is published by publishers (i.e., data owners, data providers, sellers, etc.) [10].

Unlike traditional publish/subscribe systems, geo-textual data includes a geographic description and a textual description that are continuously generated and propagated in the form of a dynamic data stream [11]. Geo-textual data has strong position sensitivity (for example, when a subscriber leaves a geographic area, the geo-related publication data is no longer sent to the subscriber) [12]. Therefore, higher efficiency and flexibility requirements have been proposed for more and more application scenarios [13].

1.1 Research Motivation

To improve the efficiency of geo-textual matching between publication and subscription, researchers have proposed many effective methods based on the index structure, such as the latest research [5] [6] [14] [15] [16]. However, most of the existing Top-k geo-textual publish/subscribe schemes have the following two deficiencies, which reduce efficiency and flexibility: (1) These schemes need to match all publication categories for each subscription, which greatly reduces the efficiency of publish/subscribe matching. For example, if the subscriber is interested only in "costumes," it is clear that there is no need to calculate the geo-textual similarity between the "costumes" subscription and "non-costumes" publications. (2) These schemes need the subscriber to set the appropriate threshold for publish/subscribe matching. First, this reduces the friendliness of the system (it is often difficult for ordinary subscribers to understand and set the appropriate thresholds). Second, this makes it difficult for the calculated number of publications, based on the threshold, to be exactly equal to the number of subscriptions (i.e., k). Thus, the flexibility of the system is reduced.

Here are some examples that explicitly illustrate the second deficiency:

(1) Threshold is too large. A subscriber only wants the system to return a publication with the highest geo-textual matching score (that is, return the Top-1 publication), but the system needs the subscriber to set a threshold between 0 and 1. It is very difficult to set an appropriate threshold. If the subscriber sets a threshold that is too large, the system calculates 10,000 results based on the threshold. Then the system returns the publication with the highest score to the subscriber. Obviously, in this process, the extra 9,999 intermediate results waste the valuable computing resources of the system.

(2) Threshold is too small. A subscriber wants to get the Top-1000 publications, but the subscriber sets a threshold that is too low. The system calculates only five results based on the threshold, which misses the other 995 results. Obviously, in this process, the system misses many results. The missed results reduce the system's accuracy, which cannot meet the subscriber’s actual requirements.

In short, unlike the k value, determining the threshold is not intuitive. This makes it very difficult for an ordinary subscriber to set an appropriate threshold. Thus, the threshold reduces the friendliness and flexibility of the publish/subscribe system.

1.2 Our Contribution

To solve the above deficiencies, we propose an efficient and flexible Top-k geo-textual publish/subscribe scheme, which is called the Efficient and Flexible Top-k Geo-textual (EFTG) publish/subscribe scheme. Our scheme begins by classifying the publication and subscription. Only the same class of publications should be scored for each subscription, which significantly enhances the publish/subscribe matching efficiency. Our scheme also provides an adaptive geo-textual publish/subscribe matching algorithm. The algorithm does not require the subscriber to set a threshold. The system can adaptively obtain Top-k results for each subscription. Thus, we improve the system's friendliness and flexibility. Specifically, our contributions are as follows:

(1) We propose a publish/subscribe classification model that enhances matching efficiency.

(2) We propose an adaptive geo-textual publish/subscribe matching method that enhances the friendliness and flexibility of the system.

(3) We propose a lightweight update algorithm that enhances the publication update efficiency for many dynamic scenarios.

Section 2 describes the previous research in this area. Section 3 introduces the publish/subscribe classification model. Section 4 presents the adaptive geo-textual publish/subscribe matching method without a threshold. Section 5 addresses the publication update problem. Section 6 presents an extensive experimental evaluation of our scheme compared with two existing schemes and Section 7 provides our conclusions.

2. Related Work

2.1 Publish/subscribe system

Users register their interests as a running query in the publish/subscribe system [18]. When the information matches the user's interest, the streaming information is sent to the user.

Many works, for example [19], involve distributed content-based publish/subscribe systems on a wide area network. Their goal is to confirm subscribers' optimal allocation of message agents [20] through some performance criteria, such as latency. Other works, such as [21], consider deploying a publish/subscribe system on a peer-to-peer network, which focuses on minimizing communication costs.

However, many existing publish/subscribe schemes [22] [23] do not consider geographic information [24] [25]. Our problems focus on the use and expansion of geographic information.

2.2 Spatial keyword query

Spatial keyword query has been extensively studied in academia. Some research retrieves a set of geographical text objects based on Boolean matching [26] [27]. Other research is based on a scoring function with a federated spatial index (such as R-Tree, Quadtree) and a text index (such as a reverse file) [13] [28].

Recent studies build publish/subscribe systems with spatial text data streams on a centralized server [6]. They focus on developing new indexes to speed up the match speed between spatial text objects and keyword continuous keyword queries [29].

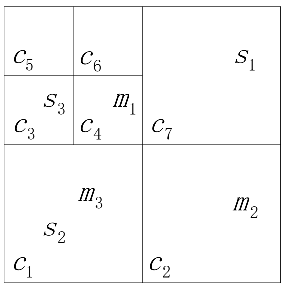
The spatial keyword query process is summarized in [7], and references [30] and [31] discuss the expansion of spatial keywords. We note that a spatial keyword query is actually a point-to-point or snapshot query. Our focus is on textual similarity matching queries.

2.3 Geo-textual query

Recently, the geo-textual publish/subscribe system has been studied in several works (for example [3] [4] [6] [7] [9] [10] [12]), but these studies have not combined geographic and textual similarity. Therefore, it is not possible to quantify the geo-textual similarity, which is essentially different from our work. At present, our work is close to the studies of the Conditional Influence Region-Based Quadtree (CIQ) [6] and Individual and Group Pruning Techniques (IGPT) [5] [14] [15] methods. These studies also support Top-k geo-textual publish/subscribe.

In CIQ, a Quadtree is used to partition the entire space. A subscription is assigned to a certain number of covering cells, which form a separate partition of the entire space. For new information, CIQ traverses all the inserted files by using corresponding cells. The cells are infiltrated by the information position in the DAAT (Document-At-A-Time) paradigm. Meanwhile, CIQ finds all subscriptions whose textual similarities are above the preset text bounds. The text paradigm in CIQ cannot integrate some similarity search techniques based on thresholds. CIQ indexes each subscription to multiple cells by the pre-computing text bounds. Because the coverage of each subscription is unlikely to be too large, CIQ's harvest is very limited. Therefore, CIQ causes high memory consumption. Therefore, CIQ is less efficient and practical than our scheme.

Figure 1 provides an index example of IGPT. IGPT assumes that all subscriptions have the same cell coverage. A text boundary needs to be calculated in advance for each subscription. A text boundary is a cell. The left side of **Fig. 1** shows the text bounds (i.e., c_2, c_7, etc.). An inserted file is created to manage the subscriptions assigned to each cell. The inserted file is sorted based on the subscription ID.



Subscription	Keywords	α	KScore
s_1	$w_1$0. 4, $w_2$0. 3, $w_3$0. 2, $w_4$0. 1	0. 6	0. 7
s_2	$w_1$0. 7, $w_2$0. 5, $w_3$0. 2	0. 5	0. 6
s_3	$w_1$0. 5, $w_2$0. 4, $w_3$0. 3, $w_4$0. 1	0. 7	0. 67

Fig. 1. An index example of IGPT

For new information (e.g., m_1 in **Fig. 1**), IGPT's index structure is designed for the TAAT (Term-At-A-Time) paradigm. IGPT integrates advanced pruning techniques based on thresholds, and many subscriptions are therefore eliminated. IGPT uses the on-the-fly spatial boundary calculation strategy. Each subscription is assigned to a single cell with good spatial granularity, which makes IGPT more efficient than CIQ. However, IGPT does not classify information, and is therefore inefficient in matching different kinds of information. On the other hand, the threshold limits the further promotion of the publish/subscribe system. Our scheme solves these two problems, markedly improves efficiency, and reduces the difficulty of using the system.

2.4 Dynamic Top-k maintenance

There is a key issue in Top-k maintenance of query results in various publish/subscribe systems. When an old publication expires, the Top-k results must be re-evaluated for the affected persistent queries (i.e., subscriptions in this paper). If we re-evaluate all the Top-k results, the cost is expensive. On the other hand, it is not feasible to store all information and scores in the cache to avoid re-evaluation.

Udomlamlert et al. [32] describe a Kmax method to balance re-evaluation cost and cache consumption. Instead of maintaining exactly the Top-k results, they maintain top-k' results. k' is between k and a parameter kmax. However, the Kmax method contains redundant elements.

The K-skyband method [33] was proposed to eliminate redundancy. Because it is very expensive to maintain complete K-skyband results for each individual query, only scores that greater than or equal to the K-th score are maintained. The K-th score is the highest score in the most recent Top-k re-evaluation [34]. We have observed that this setting corresponds to point-to-point mode, so some of the functionality may not be satisfied in practice.

Pripuzic et al. [35] propose a probabilistic K-skyband method to discard the results that are unlikely to become Top-k results. Thus, they save memory consumption and improve system efficiency. However, because they employ probabilistic methods, some Top-k results may be lost.

IGPT [5] [14] [15] presents a cost-based K-skyband technology to carefully determine the cache consumption.

However, none of above methods classifies information. Thus, they have to match different kinds of information during dynamic Top-k maintenance, which results in some redundant computing and useless information. Our scheme is more efficient than existing methods in dynamic Top-k maintenance.

3. Publish/subscribe Classification

3.1 System Structure

The system structure of the EFTG scheme is shown in **Fig. 2.**

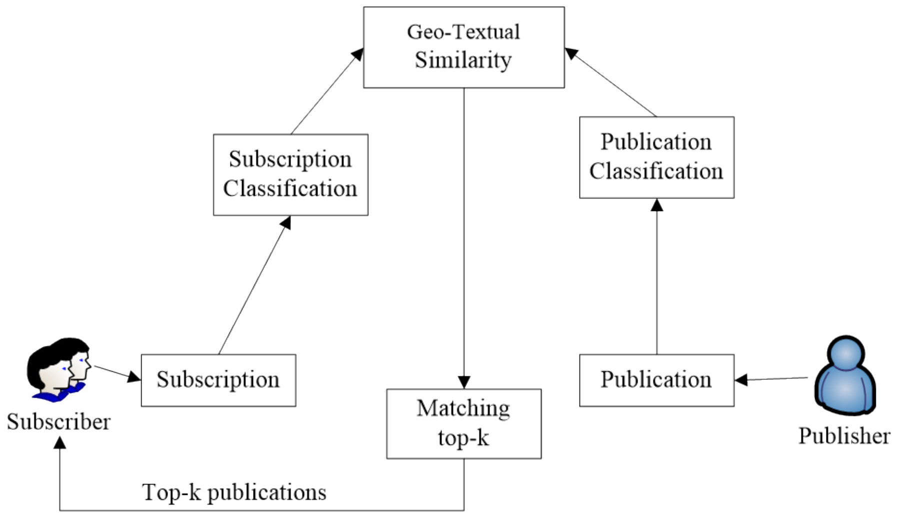


Fig. 2. System structure of the EFTG scheme

Publishers upload their geo-textual publications to the system (Internet, network, iPad, mobile phone, cloud, etc.). Then the system classifies the uploaded publications. Meanwhile, subscribers register their interests as geo-textual subscriptions in the system. In the same way, the system also classifies the uploaded subscriptions. Only when a publication and a subscription are of the same class, should the publication be scored for the subscription as a geo-textual similarity. Finally, the system returns the highest scored Top-k results to the subscriber based on geo-textual similarity.

3.2 Geo-textual Similarity

The following definitions of the EFTG scheme components help us accurately and formally define our problem.

Definition 1. (**Subscription**). A subscription is denoted as $s = (s_C, s_T, s_G, \delta, k)$. (1) In this equation, s_C denotes subscription s's class. (2) s_T denotes s's text description, which is composed of a set of keywords { t_1, t_2, ……, $t_{|s_T|}$ }. Each keyword t_i is associated with a weight $w(t_i)$. $w(t_i)$ is the TF-IDF (Term Frequency–Inverse Document Frequency) [17] weight of keyword t_i in s. (3) s_G denotes s's geographic description, which is composed of latitude and longitude. (4) δ is a preference parameter. If δ >0.5, the text description is more important than the geographic description. (5) k is the number of returned results of subscription s.

Definition 2. (**Publication**). A publication is denoted as $p = (p_C, p_T, p_G)$. (1) In this equation, p_C denotes publication p's class. (2) p_T denotes p's text description, which is composed of a set of keywords { t_1, t_2, ……, $t_{|p_T|}$ }. Each keyword t_i is associated with a weight $w(t_i)$. $w(t_i)$ is the TF-IDF weight of keyword t_i in p. (3) p_G denotes p's geographic description, which is composed of latitude and longitude.

Example 1. Table 1 shows an example with a set of subscriptions and a set of publications.

Table 1. An example of a publish/subscribe system

Subscription		**Publication**	
Subscription ID	Content	Publication ID	Content
s_0	(clothes, s_{T_0}, s_{G_0}, 0.7, 3)	p_0	(clothes, p_{T_0}, p_{G_0})
s_1	(food, s_{T_1}, s_{G_1}, 0.2, 2)	p_1	(bag, p_{T_1}, p_{G_1})
s_2	(Phone, s_{T_2}, s_{G_2}, 0.9, 1)	p_2	(food, p_{T_2}, p_{G_2})
s_3	(bag, s_{T_3}, s_{G_3}, 0.5, 3)	p_3	(Phone, p_{T_3}, p_{G_3})
s_4	(Phone, s_{T_4}, s_{G_4}, 0.6, 5)	p_4	(clothes, p_{T_4}, p_{G_4})
s_5	(Jewelry, s_{T_5}, s_{G_5}, 0.8, 4)	p_5	(food, p_{T_5}, p_{G_5})
s_6	(Skin, s_{T_6}, s_{G_6}, 0.6, 1)	p_6	(Jewelry, p_{T_6}, p_{G_6})
s_7	(Jewelry, s_{T_7}, s_{G_7}, 0.5, 2)	p_7	(food, p_{T_7}, p_{G_7})

Geo-textual publish/subscribe systems need to return each publication to related subscribers in a timely manner. Thus, we quantitatively define the similarity between subscriptions and publications as follows.

Definition 3. (**Geo-similarity**). The geo-similarity between a subscription s and a publication p is defined in Formula (1).

$$GS(s,p) = \max(0, 1 - \frac{DIST(s_G, p_G)}{D_{\max}}) \tag{1}$$

In Formula (1), $DIST(s_G, p_G)$ denotes the Euclidean distance between s_G and p_G. $D_{\max}$ denotes the max Euclidean distance that the subscriber can tolerate (a publication that exceeds the distance will not be related to the subscription).

Definition 4. (**Textual-similarity**). The textual-similarity between a subscription s and a publication p is defined in Formula (2).

$$TS(s,p) = \sum_{t_i \in s_T \cap p_T} w(s_T.t_i) \times w(p_T.t_i) \tag{2}$$

Definition 5. (**Geo-textual-similarity**). The geo-textual similarity between a subscription s and a publication p is defined in Formula (3).

$$GTS(s,p) = \delta \times TS(s,p) + (1-\delta) \times GS(s,p) \tag{3}$$

In Formula (3), δ is a preference parameter. δ is used to coordinate geo-similarity and textual-similarity. If δ >0.5, textual similarity is more important than geo-similarity.

To facilitate the next description, we use S to represent all subscriptions in the system. P represents all publications and $GTS(s, P)$ represents a set of geo-textual similarities between a subscription s and all publications P in the system.

3.3 Similarity Algorithm

There is no classification in most of the existing methods [5] [6] [14] [15]. Thus the computational complexity of similarity is $O(|S| \times |P|)$. To reduce the computational complexity, we propose a publish/subscribe classification model that classifies the publication and subscription.

```
Algorithm 1: $Similarity$
Input: $s$, $P$
Output: $GTS(s, P)$
1:    $GTS(s, P) := \phi$
2:    For all $p \in P$ do
3:      If $s_C == p_C$ then
4:        $GS(s, p) := \max(0, 1 - \frac{DIST(s_G, p_G)}{D_{\max}})$
5:        $TS(s, p) := \sum_{t_i \in s_T \cap p_T} w(s_T.t_i) \times w(p_T.t_i)$
6:        $GTS(s, p) := \delta \times TS(s, p) + (1 - \delta) \times GS(s, p)$
7:        $GTS(s, P) := GTS(s, P) \cup GTS(s, p)$
8:      End if
9:    End for
10:   Return($GTS(s, P)$)
```

Algorithm 1, $Similarity$, regulates the computing process of similarity based on our classification model. The algorithm takes a subscription s and all publications P as input. Meanwhile, the algorithm takes the geo-textual similarity set $GTS(s, P)$ as output.

First, the algorithm checks the class of subscription s and publication p (Step 3). When, and only when, s and p are of the same class, we calculate geo-textual similarity $GTS(s, p)$ between s and p (Steps 4-6). Finally, the algorithm adds all related geo-textual similarities to $GTS(s, P)$ (Step 7).

Example 2. Based on **Example 1**, the system runs $Similarity(s_1, P)$ to calculate geo-textual similarity set $GTS(s_1, P)$ between subscription s_1 and all publications P in the system. The computing result $GTS(s_1, P)$ is shown in **Table 2**.

Table 2. An example of running Algorithm 1

Geo-textual-similarity	Value
$GTS(s_1, p_2)$	0.5
$GTS(s_1, p_5)$	0.6
$GTS(s_1, p_7)$	0.9

Subscriber s_1 wants to subscribe to information about food. However, publisher p_0 sells clothes. Thus, it is clearly impossible for subscriber s_1 to subscribe to the information from p_0. That is, there is no need to calculate the similarity between s_1 and p_0. $\{p_2, p_5, p_7\}$ and s_1 are of the same class. Therefore, we only need to calculate geo-textual similarities between s_1 and $\{p_2, p_5, p_7\}$. We assume that the calculated results are 0.5, 0.6, and 0.9, respectively.

Because Algorithm 1 uses a publish/subscribe classification model, a huge number of geo-textual similarities do not need to be calculated. Moreover, to simplify the above description, we use only a classification, but in practice, a set of classifications is often used.

For example, the Level 1 classification is **food**. The Level 2 classifications of **food** include snack, **pastry**, fruit, vegetable, beverage, supplement, etc. The Level 3 classifications of **pastry** include traditional pastry, western-style pastry, biscuit, puffed food, **mooncake**, and so on. There are Level 4 classifications of **mooncake**, etc. Therefore, the computational complexity of similarity is $O(|S| \times \frac{|P|}{m^n})$ in the EFTG scheme, where n is the level number and m is the classification number in each level.

4. Geo-textual Publish/subscribe Matching

Algorithm 2, $Matching$, regulates our geo-textual publish/subscribe matching process, which takes a subscription s and the geo-textual similarity set $GTS(s, P)$ as input. The algorithm takes the matching Top-k results R_k as output.

First, Algorithm 2 selects the publication R, which has the max geo-textual similarity in $GTS(s, P)$ (Step 3). Second, it adds R to result set R_s (Step 4). Third, it removes R from $GTS(s, P)$ (Step 5). Finally, only the Top-k results should be returned to the subscriber (Step 6).

```
Algorithm 2: $Matching$
Input: $s$, $GTS(s, P)$
Output: $R_s$
1:    $R_s := \phi$
2:    While k$\neq$0
3:        Select the publication $R$ which has the max geo-textual-similarity in $GTS(s, P)$
4:        $R_s := R_s \cup R$
5:        $GTS(s, P) := GTS(s, P) / R$
6:        k--
7:    End while
8:    Return($R_s$)
```

Example 3. Based on **Example 2**, the system runs $Matching(s_1, GTS(s_1, P))$ to get Top-k results R_{s_1} ={p_7, p_5}.

From **Table 1**, we find that s_1'k=2. That is, we need to return two top score results for s_1. Because $GTS(s_1, p_7)$=0.9 is the max geo-textual similarity in $GTS(s_1, P)$, R_{s_1} ={p_7}, we remove $GTS(s_1, p_7)$ from $GTS(s_1, P)$. Because $GTS(s_1, p_5)$=0.6 is the max geo-textual similarity in the new $GTS(s_1, P)$, R_{s_1} ={ p_7 , p_5 }, we remove $GTS(s_1, p_5)$ from $GTS(s_1, P)$. That is, $GTS(s_1, P)$={$GTS(s_1, p_2)$=0.5}.

The advantages of our matching method are as follows:

(1) Publication is only scored for the same class subscription, which enhances the matching efficiency.

(2) The user does not need to set the threshold, which simplifies user operation and improves the practicability of the system.

(3) We only need to find the highest score of the k matching results without sorting all scores, which decreases the redundancy computation.

From analyzing Algorithm 2, we find that the publication has been classified. Thus our computational complexity of matching is $O(k \times \frac{|P|}{m^n})$, where n is the level number and m is the classification number in each level.

CIQ [6] first calculates the candidate results based on the text bound. Then CIQ prunes and sorts the candidate results. Finally, CIQ returns the highest score of the k results to the subscriber. Thus the computational complexity of matching in CIQ is $O(k\times|P|\times\log_2|P|)$. IGPT [5] [14] [15] uses a more advanced and efficient pruning technique than CIQ. Therefore, the computational complexity of matching in IGPT is $O(k\times|P|)$. Obviously, our method is more efficient than CIQ and IGPT.

5. Update

```
Algorithm 3: $Update$
Input: $s$, $GTS(s, P)$, $R_s$, $ot$, $p$
Output: $R_s$', $GTS(s, P)$'
1:  If $ot$ =="delete" then
2:    If $p \in GTS(s, P)$ then
3:      $GTS(s, P)$':= $GTS(s, P)/GTS(s, p)$
4:      $R_s$':=$R_s$
5:    End if
6:    If $p \in R_s$ then
7:      $R_s$':=$R_s/p$
8:      Select $p_i \in GTS(s, P)|GTS(s, p_i)$ has the highest score in $GTS(s, P)$
9:      $GTS(s, P)$':= $GTS(s, P)/GTS(s, p_i)$
10:     $R_s$':=$R_s$'$\cup p_i$
11:   End if
12: Else
13:   If $s_C == p_C$ then
```

```
14:      $GS(s, p) := \max(0, 1 - \frac{DIST(s_G, p_G)}{D_{\max}})$
15:      $TS(s, p) := \sum_{t_i \in s_T \cap p_T} w(s_T.t_i) \times w(p_T.t_i)$
16:      $GTS(s, p) := \delta \times TS(s, p) + (1 - \delta) \times GS(s, p)$
17:      Select $p_j \in R_s \mid GTS(s, p_j)$ has the lowest score in $R_s$
18:      If $GTS(s, p_j) < GTS(s, p)$ then
19:        $R_s' := R_s / p_j$
20:        $R_s' := R_s' \cup p$
21:        $GTS(s, P)' := GTS(s, P)' \cup GTS(s, p_j)$
22:        $GTS(s, P)' := GTS(s, P)' / GTS(s, p)$
23:      Else
24:        $GTS(s, P)' := GTS(s, P) \cup GTS(s, p)$
25:        $R_s' := R_s$
26:      End if
27:    End if
28:  End if
29:  Return($R_s'$, $GTS(s, P)'$)
```

Currently, most of the publish/subscribe systems require dynamic maintenance. That is, when a new publication is created or an old publication expires, the returned results should be updated in a timely manner. For this purpose, Algorithm 3, $Update$, regulates the publication update process.

The algorithm takes a subscription s, the old geo-textual similarity set $GTS(s, P)$, the old returned result set R_s, an update operator ot (which has two operations: delete and add), and an updated publication p as input. The algorithm takes the new returned result set R_s' and the new geo-textual similarity set $GTS(s, P)'$ as output.

When an old publication p expires and p is in the old geo-textual similarity set $GTS(s, P)$, p is useless. Thus, the algorithm removes p from $GTS(s, P)$ (Step 3). In this case, the result set does not need to be updated (Step 4). If p is not in $GTS(s, P)$, p may be in old result set R_s (Step 6). In this case, we remove p from R_s (Step 7). That is, the new result set R_s' only has k-1 results. Therefore, we have to add a publication p_i (which has the highest score in $GTS(s, P)$) to R_s' (Steps 8-10).

When a new publication p comes in and p and s are of the same class, we calculate the geo-textual similarity $GTS(s, p)$ between s and p (Steps 13-16). In this case, we have to select p_j (which has the lowest score in R_s) from R_s (Step 17). If new publication p has higher geo-textual similarity than p_j, we have to replace p_j with p (Steps 18-22).

Otherwise, we only need to update geo-textual similarity set $GTS(s, P)$ without updating any returned result (Steps 23-25).

Example 4. Based on **Example 3**, an old publication p_5 expires. Then a new publication p_8 comes in. Thus, the system runs $Update(s_1, GTS(s_1, P), R_{s_1}, delete, p_5)$ and $Update(s_1, GTS(s_1, P), R_{s_1}, add, p_8)$ successively. Finally, a new result set R_{s_1}={ p_7, p_2 } and a new geo-textual similarity set $GTS(s_1, P)$={ $GTS(s_1, p_8)$=0.1}.

(1) Algorithm 3 runs $Update(s_1, GTS(s_1, P), R_{s_1}, delete, p_5)$. From **Example 3**, we find that old $GTS(s_1, P)$ ={ $GTS(s_1, p_2)$ =0.5} and old R_{s_1} ={ p_7 , p_5 }. Obviously, $p_5 \notin GTS(s_1, P)$ and $p_5 \in R_{s_1}$. Thus, R_{s_1} ={ p_7 } (Step 7). Because $GTS(s_1, P)$ has only one geo-textual similarity $GTS(s_1, p_2)$, $GTS(s_1, P)=\phi$ (Step 9) and R_{s_1} ={ p_7, p_2 } (Step 10).

(2) Algorithm 3 runs $Update(s_1, \phi, R_{s_1}, add, p_8)$. We assume that p_8 and s_1 are of the same class and $GTS(s_1, p_8)$=0.1. Because $GTS(s_1, p_8) < GTS(s_1, p_2)$, we only need to update geo-textual similarity set $GTS(s_1, P)$={ $GTS(s_1, p_8)$ =0.1} without updating any returned result (Steps 23-25).

When a publication is updated, CIQ [6] needs to update all related subscriptions. Thus the computational complexity of the update is $O(|S| \times |P| \times \log_2 |P|)$. Based on a novel K-skyband cost model, IGPT [5] [14] [15] finds the highest score result for each related subscription. Thus the computational complexity of the update is $O(|S| \times |P|)$.

In the worst case, our EFTG scheme needs to compare the updated publication to k existing publications about geo-textual similarity. Therefore, the computational complexity of the update in our scheme is $O(k \times |S| \times \frac{|P|}{m^n})$. Because $k << |P|$, our scheme is more efficient than CIQ and IGPT.

Table 3 shows a comparison of the computational complexity of the EFTG, IGPT, and CIQ schemes. In **Table 3,** $|S|$ is the number of subscriptions, $|P|$ is the number of publications, n is the level number, and m is the classification number in each level.

Table 3. The comparison of computational complexity

	EFTG	IGPT[5] [14] [15]	CIQ[6]
Similarity	$O(\|S\| \times \frac{\|P\|}{m^n})$	$O(\|S\| \times \|P\|)$	$O(\|S\| \times \|P\|)$
Matching	$O(k \times \frac{\|P\|}{m^n})$	$O(k \times \|P\|)$	$O(k \times \|P\| \times \log_2 \|P\|)$
Update	$O(k \times \|S\| \times \frac{\|P\|}{m^n})$	$O(\|S\| \times \|P\|)$	$O(\|S\| \times \|P\| \times \log_2 \|P\|)$

6. Experiment Evaluation

In this section, we verify the efficiency and flexibility of our EFTG scheme through extensive experiments. To this end, we compare EFTG with IGPT [5] [14] [15] and CIQ [6].

6.1 Experimental Setup

Each procedure is programmed using Visual C++ 6.0. Two computers are used in our experiments. One is used as the system server, whereas the other is for subscribers and publishers. They both have a 3.4-GHz dual-core CPU and 32 GB of memory. Following the general setting of traditional publish/subscribe systems [6] [14], we assume that the index is stored in server memory to support real-time response.

Publish/subscribe datasets are randomly generated by the system. All geo-textual similarities are randomly generated in the interval [0, 1]. During each comparison, the EFTG, IGPT, and CIQ schemes always have the same dataset.

In our system, there are between 10 and 50 keywords. Each keyword follows the TF-IDF weight [17]. The k value (i.e., the number of results returned to the subscriber) is between 2 and 10. The number of subscriptions |S| is between 10 M (million) and 50 M. The number of publications |P| is between 2 M and 10 M. The details are shown in **Table 4**.

Table 4. Experimental parameters

Keyword number	10, 20, 30, 40, 50
k	2, 4, 6, 8, 10
|S|	10M, 20M, 30M, 40M, 50M
|P|	2M, 4M, 6M, 8M, 10M

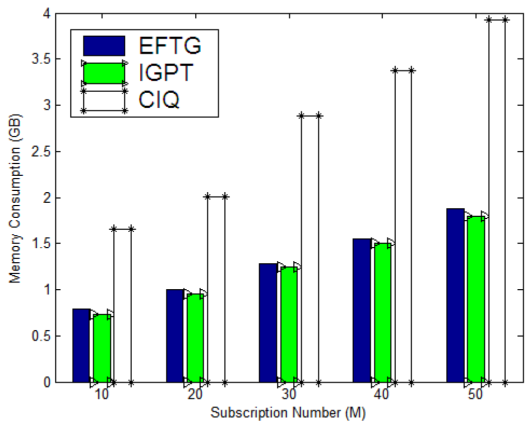
6.2 Memory Consumption

Because the index is stored in memory, the first experiment tests the memory consumption of the EFTG, IGPT, and CIQ schemes. There are two parameters that affect memory consumption—the number of subscriptions and the number of keywords.

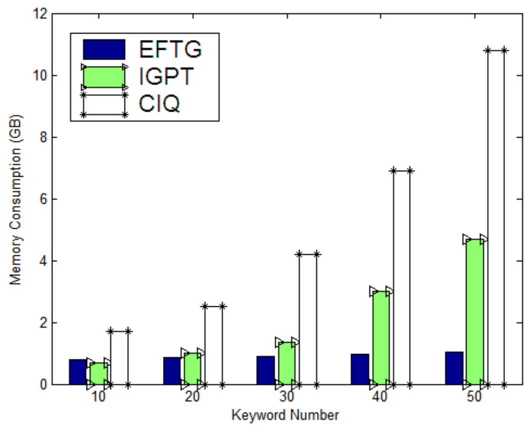
The experimental results are shown in **Fig. 3**, where the Y-axis denotes memory consumption in GB. Meanwhile, the X-axes of **Fig. 3(a)** and **3(b)** denote the number of subscriptions |S| (whose unit is M) and the number of keywords, respectively.

(1) **Fig. 3(a)**. On one hand, we remove the threshold from IGPT, which decreases memory consumption, and we add classification to the publish/subscribe system, which increases memory consumption. Therefore, from **Fig. 3(a)**, we can see that the memory consumption of IGPT is slightly less than the EFTG scheme. On the other hand, IGPT uses more advanced pruning techniques than CIQ, which removes a large number of redundant subscriptions. The memory consumption of IGPT is only 49.5% of CIQ. Therefore, CIQ's memory consumption is much higher than the EFTG scheme.

(2) **Fig. 3(b)**. When the number of keywords increases from 10 to 50, the memory consumption of IGPT and CIQ increases dramatically. However, the increasing amplitude of the EFTG scheme is lower than IGPT and CIQ. This is because EFTG has a lower threshold than IGPT. There is no index associated with the threshold. Thus, memory consumption is greatly reduced. Meanwhile, the coverage cell of each subscription in CIQ is unlikely to be too large. The memory capacity of CIQ’s cell is very limited. Therefore, CIQ experiences the highest memory consumption.



(a) Subscription number



(b) Keyword number

Fig. 3. Memory consumption comparison

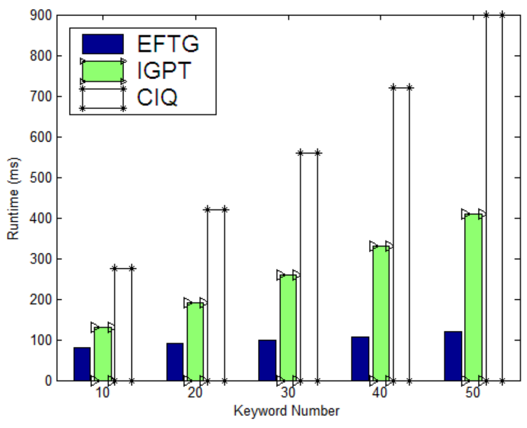
6.3 Similarity

From **Table 3**, we can see that the geo-textual similarity is related to the number of subscriptions |S| and the number of publications |P|. However in fact, the geo-textual similarity of our scheme and existing methods is also related to the number of keywords. **Table 3** omits this parameter to highlight the difference between our scheme and existing schemes. However, in actual use, we have to consider this parameter. Thus, the experimental results are shown in **Fig. 4**, where the Y-axis denotes runtime in milliseconds (ms). The X-axes of **Fig. 4(a), 4(b)** and **4(c)** denote the number of keywords, the number of subscriptions |S|, and the number of publications |P|, respectively.

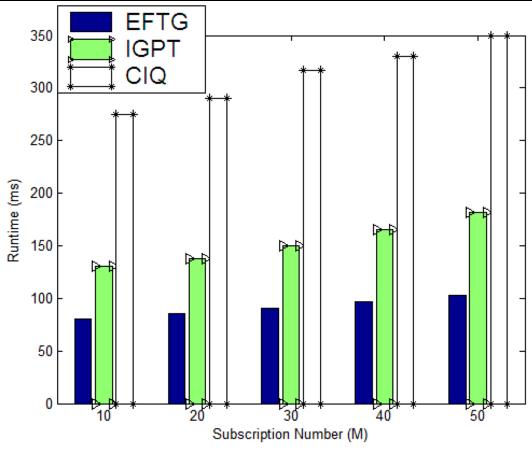
(1) **Fig. 4(a)**. When the number of keywords increases from 10 to 50, CIQ pays the highest cost for the similarity. The runtime of EFTG is less than CIQ and IGPT. The EFTG scheme removes the threshold and classifies subscriptions. Thus, the runtime increase in EFTG is much slower than IGPT and CIQ. CIQ uses a Quadtree for the entire geo-textual partition. Thus, CIQ has the maximum runtime.

(2) **Fig. 4(b)**. When the number of subscriptions increases from 10 M to 50 M, the runtime increases slowly in the three schemes. This means that the three schemes are all suitable for scenarios with a large number of subscriptions. However, we classify the publication. Thus, the EFTG scheme has the best performance.

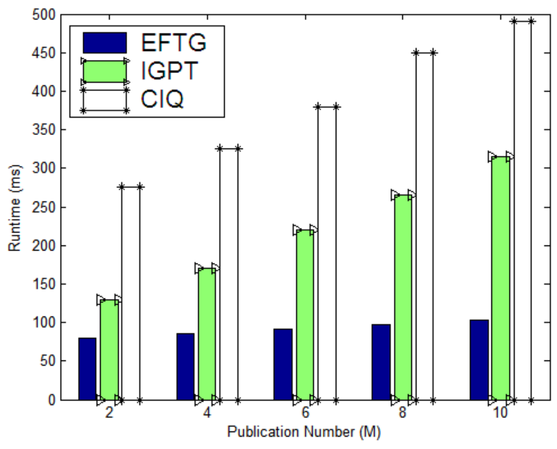
(3) **Fig. 4(c)**. Because IGPT and CIQ do not classify publications, each publication must be scored for all subscriptions by computing geo-textual similarities. Obviously, there are many redundant calculations in IGPT and CIQ. Therefore, when the number of publications increase from 2 M to 10 M, the EFTG scheme experiences the minimum runtime.



(a) Keyword number



(b) Subscription number



(c) Publication number

Fig. 4. Similarity comparison

6.4 Matching efficiency

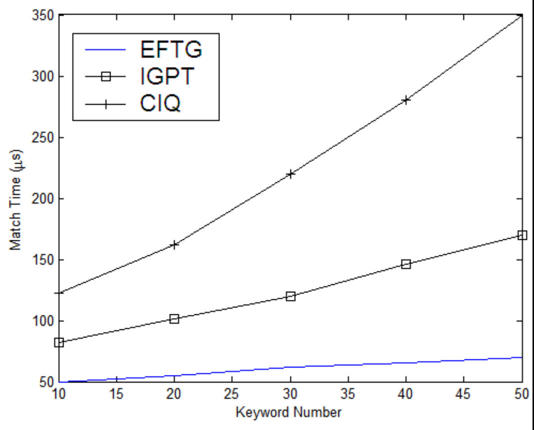
Matching efficiency is the main performance criterion for a publish/subscribe system. Faster matching speed brings faster query speed for the user. Therefore, matching efficiency is the key performance parameter of the whole system. In our experiments, match time is the average runtime of a subscriber querying a set of publications. The match time is associated with the number of keywords, k values, and publications. The experimental results are shown in **Fig. 5**, where the Y-axis denotes the match time, whose unit is the microsecond (μs). The X-axes of **Fig. 5(a), 5(b)** and **5(c)** denote the number of keywords, k values (i.e., the number of returned results), and the number of publications |P|, respectively.

(1) **Fig. 5(a)**. The EFTG matching number is exactly k (i.e., the number of returned results). However, the matching numbers of IGPT and CIQ are likely to exceed k. Thus, when the number of keywords increases from 10 to 50, the performance of EFTG is significantly higher than IGPT and CIQ. IGPT trims individual subscriptions more efficiently than CIQ based on location-aware prefix filtering technology. Therefore, CIQ has the lowest performance.

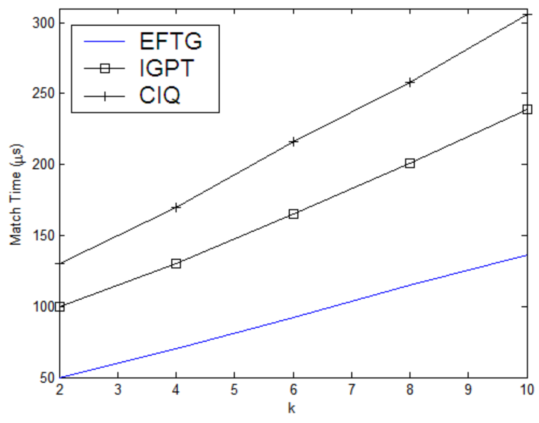
(2) **Fig. 5(b)**. When the k value increases from 2 to 10, there are significantly fewer matches for the EFTG scheme than for IGPT and CIQ. Meanwhile, CIQ faces a more complex cutting

operation than EFTG and IGPT. Therefore, EFTG has the best performance.

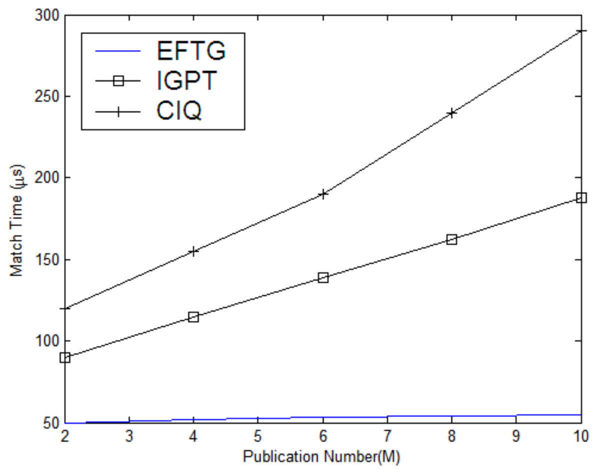
(3) **Fig. 5(c)**. EFTG classifies the publication, which reduces the number of matches. Therefore, the EFTG match time increases much more slowly than the match time for IGPT and CIQ. IGPT adopts an index structure based on the TAAT paradigm, which is more complex than EFTG. This requires more match time than the EFTG scheme. CIQ traverses all files that were inserted by corresponding cells. The cell is infiltrated by information location under the DAAT paradigm. Thus, when the number of publications increases, the match time increases sharply. Therefore, when the number of publications increases from 2 M to 10 M, EFTG is more efficient than IGPT and CIQ.



(a) Keyword number



(b) k



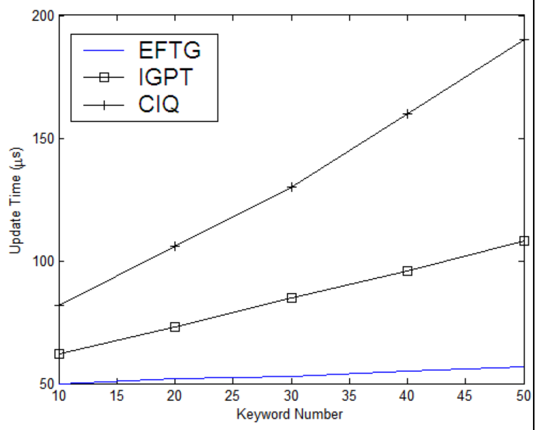
(c) Publication number

Fig. 5. Matching comparison

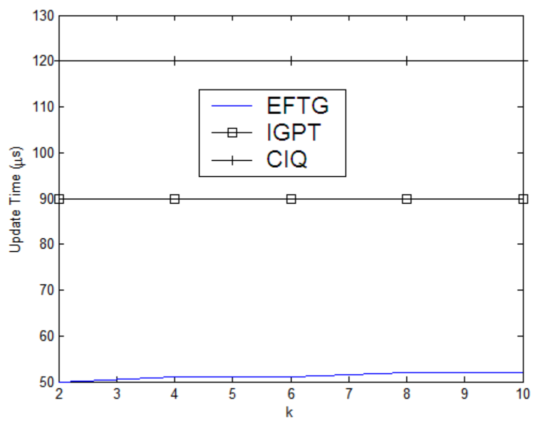
6.5 Update

As you can see from **Table 3**, the publication update is related to the number of keywords, k values, subscriptions, and publications. The experimental results are shown in **Fig. 6**. In **Fig. 6**, the type of update operation is random. That is, we randomly delete old publications or add new publications. The experimental results are the average of 100 random results.

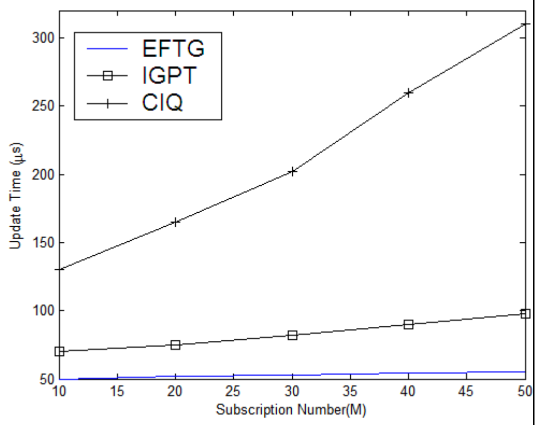
The Y-axis of **Fig. 6** denotes the runtime of updating a publication, whose unit is microseconds (μs). The X-axes of Fig.s 6(a), 6(b), 6(c), and 6(d) denote the number of keywords, k values, subscriptions, and publications, respectively.



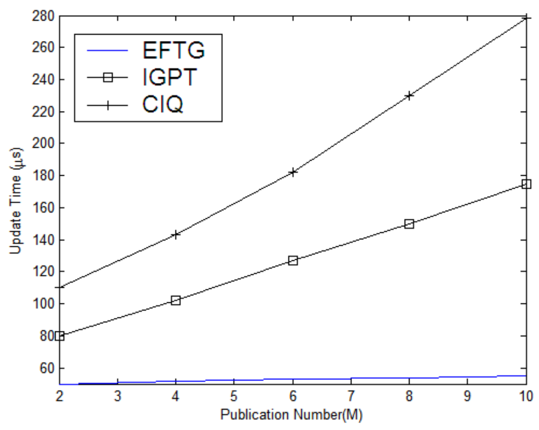
(a) Keyword number



(b) k



(c) Subscription number



(d) Publication number

Fig. 6. Publication update comparison

(1) Fig. 6(a). When a publication is updated, the EFTG scheme might not need to compute geo-textual similarity for this updated publication based on classification. IGPT and CIQ should compute the geo-textual similarity. Therefore, when the number of keywords increases from 10 to 50, the EFTG update time increases very slowly. CIQ uses cells to store keywords. The update time of CIQ increases dramatically with an increase in the number of keywords. Therefore, the EFTG scheme has the best performance.

(2) Fig. 6(b). When the k value increases from 2 to 10, the EFTG update time increases very slowly because the threshold is removed. Meanwhile, the CIQ and IGPT update processes are almost unrelated to the k value, and their update times are therefore almost unaffected by the k value. However, the overall performance of the EFTG scheme is optimal.

(3) Fig. 6(c). Because each subscription in the EFTG scheme has a classification token, a publication update cannot affect all subscriptions. Actually, the publication update can only affect a few subscriptions in EFTG. However, a publication update may affect all subscriptions in CIQ and IGPT. Therefore, EFTG is more efficient than CIQ and IGPT when the number of subscriptions increases from 10 M to 50 M. CIQ indexes each subscription to

multiple cells based on precomputing text bounds. Thus, when the subscription number increases dramatically, the update time increases dramatically too.

(4) **Fig. 6(d)**. The EFTG scheme classifies publications, which decreases the complexity of publication updates. Each publication in EFTG has a classification token, and a publication update can only affect a small number of related subscriptions. Therefore, when the number of publications increases from 2 M to 10 M, EFTG has the best performance.

Based on all the experimental data, we can draw the following conclusions:

(1) The comprehensive performance of the EFTG scheme is significantly better than the two existing methods, IGPT and CIQ.

(2)The EFTG scheme has good performance in the face of various parameters. Therefore, EFTG can be widely used for various application scenarios, such as large data, distributed computing, e-commerce, mobile platform, cloud computing, etc.

(3) Because the user is not required to set the threshold, the user's actions are more convenient. That is, we improve the availability of geo-textual publish/subscribe systems.

7. Conclusion

Most recent Top-k geo-textual publishing/subscription approaches do not classify information. This results in needless matching of different kinds of information, which wastes valuable system resources. The user is required to set thresholds, and based on those thresholds, the system’s scored publications are difficult to match to the user’s required publications. Actually, many publications should not be scored. On the one hand, the user wastes time setting the parameter. On the other hand, the system wastes valuable resources computing redundant scores.

Therefore, we propose an efficient and flexible Top-k geo-textual publish/subscribe scheme. Our scheme improves the system efficiency by classifying information. Because we remove the threshold, we reduce redundant calculations, improve the recall rate, and improve the user experience. Extensive experiments prove that the proposed scheme is more efficient and effective.

References

[1] X. Liu, C. Yuan, E. Peng and Z. Yang, “Combined Service Subscription and Delivery Energy-Efficient Scheduling in Mobile Cloud Computing,” *KSII Transactions on Internet and Information Systems*, vol. 9, no. 5, pp. 1587-1605, 2015. Article (CrossRef Link).

[2] G. Li, Y. Wang, T. Wang and J. Feng, “Location-aware publish/subscribe,” in *Proc. of 19th international conference on Knowledge discovery and data mining*, SIGKDD 2013, pp. 802-810, 2013. Article (CrossRef Link).

[3] L. Chen, G. Cong, X. Cao, “An efficient query indexing mechanism for filtering geo-textual data,” in *Proc. of 2013 ACM International Conference on Management of Data*, SIGMOD 2013, pp. 749-760, 2013. Article (CrossRef Link).

[4] X. Wang, Y. Zhang, W. Zhang, X. Lin and W.Wang, “AP-Tree: efficiently support location-aware Publish/Subscribe,” *The International Journal on Very Large Data Bases*, vol. 24, no. 6, pp. 823-848, 2015. Article (CrossRef Link).

[5] H. Hu, Y. Liu, G. Li, J. Feng and K. L. Tan, “A location-aware publish/subscribe framework for parameterized spatio-textual subscriptions,” in *Proc. of IEEE 31st International Conference on Data Engineering*, ICDE 2015, pp. 711-722, 2015. Article (CrossRef Link).

[6] L. Chen, G. Cong, X. Cao, and K. L. Tan, "Temporal spatial-keyword top-k publish/subscribe," in Proc. of IEEE 31st International Conference on Data Engineering, ICDE 2015, pp. 255-266, 2015. Article (CrossRef Link).

[7] G. Cong and C. S. Jensen, "Querying Geo-Textual Data: Spatial Keyword Queries and Beyond," in *Proc. of 2016 International Conference on Management of Data*, SIGMOD 2016, pp. 2207-2212, 2016. Article (CrossRef Link).

[8] A. A. Diro, N. Chilamkurti and N. Kumar, "Lightweight Cybersecurity Schemes Using Elliptic Curve Cryptography in Publish-Subscribe fog Computing," *Mobile Networks and Applications*, vol. 22, no. 5, pp. 848–858, 2017. Article (CrossRef Link).

[9] Z. Chen, G. Cong, Z. Zhang, T. Z. J. Fuz and L. Chen, "Distributed Publish/Subscribe Query Processing on the Spatio-Textual Data Stream," in *Proc. of IEEE 33rd International Conference on Data Engineering*, ICDE 2017, pp. 1095-1106, 2017. Article (CrossRef Link).

[10] K. Zhao, Y. Liu, Q. Yuan, L. Chen, Z. Chen and G. Cong, "Towards personalized maps: mining user preferences from geo-textual data," *VLDB Endowment*, vol. 9, no. 13, pp. 1545-1548. Article (CrossRef Link).

[11] Z. Wu, H. Zhu, G. Li, Z. Cui, H. Huang, J. Li, E. Chen and G. Xu, "An efficient Wikipedia semantic matching approach to text document classification," *Information Sciences*, vol. 393, pp. 15-28. Article (CrossRef Link).

[12] H. Jiang, P. Zhao, V. S. Sheng, J. Xu, A. Liu, J. Wu and Z. Cui, "An Efficient Location-Aware Top-k Subscription Matching for Publish/Subscribe with Boolean Expressions," in *Proc. of International Conference on Database Systems for Advanced Applications*, DASFAA 2016, pp. 335-350, 2016. Article (CrossRef Link).

[13] C. Zhang, Y. Zhang, W. Zhang and X. Lin, "Inverted linear quadtree: Efficient top k spatial keyword search," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 7, pp. 1706-1721, 2016. Article (CrossRef Link).

[14] X. Wang, Y. Zhang, W. Zhang, X. Lin and Z. Huang, "Skype: top-k spatial-keyword publish/subscribe over sliding window," *VLDB Endowment*, vol. 9, no. 7, pp. 588-599, 2016. Article (CrossRef Link).

[15] X. Wang, Y. Zhang, W. Zhang, X. Lin and Z. Huang, "Top-k Spatial-keyword Publish/Subscribe Over Sliding Window," *The International Journal on Very Large Data Bases*, vol. 26, no. 3, pp. 301–326, 2017. Article (CrossRef Link).

[16] B. Wang, R. Zhu, X. Yang and G. Wang, "Top-K representative documents query over geo-textual data stream," *World Wide Web*, vol. 21, no. 2, pp. 537–555, 2018. Article (CrossRef Link).

[17] A. Wen, W. Lin, Y. Ma, H. Xie and G. Zhang, "News event evolution model based on the reading willingness and modified TF-IDF formula," Journal of High Speed Networks, vol. 23, no. 1, pp. 33-47, 2017. Article (CrossRef Link).

[18] M. Hoefling, C. G. Mills and M. Menth, "Distributed Load Balancing for the Resilient Publish/Subscribe Overlay in SeDAX," *IEEE Transactions on Network and Service Management*, vol. 14, no. 1, pp. 147-160, 2017. Article (CrossRef Link).

[19] A. Yu, P. K. Agarwal and J. Yang, "Subscriber assignment for wide-area content-based publish/subscribe," *IEEE Transactions on Knowledge and Data Engineering*, vol. 24, no. 10, pp. 1833-1847, 2012. Article (CrossRef Link).

[20] D. Zhang, C. Y. Chan, K and L.Tan, "An efficient publish/subscribe index for e-commerce databases," *VLDB Endowment*, vol. 7, no. 8, pp. 613-624, 2014. Article (CrossRef Link).

[21] A. Rizzardi, S. Sicari, D. Miorandi and A. Coen-Porisini, "AUPS: An Open Source AUthenticated Publish/Subscribe system for the Internet of Things," *Information Systems*, vol. 62, pp. 29-41, 2016. Article (CrossRef Link).

[22] A. Shraer, M. Gurevich, M. Fontoura and V. Josifovski, "Top-k publish-subscribe for social annotation of news," *VLDB Endowment*, vol. 6, no. 6, pp. 385-396, 2013. Article (CrossRef Link).

[23] Z. Cui, Z. Wu, C. Zhou, G. Gao, J. Yu, Z. Zhao and B. Wu, "An efficient subscription index for publication matching in the cloud," *Knowledge-Based Systems*, vol. 110, pp. 110-120, 2016. Article (CrossRef Link).

[24] V. Valero, G. Díaz and M. E. Cambronero, "Timed Automata Modeling and Verification for Publish-Subscribe Structures Using Distributed Resources," *IEEE Transactions on Software Engineering*, vol. 43, no. 1, pp. 76-99, 2017. Article (CrossRef Link).

[25] Z. Hmedeh, H. Kourdounakis, V. Christophides, C. d. Mouza, M. Scholl and N. Travers, "Content-Based Publish/Subscribe System for Web Syndication," *Journal of Computer Science and Technology*, vol. 31, no. 2, pp. 359–380, 2016. Article (CrossRef Link).

[26] L. Chen, G. Cong, C. S. Jensen and D. Wu, "Spatial keyword query processing: an experimental evaluation," VLDB Endowment, vol. 6, no. 3, pp. 217-228, 2013. Article (CrossRef Link).

[27] T. Silavi, F. Hakimpour, C. Claramunt and F. Nourian, "Design of a spatial database to analyze the forms and responsiveness of an urban environment using an ontological approach," *Cities*, vol. 52, pp. 8-19, 2016. Article (CrossRef Link).

[28] L. Zhao, L. Chen, R. Ranjan, K.K.R. Choo and J. He, "Geographical information system parallelization for spatial big data processing: a review," *Cluster Computing*, vol. 19, no. 1, pp. 139-152, 2016. Article (CrossRef Link).

[29] M. Yu, G. Li and J. Feng, "A cost-based method for location-aware publish/subscribe services," in *Proc. of 24th ACM International on Conference on Information and Knowledge Management*, CIKM2015, pp. 693-702, 2015. Article (CrossRef Link).

[30] Y. Sun, Y. Yin, Y. Jin and S. Gao, "Quad-tree spatial index for dynamic allocation of sea ice modeling in ice navigation simulator," in *Proc. of 15th ACM SIGGRAPH Conference on Virtual-Reality Continuum and Its Applications in Industry*, VRCAI 2016, vol. 1, pp. 115-118, 2016. Article (CrossRef Link).

[31] M. Tang, Y. Yu, Q. M. Malluhi, M. Ouzzani and W. G. Aref, "Locationspark: a distributed in-memory data management system for big spatial data," *VLDB Endowment*, vol. 9, no. 13, pp. 1565-1568, 2016. Article (CrossRef Link).

[32] K. Udomlamlert, T. Hara and S. Nishio, "Subscription-based data aggregation techniques for top-k monitoring queries," *World Wide Web*, vol. 20, no. 2, pp. 237–265, 2017. Article (CrossRef Link).

[33] R. Zhu, B. Wang and G.Wang, "Continuous Top-K Remarkable Comments over Textual Streaming Data Using ELM," in *Proc. of* ELM-2015, vol. 2, pp. 155-168, 2016. Article (CrossRef Link).

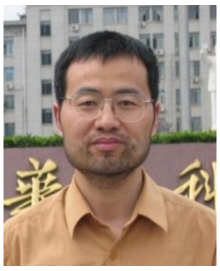
[34] K. Pripužić, I. P. Žarko and K. Aberer, "Top-k/w publish/subscribe: A publish/subscribe model for continuous top-k processing over data streams," *Information systems*, vol. 39, pp. 256-276, 2014. Article (CrossRef Link).

[35] K. Pripužić, I. P. Žarko and K.Aberer, "Time-and space-efficient sliding window top-k query processing," *ACM Transactions on Database Systems*, vol. 40, no. 1, Article No. 1, 2015. Article (CrossRef Link).

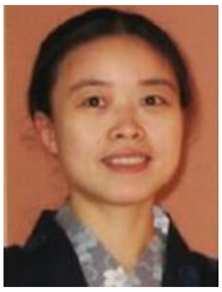
Hong Zhu received her B.S. degree, M.S. degree and Ph.D. degree from Huazhong University of Science and Technology in 1987, 1990 and 2001, respectively. Currently, she is a professor and a Ph.D. supervisor in School of Computer Science and Technology, Huazhong University of Science and Technology in China. Her main research areas are big data management technology, big data security technology, database theory and implementation technology, and database security technology.

Hongbo Li received his B.S. degree from Harbin Institute of Technology in 2006. Currently, he is a lecturer in School of Computer Science and Information Technology, Daqing Normal University in China. Meanwhile, he is a Ph.D. candidate in School of Computer Science and Technology, Huazhong University of Science and Technology in China. His main research areas are publish/subscribe system, data query, etc.

Zongmin Cui received the B.E. degree from Southeast University in 2002 and the M.S. degree from Huazhong University of Science and Technology in 2006. He received the Ph.D. Degree from Huazhong University of Science and Technology in 2014. He is currently an associate professor with the School of Information Science and Technology, Jiujiang University. His research interests include privacy protection, personalized recommendation, information security and data query.



Zhongsheng Cao received his M.S. degree and Ph.D. degree from Huazhong University of Science and Technology in 1991 and 1994, respectively. Currently, he is an associate professor and a M.S. supervisor in School of Computer Science and Technology, Huazhong University of Science and Technology in China. His main research areas are big data management technology and database.



Meiyi Xie received her B.S. degree, M.S. degree and Ph.D. degree from Huazhong University of Science and Technology in 1996, 1999 and 2009, respectively. Currently, she is a lecturer in School of Computer Science and Technology, Huazhong University of Science and Technology in China. Her main research areas are modern database theory and technology, database security, and cloud security.