A Location-Aware Publish/Subscribe Framework for Parameterized Spatio-Textual Subscriptions

Huiqi Hu† Yiqun Liu† Guoliang Li† Jianhua Feng† Kian-Lee Tan‡

†Department of Computer Science and Technology, TNList, Tsinghua University, Beijing 100084, China
‡School of Computing, National University of Singapore, Singapore.
hhq11@mails.tsinghua.edu.cn,{liguoliang,yiqunliu,fengjh}@tsinghua.edu.cn,tankl@comp.nus.edu.sg

***Abstract*—With the rapid progress of mobile Internet and the growing popularity of smartphones, location-aware publish/subscribe systems have recently attracted significant attention. Different from traditional content-based publish/subscribe, subscriptions registered by subscribers and messages published by publishers include both spatial information and textual descriptions, and messages should be delivered to relevant subscribers whose subscriptions have high relevancy to the messages. To evaluate the relevancy between spatio-textual messages and subscriptions, we should combine the spatial proximity and textual relevancy. Since subscribers have different preferences – some subscribers prefer messages with high spatial proximity and some subscribers pay more attention to messages with high textual relevancy, it calls for new location-aware publish/subscribe techniques to meet various needs from different subscribers.**

In this paper, we allow subscribers to parameterize their subscriptions and study the location-aware publish/subscribe problem on parameterized spatio-textual subscriptions. One big challenge is to achieve high performance. To meet this requirement, we propose a filter-verification framework to efficiently deliver messages to relevant subscribers. In the filter step, we devise effective filters to prune large numbers of irreverent results and obtain some candidates. In the verification step, we verify the candidates to generate the answers. We propose three effective filters by integrating prefix filtering and spatial pruning techniques. Experimental results show our method achieves higher performance and better quality than baseline approaches.

I. Introduction

Content-based publish/subscribe systems have been widely deployed and accepted in many applications, e.g., dbworld[1] and Google scholar citation alert[2]. Subscribers register their interests as subscriptions and publishers post messages in a publish/subscribe system, and the system delivers messages to relevant subscribers whose subscriptions have high relevancy to the messages. With the rapid progress of mobile Internet and the growing popularity of smartphones, subscribers have location-aware requirement in their subscriptions. For example, in a Groupon system, subscribers register their spatio-textual subscriptions to capture their interests (e.g., "`nike t-shirt discount` at Seoul, Korea"). For each Groupon message with textual description and location (e.g., "`nike clothing at cheap prices, including running shoes, t-shirts` at nike factory store, Seoul, Korea"), the system delivers the message to relevant subscribers. There are many other real-world applications, e.g., location-aware advertisements and location-aware tweets/news delivery [3, 15]. For example, seafood restaurants in `Korea` want to deliver their advertisements to tourists at `Korea` (e.g., `ICDE'15 attendees`) who like seafood. Twitter users want to be informed of relevant tweets posed near their locations.

Different from the content-based publish/subscribe problem, subscriptions and messages in our problem include both spatial information and textual descriptions. Although Li et. al. [15] studied the location-aware publish/subscribe problem, they separately handled the textual relevancy and spatial proximity and delivered a message to relevant subscribers if (1) all tokens of their subscriptions appear in the message and (2) these subscriptions have spatial overlaps with the message. To better quantify the relevancy, existing studies used a parameter to combine textual relevancy and spatial proximity [4, 6]. Generally subscribers have different preferences – some subscribers prefer messages with high spatial proximity while other subscribers pay more attention to messages with large textual relevancy to their subscriptions. It calls for new location-aware publish/subscribe techniques to meet various needs from different subscribers.

In this paper, we allow subscribers to parameterize their subscriptions and study the location-aware publish/subscribe problem on parameterized spatio-textual subscriptions. One big challenge in a location-aware publish/subscribe system is to achieve high performance as a publish/subscribe system is required to support millions of subscriptions and filter a message in milliseconds. To meet this high-performance requirement, we propose a filter-verification framework to efficiently deliver messages to relevant subscribers. In the filter step, we devise effective filters to prune large numbers of irreverent results and obtain some candidates. In the verification step, we verify the candidates to generate the final answers. We propose three effective filters to achieve high filtering power.

To summarize, we make the following contributions. (1) We formulate the location-aware publish/subscribe problem on parameterized spatio-textual subscriptions which support flexible requirements from different subscribers (see Section II). (2) We devise a filter-verification framework to deliver messages to relevant subscribers. We propose the spatial-oriented prefix which integrates spatial-pruning techniques into the prefix to prune irrelevant subscriptions (see Section III). (3) We present the region-aware prefix which utilizes hierarchical spatial index to improve the pruning power (see Section IV). (4) We seamlessly integrate prefix filtering and spatial-pruning techniques and propose the spatio-textual prefix to further enhance pruning power. We devise a cost-based method to select the best filtering strategy (see Section V). To the best of our knowledge, this is the first attempt to integrate prefix filtering and spatial-pruning techniques to support location-aware publish/subscribe. (5) The experimental results show our method achieves high quality and good performance and significantly outperforms baseline approaches (see Section VII).

[1]https://research.cs.wisc.edu/dbworld/
[2]http://scholar.google.com

II. Problem Formulation

Spatio-Textual Subscription. Subscribers register spatio-textual subscriptions to capture their interests. A spatio-textual subscription s includes a textual description s_{T} and a spatial location s_{L}, denoted by $s = (s_{\text{T}}, s_{\text{L}})$. s_{T} is a set of tokens $\{t_1, t_2, \cdots, t_{|s_{\text{T}}|}\}$ and each token t_i is associated with a weight $w(t_i)$. The token weight can be set as the inverted document frequency (IDF) of the token. s_{L} is a spatial location with latitude and longitude. For simplicity, in the paper we use subscribers and subscriptions interchangeably.

Spatio-Textual Message. A spatio-textual message m also consists of a textual description m_{T} and a message location m_{L}, denoted by $m = (m_{\text{T}}, m_{\text{L}})$.

Location-Aware Publish/Subscribe. A location-aware publish/subscribe system delivers each message to its relevant subscriptions. To quantify the relevancy between a subscription and a message, we use a similarity-based method. Before introducing the spatio-textual similarity metrics, we first define the spatial similarity between a message m and a subscription s. For simplicity, we use m, m_{T}, m_{L} (and s, s_{T}, s_{L}) interchangeably if the context is clear.

Definition 1 (Spatial Similarity): The spatial similarity $\text{sSIM}(s, m)$ between s and m is defined as

$$\text{sSIM}(s, m) = \max(0, 1 - \frac{\text{DIST}(s_{\text{L}}, m_{\text{L}})}{\text{MAXDIST}}).$$

where $\text{DIST}(s_{\text{L}}, m_{\text{L}})$ is the Euclidian distance between s_{L} and m_{L}, and MAXDIST is the maximum user-tolerated Euclidian distance between subscriptions and messages (which can be set to the maximum distance between subscriptions).

Next we define the textual similarity between a subscription s and a message m.

Definition 2 (Textual Similarity): The textual similarity $\text{TSIM}(s, m)$ between s and m is defined as

$$\text{TSIM}(s, m) = \frac{\sum_{t \in s_{\text{T}} \cap m_{\text{T}}} w(t)}{\sum_{t \in s_{\text{T}}} w(t)}.$$

The function is similar to the weighted Jaccard coefficient and the difference is that the Jaccard function uses the union size of two token sets as the denominator. However, in a publish/subscribe system, messages usually have many more tokens than subscriptions, and the union size will significantly affect the textual similarity. For instance, consider a subscription "`kobe, shoes`" and two messages "`Kobe NBA player`" and "`Kobe Bryant basketball shoes sales promotion. Large amount of discount.`" Obviously the second message is more relevant to the subscription. However, the second message has smaller Jaccard similarity to the subscription. Instead, our similarity function can alleviate this problem. Moreover, our techniques can be easily extended to support other functions (see Section VI).

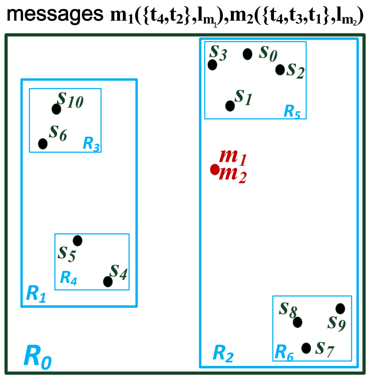
Then, we combine the textual similarity and spatial similarity to quantify the spatio-textual similarity.

Definition 3 (Spatio-Textual Similarity): The spatio-textual similarity $\text{SIM}(s, m)$ between s and m is

$$\text{SIM}(s, m) = \delta \cdot \text{TSIM}(s, m) + (1 - \delta) \cdot \text{sSIM}(s, m).$$

subscriptions

s_0: $(\{t_4,t_2\},l_0,0.7,0.8)$
s_1: $(\{t_4,t_3,t_2\},l_1,0.5,0.8)$
s_2: $(\{t_4,t_3,t_1\},l_2,0.5,0.7)$
s_3: $(\{t_5,t_4,t_1\},l_3,0.6,0.75)$
s_4: $(\{t_5,t_2,t_1\},l_4,0.5,0.6)$
s_5: $(\{t_3,t_2,t_1\},l_5,0.5,0.6)$
s_6: $(\{t_3,t_2,t_1\},l_6,0.2,0.7)$
s_7: $(\{t_5,t_4,t_1\},l_7,0.6,0.7)$
s_8: $(\{t_5,t_4,t_2\},l_8,0.7,0.7)$
s_9: $(\{t_4,t_2,t_1\},l_9,0.5,0.8)$
s_{10}: $(\{t_5,t_3\},l_{10},0.5,0.8)$



tokens

token	discount	adidas	nike	t-shirt	shoes
id	t_5	t_4	t_3	t_2	t_1
weight	0.5	0.4	0.3	0.2	0.1

spatial similarity

s	s_0	s_1	s_2	s_3	s_4	s_5	s_6	s_7	s_8	s_9	s_{10}
sSIM(s,m)	0.6	0.7	0.6	0.6	0.55	0.55	0.4	0.3	0.4	0.3	0.4

Fig. 1. A Running Example.

where δ is a preference parameter to tune the weight of textual and spatial similarity. The larger δ is, the more important the textual similarity is. A subscription s and a message m is called *relevant* if their similarity exceeds a threshold τ. Subscribers usually have different preferences and requirements on δ and τ. For instance, some subscribers prefer highly relevant results while some subscribers want to get more results. To address these problems, we allow subscribers to parameterize their parameters δ and τ.

Parameterized Spatio-Textual Subscription. A parameterized spatio-textual subscription s includes a textual description s_{T}, a spatial location s_{L}, a preference parameter s_δ, and a threshold s_τ, denoted by $s = (s_{\text{T}}, s_{\text{L}}, s_\delta, s_\tau)$.

Location-Aware Publish/Subscribe Problem on Parameterized Spatio-Textual Subscriptions. Given a set of subscriptions $\mathcal{S}$, a message m, deliver the message m to the subscription $s \in \mathcal{S}$ if $\text{SIM}(s, m) \geq s_\tau$.[3]

Example 1: Figure 1 shows 11 parameterized spatio-textual subscriptions and 2 messages m_1=($\{t_4$=`adidas`, t_2=`t-shirt`$\}$,l_{m_1}) and m_2=($\{t_4$=`adidas`, t_3=`nike`, t_1=`shoes`$\}$, l_{m_2}), where l_{m_1} and l_{m_2} denote the same location. The spatial similarities between messages and subscriptions are shown in the figure. The weights of t_5, t_4, t_3, t_2, t_1 are respectively $0.5, 0.4, 0.3, 0.2, 0.1$. Consider subscription $s_0 = (\{t_4$=`adidas`, t_2=`t-shirt`$\}, l_0, 0.7, 0.8)$. $\text{SIM}(s_0, m_1) = 0.7 * \frac{0.4+0.2}{0.4+0.2} + 0.3 * 0.6 = 0.88 > s_{0\tau} = 0.8$. Thus s_0 is an answer of m_1. Consider subscription $s_8 = (\{t_5$=`discount`, t_4=`adidas`,t_2=`t-shirt`$\}, l_8, 0.7, 0.7)$. $\text{SIM}(s_8, m_1) = 0.7 * \frac{0.4+0.2}{0.5+0.4+0.2} + 0.3 * 0.4 = 0.5 < s_{8\tau} = 0.7$. Thus s_8 is not an answer of m_1. Similarly, we can compute the result of m_1: $\{s_0\}$, and the result of m_2: $\{s_2, s_5\}$.

III. Spatial-Oriented Prefix

In this section, we first introduce the spatial-oriented prefix filter (Section III-A) and then develop a spatial-aware pruning technique (Section III-B). Finally we devise a spatial-oriented prefix based filter-verification framework (Section III-C).

[3]In this paper we focus on efficiently filtering parameterized spatio-textual subscriptions and obviously our method can also support subscriptions with the same parameters.

A. Spatial-Oriented Prefix Filter

Based on the spatio-textual function, given a subscription $s = (s_{\mathrm{T}}, s_{\mathrm{L}}, s_\delta, s_\tau)$, as the spatial similarity cannot exceed 1, we can deduce a textual similarity threshold

$$s_\tau^{\mathsf{T}} = \frac{s_\tau - (1 - s_\delta)}{s_\delta}. \quad (1)$$

We first assume $s_\tau^{\mathsf{T}} > 0$. If a message m is similar to s, the textual similarity cannot be smaller than s_τ^{T}. Based on the threshold, we can select a prefix for each subscription as follows. We first determine a global token ordering. (In this paper we sort the tokens by their weights in descending order.) For each subscription s, we deduce its textual similarity threshold s_τ^{T}. Based on its token set $s_{\mathrm{T}} = \{t_1, t_2, \cdots, t_{|s_{\mathrm{T}}|}\}$, we compute minimum p such that

$$\frac{\sum_{i=p}^{|s_{\mathrm{T}}|} w(t_i)}{w(s_{\mathrm{T}})} < s_\tau^{\mathsf{T}}, \quad (2)$$

where $w(s_{\mathrm{T}}) = \sum_{i=1}^{|s_{\mathrm{T}}|} w(t_i)$ is the total weight of tokens in s_{T}. The *spatial-oriented prefix* of s_{T} is $\textsc{Sig}(s) = \{t_1, t_2, ..., t_{p-1}\}$ and each token in $\textsc{Sig}(s)$ is called a *spatial-oriented token*. We can prove if a subscription s is similar to a message m (i.e., $\textsc{tSim}(s, m) \geq s_\tau^{\mathsf{T}}$), they must share at least one common spatial-oriented token, because the total weight of tokens after t_p is smaller than s_τ^{T}.

For example, consider two subscriptions $s_9 = (\{t_4, t_2, t_1\}, l_9, 0.5, 0.8)$ and $s_3 = (\{t_5, t_4, t_1\}, l_3, 0.6, 0.75)$ in Figure 1. We can get the textual similarity threshold ${s_9}_\tau^{\mathsf{T}} = \frac{0.8-(1-0.5)}{0.5} = 0.6$ and ${s_3}_\tau^{\mathsf{T}} = \frac{0.75-(1-0.6)}{0.6} = 0.58$. For s_9, as $w(t_2) + w(t_1) < w(s_{9\mathrm{T}}) \cdot 0.6$ and $w(t_4) + w(t_2) + w(t_1) > w(s_{9\mathrm{T}}) \cdot 0.6$, $\textsc{Sig}(s_9) = \{t_4\}$. Similarly for s_3, $\textsc{Sig}(s_3) = \{t_5\}$. For message $m_1 = (\{t_4, t_2\}, l_{m_1})$, s_9 is a candidate as m_1 contains a spatial-oriented token t_4 while s_3 can be pruned as m_1 does not include any spatial-oriented token of s_3.

Special Case. If $s_\tau^{\mathsf{T}} = \frac{s_\tau - (1-s_\delta)}{s_\delta} \leq 0$, a message m may be similar to s no matter whether they share common tokens. To address this issue, for a subscription s, if $s_\tau \leq 1 - s_\delta$, we introduce a virtual dummy token $*$ with weight of 0 (i.e., $w(*) = 0$), and the prefix of s includes its tokens and $*$. For example, consider $s_6 = (\{t_3, t_2, t_1\}, l_6, 0.2, 0.7)$. $s_\tau^{\mathsf{T}} = \frac{0.7-(1-0.2)}{0.2} = -0.5$ and $\textsc{Sig}(s_6) = \{t_3, t_2, t_1, *\}$.

B. Spatial-Aware Pruning

The spatial-oriented prefix uses the maximum spatial similarity (i.e., 1) as a loose spatial similarity bound, and it may involve many candidates. To alleviate this problem, we propose a spatial-aware pruning technique. Given a subscription s and a message m, suppose tokens in s and m are globally sorted. Let t_i denote the *first match token* between $\textsc{Sig}(s)$ and m (i.e., m does not contain tokens before t_i in $\textsc{Sig}(s)$). We can estimate an upper textual similarity bound of the message to subscription s based on the first match token as below.

$$\mathsf{UTB}(s \mid t_i) = \frac{\sum_{j=i}^{|s_{\mathrm{T}}|} w(t_j)}{\sum_{j=1}^{|s_{\mathrm{T}}|} w(t_j)} \geq \textsc{tSim}(s, m). \quad (3)$$

Accordingly, we can estimate a lower spatial similarity bound between m and s as below.

$$\mathsf{LSB}(s \mid t_i) = \frac{s_\tau - s_\delta \cdot \mathsf{UTB}(s \mid t_i)}{1 - s_\delta} \leq \textsc{sSim}(s, m). \quad (4)$$

For any message, if its spatial similarity to s is smaller than the lower spatial similarity bound $\mathsf{LSB}(s \mid t_i)$, the subscription s can be pruned as formalized in Lemma 1. Due to space constraints, we omit all the proofs in the paper.

Lemma 1: Given a message m and a subscription s, suppose their first match token is t_i. If $\textsc{sSim}(s, m) < \mathsf{LSB}(s \mid t_i)$, s is not similar to m.

For example, consider $s_9 = (\{t_4, t_2, t_1\}, l_9, 0.5, 0.8)$ and $m_1 = (\{t_4, t_2\}, l_{m_1})$. Their first match token is t_4. We compute the upper textual similarity bound $\mathsf{UTB}(s_9 \mid t_4) = \frac{0.7}{0.7} = 1$ and lower spatial similarity bound $\mathsf{LSB}(s_9 \mid t_4) = \frac{0.8-0.5}{0.5} = 0.6$. As $\textsc{sSim}(s_9, m_1)$=0.3 $< \mathsf{LSB}(s_9 \mid t_4)$, we prune s_9 without needing to verify its similarity to m_1.

Given a subscription s and a message m, we do not know which token is their first match token (if they have), and the first match tokens for different messages to the subscription are different. To address this issue, for each token t in the spatial-oriented prefix of s, we compute the upper textual similarity bound $\mathsf{UTB}(s \mid t)$ and the lower spatial similarity bound $\mathsf{LSB}(s \mid t)$. If subscription s is similar to message m, there must exist a token t in $s \cap m$ such that $\textsc{sSim}(s, m) \geq \mathsf{LSB}(s \mid t)$ as formalized in Lemma 2.

Lemma 2: Given a message m, if a subscription s is similar to the message, there must exist a token t in $s \cap m$ such that $\textsc{sSim}(s, t) \geq \mathsf{LSB}(s \mid t)$.

For example, consider $s_6 = (\{t_3, t_2, t_1, *\}, l_6, 0.2, 0.7)$. Its prefix is $\{t_3, t_2, t_1, *\}$, and the lower spatial similarity bounds for each token are $\{0.63, 0.75, 0.83, 0.88\}$. Especially, for token $*$, we estimate $\mathsf{UTB}(s_6 \mid *) = 0$ and $\mathsf{LSB}(s_6 \mid *) = \frac{0.7-0}{1-0.2} = 0.88$. Given a message $m_1 = (\{t_3, t_2\}, l_{m_1})$, as $\textsc{sSim}(s_6, m_1) = 0.4$ which is smaller than all the spatial similarity bounds, s_6 is not similar to m_1.

C. The Filter-Verification Framework

Spatial-Oriented Prefix Index. To index the spatial-oriented prefixes, we build an inverted index. The entries are tokens in the spatial-oriented prefix. Each token t is associated with an inverted list of triples $\langle s, \mathsf{LSB}(s \mid t), \mathsf{Idx}(s, t)\rangle$, where s is a subscription that contains the token, $\mathsf{LSB}(s \mid t)$ is the lower spatial similarity bound, and $\mathsf{Idx}(s, t)$ is the token order of t in s. We utilize $\mathcal{L}(t)$ to denote the inverted list of token t. Figure 2 shows the spatial-oriented prefix index. Consider $s_6 = (\{t_3, t_2, t_1, *\}, l_6, 0.2, 0.7)$. Its prefix is $\{t_3, t_2, t_1, *\}$ and lower spatial similarity bounds are $\{0.63, 0.75, 0.83, 0.88\}$. We insert $\{\langle s_6, 0.63, 1\rangle, \langle s_6, 0.75, 2\rangle, \langle s_6, 0.83, 3\rangle, \langle s_6, 0.88, *\rangle\}$ into inverted lists of $t_3, t_2, t_1, *$ respectively, where $1, 2, 3$ denote token orders of t_3, t_2, t_1 and $*$ denotes the dummy token. The number of entries is the number of distinct tokens. The total size of the inverted lists is at most the total number of tokens in the subscriptions. Thus the space complexity is $\mathcal{O}(\sum_{s \in \mathcal{S}} |s|)$.

Spatial-Oriented Filter-Verifiation Framework. Based on Lemma 2, given a message m, if a subscription s is a candidate of m, s must share at least one common token (denoted as t) with m such that $\textsc{sSim}(s, m) \geq \mathsf{LSB}(s \mid t)$. We utilize this property to devise a filter-verification framework.

Filter. For each token t in m (including the dummy token $*$), we identify its inverted list $\mathcal{L}(t)$. For each element $\langle s, \mathsf{LSB}(s \mid t), \mathsf{Idx}(s, t)\rangle \in \mathcal{L}(t)$, if $\textsc{sSim}(s, m) \geq \mathsf{LSB}(s \mid t)$,

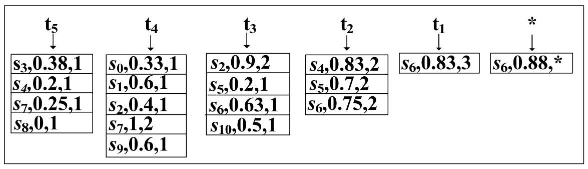


Fig. 2. Spatial-Oriented Prefix Index.

s is a candidate to the message m and we add $\langle s, \mathsf{Idx}(s,t)\rangle$ into the candidate set $\mathcal{C}$; otherwise, we prune the subscription. The filtering complexity is $\mathcal{O}(\sum_{t\in m}|\mathcal{L}(t)|)$.

Verification. We verify the candidate $\langle s, \mathsf{Idx}(s,t)\rangle$. If s is an answer, we deliver the message m to the subscription s. Based on the similarity function, s and m are similar, if and only if $w(s\cap m) = \sum_{t\in s_{\mathrm{T}}\cap m_{\mathrm{T}}} w(t) \geq \mathsf{minw}$, where

$$\mathsf{minw} = \frac{s_{\mathcal{T}} - (1-s_\delta)\cdot \mathrm{sSIM}(s,m)}{s_\delta}\cdot w(s_{\mathrm{T}}). \tag{5}$$

$\mathrm{sSIM}(s,m)$ can be easily computed in $\mathcal{O}(1)$ time and $w(s_{\mathrm{T}})$ can be materialized, thus it is easy to compute minw. To compute $w(s\cap m)$, we check whether each token t in s from position $\mathsf{Idx}(s,t)$ appears in m. If yes, we add the corresponding weight $w(t)$ into $w(s\cap m)$. To facilitate the checking, we build a hash map for tokens in m. (We only need to build the hash map for the message once.) Thus the time complexity of verifying a subscription s is $\mathcal{O}(|s|)$.

Filtering Algorithm. The pseudo-code is shown in Figure 3. To cover the special case, we add a dummy token $*$ into the message (line 2). For each message token t, we retrieve $\mathcal{L}(t)$. For each element $\langle s, \mathsf{LSB}(s\mid t), \mathsf{Idx}(s,t)\rangle$ in $\mathcal{L}(t)$, if $\mathrm{sSIM}(s,m) \geq \mathsf{LSB}(s\mid t)$, s is a candidate, and if it is not in the candidate set, we add $\langle s, \mathsf{Idx}(s,t)\rangle$ as a candidate (lines 4-7). Finally we verify the candidates to generate the answer (lines 8-10). In the verification algorithm, we compute minw based on Equation 5 (line 2) and $w(s\cap m)$ by checking whether each token in s from position $\mathsf{Idx}(s,t)$ appears in m. If a token appears in m, we add its weight to $w(s\cap m)$ (lines 3-4). If $w(s\cap m)\geq \mathsf{minw}$, s is an answer (line 5).

IV. REGION-AWARE PREFIX

The spatial-oriented prefix based method requires to scan the inverted lists of message tokens. If the inverted lists are long, the method is expensive. To address this issue, we group the subscriptions based on their locality and prune a group if all subscriptions in the group have small similarity to the message. To achieve this goal, we propose region-based pruning techniques (Section IV-A) and devise the region-aware prefix (Section IV-B). Finally we incorporate these techniques into our framework (Section IV-C).

A. Region-Based Pruning

We partition the whole space into multiple regions (e.g., R-tree) and split the subscriptions in each inverted list $\mathcal{L}(t)$ into several sublists based on the regions. Formally, for each region $\mathcal{R}$, we create a sublist $\mathcal{L}(t,\mathcal{R})$ which contains all subscriptions in $\mathcal{L}(t)$ that appear in $\mathcal{R}$. For each region $\mathcal{R}$, we maintain a lower spatial similarity bound to each token t, denoted by $\mathsf{LSB}(\mathcal{R}\mid t)$, which is the minimum value among all lower spatial similarity bounds of subscriptions in $\mathcal{R}$, i.e.,

$$\mathsf{LSB}(\mathcal{R}\mid t) = \min_{s\in\mathcal{L}(t,\mathcal{R})} \mathsf{LSB}(s\mid t). \tag{6}$$

Then given a message m, for each token t in the message, we retrieve the sublists of the token. For each sublist $\mathcal{L}(t,\mathcal{R})$, we

```
Algorithm 1: Spatial-Oriented Prefix Filtering
   Input: $m$: A message; $\mathcal{L}$: Spatial-oriented index for $\mathcal{S}$
   Output: $\mathcal{A}$: Answers of $m$
 1 begin
 2     $m_{\mathrm{T}}$.Push($*$) and sort $m_{\mathrm{T}}$;
 3     Candidate set $\mathcal{C} = \emptyset$;
 4     for $t \in m$ do
 5         for $\langle s, LSB(s\mid t), Idx(s,t)\rangle$ on $\mathcal{L}(t)$ do
 6             if $\mathrm{sSIM}(s,m) \geq LSB(s\mid t)$ then
 7                 if $s \notin \mathcal{C}$ then $\mathcal{C}$.Add($\langle s, \mathsf{Idx}(s,t)\rangle$);
 8     for $\langle s, Idx(s,t)\rangle \in \mathcal{C}$ do
 9         if VERIFIY($s, Idx(s,t), m$) then
10             $\mathcal{A}$.Add($s$);
11     return $\mathcal{A}$;
12 end
```

```
Function VERIFIY (s, Idx (s,t) , m)
   Input: $s$: A subscription; $\mathsf{Idx}(s,t)$: Token order;
          $m$: A message
   Output: true or false
 1 begin
 2     $w(s\cap m)=0$; $\mathsf{minw} = \frac{s_{\mathcal{T}}-(1-s_\delta)\cdot\mathrm{sSIM}(s,m)}{s_\delta}\cdot w(s_{\mathrm{T}})$ ;
 3     for $i \in [Idx(s,t), |s_{\mathrm{T}}|]$ do
 4         if $t_i \in m$ then $w(s\cap m) = w(s\cap m) + w(t_i)$;
 5     if $w(s\cap m) \geq minw$ then return true;
 6     else return false ;
 7 end
```

Fig. 3. Spatial-Oriented Filtering Algorithm.

compute the maximum spatial similarity from the message to the corresponding region $\mathcal{R}$ as below.

$$\mathrm{MAXSIM}(m,\mathcal{R}) = \max(0, 1-\frac{\mathrm{MINDIST}(m,\mathcal{R})}{\mathrm{MAXDIST}}) \tag{7}$$

where $\mathrm{MINDIST}(m,\mathcal{R})$ is the minimum distance from the message location to the region. If $m\in\mathcal{R}$, MINDIST $(m, \mathcal{R})$ = 0; otherwise it is the minimum value among the distances from the message to the four corners and four boundaries of the region (which can be easily computed in $\mathcal{O}(1)$ time).

We prove $\mathrm{sSIM}(m,s) \leq \mathrm{MAXSIM}(m,\mathcal{R})$ as stated in Lemma 3. Based on this property, for each region $\mathcal{R}$, if $\mathrm{MAXSIM}(m,\mathcal{R}) < \mathsf{LSB}(\mathcal{R}\mid t)$, we have $\mathrm{sSIM}(m,s) < \mathsf{LSB}(\mathcal{R}\mid t)$ for $s\in\mathcal{R}$, and thus we can prune the sublist $\mathcal{L}(t,\mathcal{R})$. On the contrary, we need to access each subscription on the sublist $\mathcal{L}(t,\mathcal{R})$.

Lemma 3: Given a message m and a subscription $s\in\mathcal{R}$, we have $\mathrm{sSIM}(m,s)\leq\mathrm{MAXSIM}(m,\mathcal{R})$.

Example 2: Consider regions $\mathcal{R}_3$-$\mathcal{R}_6$ in Figure 1. $\mathcal{L}(t_4) = \{s_0,s_1,s_2,s_7,s_9\}$. Region $\mathcal{R}_5$ contains subscriptions $\{s_0,s_1,s_2\}$ and region $\mathcal{R}_6$ contains subscriptions $\{s_7,s_9\}$. We split the list into sublists $\mathcal{L}(t_4,\mathcal{R}_5) = \{s_0,s_1,s_2\}$ and $\mathcal{L}(t_4,\mathcal{R}_6)=\{s_7,s_9\}$. We compute $\mathsf{LSB}(\mathcal{R}_5\mid t_4)$=0.33. $\mathsf{LSB}(\mathcal{R}_6\mid t_4)$=0.6. Given a message m_1, as $\mathrm{MAXSIM}(m_1,\mathcal{R}_6) = 0.55 < \mathsf{LSB}(\mathcal{R}_6\mid t_4)$, we prune $\mathcal{L}(t_4,\mathcal{R}_6)$. As $\mathrm{MAXSIM}(m_1,\mathcal{R}_5)$=0.75>$\mathsf{LSB}(\mathcal{R}_5\mid t_4)$, we access the subscriptions on the sublist $\mathcal{L}(t_4,\mathcal{R}_5)$.

B. Region-Aware Prefix

The spatial-oriented prefix filter uses the maximum spatial similarity to estimate the spatial similarity bound. If a message is outside of a region, the estimation is loose and we need to deduce a tighter bound. To this end, we propose a new region-aware prefix to prune the messages outside the region $\mathcal{R}$.

Considering each subscription s in $\mathcal{R}$, we compute its maximum spatial similarity to the messages outside $\mathcal{R}$, denoted by $\textsc{MaxSim}(s,\tilde{\mathcal{R}})$, which can be computed as below.

$$\textsc{MaxSim}(s,\tilde{\mathcal{R}}) = \max(0, 1 - \frac{\textsc{MinDist}(s,\tilde{\mathcal{R}})}{\textsc{MaxDist}}), \quad (8)$$

where $\textsc{MinDist}(s,\tilde{\mathcal{R}})$ is the minimum distance from subscription s to the four boundaries of region $\mathcal{R}$.

Thus for messages outside region $\mathcal{R}$, we can deduce a tighter spatial similarity bound

$$s_\tau^{\mathsf{T}}(\tilde{\mathcal{R}}) = \frac{s_\tau - (1-s_\delta)\cdot\textsc{MaxSim}(s,\tilde{\mathcal{R}})}{s_\delta}. \quad (9)$$

Obviously $s_\tau^{\mathsf{T}} \le s_\tau^{\mathsf{T}}(\tilde{\mathcal{R}})$. Using the tighter bound $s_\tau^{\mathsf{T}}(\tilde{\mathcal{R}})$, we select the region-aware prefix for each subscription to prune messages outside the region $\mathcal{R}$. Given a subscription s and a region $\mathcal{R}$ ($s \in \mathcal{R}$), the region-aware prefix is $\textsc{Sig}(s,\tilde{\mathcal{R}}) = \{t_1, \cdots, t_{r-1}\}$ where r is the minimum value such that

$$\frac{\sum_{i=r}^{|s_{\mathrm{T}}|} w(t_i)}{w(s_{\mathrm{T}})} < s_\tau^{\mathsf{T}}(\tilde{\mathcal{R}}), \quad (10)$$

The region-aware prefix is a subset of the spatial-oriented prefix. The larger the region, the shorter the region-aware prefix. Consider region $\mathcal{R}_6$ with subscriptions $\{s_7, s_8, s_9\}$. For $s_7 = (\{t_5, t_4, t_1\}, l_7, 0.6, 0.7)$, $s_\tau^{\mathsf{T}} = \frac{0.7-(1-0.6)}{0.6} = 0.5$. The spatial-oriented prefix of s_7 is $\textsc{Sig}(s_7) = \{t_5, t_4\}$. As $\textsc{MaxSim}(s_7,\tilde{\mathcal{R}_6}) = 0.9$, we compute a tighter spatial similarity bound $s_\tau^{\mathsf{T}}(\tilde{\mathcal{R}_6}) = \frac{0.7-0.4*0.9}{0.6} = 0.57$. Its region-aware prefix contains token t_5 and $\textsc{Sig}(s_7,\tilde{\mathcal{R}_6})=\{t_5\}$.

C. Hierarchical Spatial Prefix Index

The region-based pruning technique and the region-aware prefix have a limitation. On one hand, if each region is very large, it is hard to prune a whole region. On the other hand, if each region is small, a region contains a small number of subscriptions and the pruning power is low. It is rather hard to select an appropriate region granularity. To address this issue, we propose a hierarchical spatial index. Notice that any existing hierarchical indexes can be integrated into our techniques, and here we adopt the R-tree as an example.

Observations. To use the region-based pruning and region-aware prefix to filter a message, we make two observations. First, if the message m is in region $\mathcal{R}$, we use the spatial-oriented prefix to find answers in the region. If m is outside of $\mathcal{R}$, we use the region-aware prefix to find answers in the region. Second, the region-based pruning can prune $\mathcal{R}$ if $\textsc{MaxSim}(m,\mathcal{R}) < \mathsf{LSB}(\mathcal{R} \mid t)$ for $t \in m$. However if $m \in \mathcal{R}$, $\textsc{MaxSim}(m,\mathcal{R}) = 1$, and it cannot prune $\mathcal{R}$. In other words, when computing $\mathsf{LSB}(\mathcal{R} \mid t)$, we only need to consider the tokens in the region-aware prefix. Following these two observations, we build a hierarchical prefix index.

Hierarchical Prefix Index. We first build a spatial index. For each leaf region $\mathcal{R}$, we generate the spatial-oriented prefix and region-aware prefix for each subscription, and build the inverted indexes for them. For each token in the spatial-oriented prefix, we keep a spatial-oriented list $\mathcal{L}(t,\mathcal{R})$, and for each token in the region-aware prefix, we keep a region-aware list $\mathcal{L}(t,\tilde{\mathcal{R}})$. As the region-aware prefix is a subset of the spatial-oriented prefix, physically we only maintain a spatial-oriented list, and for each subscription on the list, we add a "region-aware" mark if the token is also in the region-aware prefix of the subscription. For ease of presentation, we still say there are two lists: the region-aware list is used to filter the messages outside the region and the spatial-oriented list is used to filter messages inside the region.

To prune a whole region, for each (leaf or non-leaf) region $\mathcal{R}$, we also maintain a lightweight signature which is a set of pairs $\langle t, \mathsf{LSB}(\mathcal{R} \mid t)\rangle$ where t is a token in the region-aware prefix and $\mathsf{LSB}(\mathcal{R} \mid t)$ is the lower spatial similarity bound which is the minimum value among lower spatial similarity bounds of all subscriptions in the region. It is worth noting that for different regions, $\mathsf{LSB}(\mathcal{R} \mid t)$ may be different.

Suppose the height of the spatial index is $\mathcal{D}$. For each token in the subscriptions, the token is visited at most $\mathcal{D}$ times to generate the lower bounds for different regions. Thus the worst-case complexity for building the index is $\mathcal{O}(\mathcal{D}\cdot\sum_{s\in\mathcal{S}}|s|)$ and the space complexity is also $\mathcal{O}(\mathcal{D}\cdot\sum_{s\in\mathcal{S}}|s|)$.

Example 3: Figure 4 shows the hierarchical prefix index. Consider the leaf region $\mathcal{R}_6$ with subscriptions $\{s_7, s_8, s_9\}$. $\textsc{Sig}(s_7)=\{t_5,t_4\}$, $\textsc{Sig}(s_8)=\{t_5\}$, $\textsc{Sig}(s_9)=\{t_4\}$. $\textsc{Sig}(s_7,\tilde{\mathcal{R}_6})=\{t_5\}$. $\textsc{Sig}(s_8,\tilde{\mathcal{R}_6})=\{t_5\}$. $\textsc{Sig}(s_9,\tilde{\mathcal{R}_6})=\{t_4\}$. t_5 is in the region-aware prefix of s_7 while t_4 is not. $\mathcal{L}(t_4,\mathcal{R}_6)=\{s_7,s_9\}$ and $\mathcal{L}(t_4,\tilde{\mathcal{R}_6})=\{s_9\}$. $\mathcal{L}(t_5,\mathcal{R}_6)=\mathcal{L}(t_5,\tilde{\mathcal{R}_6})=\{s_7,s_8\}$. As there are two region-aware tokens t_4 and t_5 in $\mathcal{R}_6$, we have $\mathsf{LSB}(\mathcal{R}_6 \mid t_4)=\mathsf{LSB}(s_9 \mid t_4)=0.6$ and $\mathsf{LSB}(\mathcal{R}_6 \mid t_5)=\min(\mathsf{LSB}(s_7 \mid t_5), \mathsf{LSB}(s_8 \mid t_5))=0$. Consider the non-leaf region $\mathcal{R}_1$. $\mathcal{R}_4$ is one of its subregion. $\textsc{Sig}(s_5,\tilde{\mathcal{R}_4})=\{t_3,t_2\}$. As $\textsc{MaxSim}(s_5,\tilde{\mathcal{R}_1})=0.65$, we compute $s_{5\tau}^{\mathsf{T}}(\tilde{\mathcal{R}_1})=0.55$ and $\textsc{Sig}(s_5,\tilde{\mathcal{R}_1})=\{t_3\}$. Therefore, we find that $\mathsf{LSB}(\mathcal{R}_1 \mid t_2)=\mathsf{LSB}(s_6 \mid t_2)=0.75>\mathsf{LSB}(\mathcal{R}_4 \mid t_2)=0.7$. For $m_1=(\{t_4,t_2\},l_{m_1})$, which is outside of $\mathcal{R}_1$, as $\textsc{MaxSim}(m_1,\mathcal{R}_1)=0.7<\mathsf{LSB}(\mathcal{R}_1 \mid t_2)$, we prune $\mathcal{R}_1$.

Filtering Algorithm. Given a message m, we access the children of the root. Consider each region $\mathcal{R}$.

Case 1: The message is inside $\mathcal{R}$. If $\mathcal{R}$ is a non-leaf region, we access its children; otherwise we scan the spatial-oriented list of t, $\mathcal{L}(t,\mathcal{R})$. For each subscription s on the list, if $\textsc{sSim}(s,m) < \mathsf{LSB}(s \mid t)$, we prune the subscription; otherwise s is a candidate.

Case 2: The message is outside of $\mathcal{R}$. We check each pair $\langle t, \mathsf{LSB}(\mathcal{R} \mid t)\rangle$. (2.1) If there does not exist a token t such that $\textsc{MaxSim}(m,\mathcal{R}) \ge \mathsf{LSB}(\mathcal{R} \mid t)$, we prune the region as formalized in Lemma 4; (2.2) otherwise, if the region is a non-leaf region, we access its children; otherwise, for each token t such that $\textsc{MaxSim}(m,\mathcal{R}) \ge \mathsf{LSB}(\mathcal{R} \mid t)$, we enumerate the subscriptions on the region-aware list $\mathcal{L}(t,\tilde{\mathcal{R}})$. For each subscription s on the list, if $\textsc{sSim}(s,m) < \mathsf{LSB}(s \mid t)$, we prune the subscription; otherwise s is a candidate.

Lemma 4: Given a message m, a region $\mathcal{R}$, and m is outside of $\mathcal{R}$, if there does not exist a token $t \in m$ such that $\textsc{MaxSim}(m,\mathcal{R}) \ge \mathsf{LSB}(\mathcal{R} \mid t)$, we can safely prune $\mathcal{R}$.

Figure 5 shows the pseudo-code of the algorithm. We first add the dummy token $*$, initialize the queue $\mathcal{Q}$, and push the

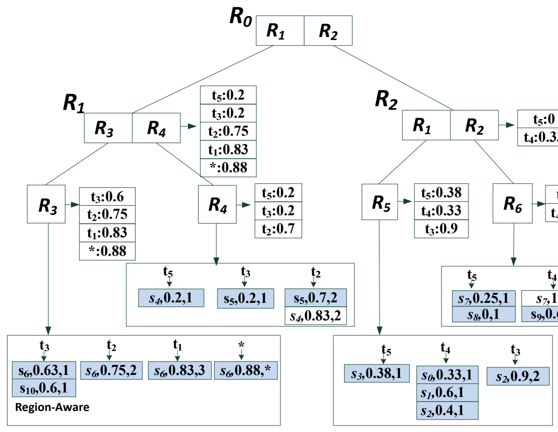


Fig. 4. Hierarchical Spatial Prefix Index.

root into the queue (line 2). Then we process each region in $\mathcal{Q}$. For each popped region $\mathcal{R}$, if $m \in \mathcal{R}$ and the region is a non-leaf region, we push each children of the region into $\mathcal{Q}$ (lines 5-8). If it is a leaf region, for each $t \in m$ which satisfies $\textsc{MaxSim}(m, \mathcal{R}) \geq \mathsf{LSB}(\mathcal{R} \mid t)$, we check the spatial-oriented list to generate the answer (lines 9-12). For $m \notin \mathcal{R}$, if the region is a non-leaf region and there exists $t \in m$ such that $\textsc{MaxSim}(m, \mathcal{R}) \geq \mathsf{LSB}(\mathcal{R} \mid t)$, we access the children of $\mathcal{R}$ and push the children into $\mathcal{Q}$ (lines 14-17); otherwise for each $t \in m$ such that $\textsc{MaxSim}(m, \mathcal{R}) \geq \mathsf{LSB}(\mathcal{R} \mid t)$, we check the region-aware list to generate the answer (lines 18-22). Finally we verify the candidates (lines 23-25).

V. Spatio-Textual Prefix

The spatial-oriented and region-aware prefix based methods generate candidates sharing a single token with the message. However, for some subscriptions, a single token is not enough. Consider $s_1 = (\{t_4, t_3, t_2\}, l_1, 0.5, 0.8)$ in Figure 1 and m shares a common token t_4 with s_1. Even if m and s_1 have the maximum spatial similarity (i.e. $\textsc{sSim}(s_1, m)$=1), $\textsc{Sim}(s_1, m)$ is at most $0.5 \cdot \frac{0.4}{0.9}$+0.5=0.72 $<$ ${s_1}_\tau$=0.8. Thus if s_1 is similar to m, s_1 must share another common token with m. Previous work [22] used this observation to improve the pruning ability for set similarity joins. However, it only considered the textual dimension with a unified threshold while we face spatio-textual subscriptions with diverse thresholds. We propose different indexes, algorithms and cost models to deal with spatio-textual data. We first propose the spatio-textual prefix (Section V-A) and then devise the corresponding filtering strategy (Section V-B). Finally we develop a cost-based method to select the best filtering strategy (Section V-C).

A. Spatio-Textual Prefix

Given a subscription s, if a message m shares only one common token $t_i \in \textsc{Sig}(s)$, the spatial-oriented prefix based method may take s as a candidate. However we observe that if $s_\delta \cdot \frac{w(t_i)}{w(s_\mathrm{T})} + (1 - s_\delta) < s_\tau$ (i.e., $\frac{w(t_i)}{w(s_\mathrm{T})} < s_\tau^\mathsf{T}$), m requires to share another token with s (if m is similar to s). Thus, given a subscription s, we generate its *spatio-textual prefix*, $\textsc{Sig}_1^+(s) = \{t_p, t_{p+1}, \cdots, t_{q-1}\}$, where p is computed by Equation 2 and q is the minimum number such that

$$\frac{w(t_1) + \sum_{j=q}^{|s_\mathrm{T}|} w(t_j)}{w(s_\mathrm{T})} < s_\tau^\mathsf{T}. \tag{11}$$

If there is no such q (i.e., $\frac{w(t_1)+w(t_{|s_\mathrm{T}|})}{w(s_\mathrm{T})} \geq s_\tau^\mathsf{T}$), $\textsc{Sig}_1^+(s) = \{t_p, t_{p+1}, \cdots, t_{|s_\mathrm{T}|}\}$. As $w(t_i) \leq w(t_1)$, if m does not share another token with s in $\textsc{Sig}_1^+(s)$, we have $\frac{w(t_i)+\sum_{j=q}^{|s_\mathrm{T}|} w(t_j)}{w(s_\mathrm{T})} <$ s_τ^T, and thus m is not similar to s. To utilize this property to do further pruning, besides maintaining the spatial-oriented prefix $\textsc{Sig}(s) = \{t_1, t_2, \cdots, t_{p-1}\}$, we also generate its spatio-textual prefix $\textsc{Sig}_1^+(s) = \{t_p, t_{p+1}, \cdots, t_{q-1}\}$. For each token t_j in $\textsc{Sig}_1^+(s)$, we also keep a lower spatial similarity bound $\mathsf{LSB}_1^+(s \mid t_j)$ where

Algorithm 2: Spatial Index Based Prefix Filtering

```
    Input: $m$: A message; $\mathcal{HI}$: Hierarchical Index of $\mathcal{S}$
    Output: $\mathcal{A}$: Answers of $m$
1   begin
2       $m_\mathrm{T}$.Push($*$) and sort $m_\mathrm{T}$;  $\mathcal{A} = \emptyset$;  $\mathcal{Q}$.Push($\mathcal{HI}$.root);
3       while $\mathcal{Q}$ is not empty do
4           $\mathcal{R} = \mathcal{Q}$.Pop();
5           if $m \in \mathcal{R}$ then
6               if $\mathcal{R}$ is a non-leaf node then
7                   for each child region $\mathcal{R}_c$ of $\mathcal{R}$ do
8                       $\mathcal{Q}$.Push($\mathcal{R}_c$);
9               else for $t \in m$ do
10                  for $s \in \mathcal{L}(t, \mathcal{R})$ do
11                      if $\textsc{sSim}(s, m) \geq \mathit{LSB}(s \mid t)$ then
12                          if $s \notin \mathcal{C}$ then  $\mathcal{C}$.Add($\langle s, \mathsf{Idx}(s, t)\rangle$);
13          else
14              if $\mathcal{R}$ is a non-leaf node then
15                  if $\exists t \in m, \textsc{MaxSim}(m, \mathcal{R}) \geq \mathit{LSB}(\mathcal{R} \mid t)$ then
16                      for each child region $\mathcal{R}_c$ of $\mathcal{R}$ do
17                          $\mathcal{Q}$.Push($\mathcal{R}_c$);
18              else   for $t \in m$ do
19                  if $\textsc{MaxSim}(m, \mathcal{R}) \geq \mathit{LSB}(\mathcal{R} \mid t)$ then
20                      for $s \in \mathcal{L}(t, \tilde{\mathcal{R}})$ do
21                          if $\textsc{sSim}(s, m) \geq \mathit{LSB}(s \mid t)$ then
22                              if $s \notin \mathcal{C}$ then
                                    $\mathcal{C}$.Add($\langle s, \mathsf{Idx}(s, t)\rangle$);
23      for $s \in \mathcal{C}$ do
24          if $\textsc{Verify}((s, \mathit{Idx}(s, t)), m)$ then
25              $\mathcal{A}$.Add($s$);
26      return $\mathcal{A}$;
27  end
```

Fig. 5. Hierarchical Prefix Filtering Algorithm.

$$\mathsf{LSB}_1^+(s \mid t_j) = \frac{s_\tau - s_\delta \cdot \mathsf{UTB}_1^+(s \mid t_j)}{1 - s_\delta}, \tag{12}$$

$$\mathsf{UTB}_1^+(s \mid t_j) = \frac{w(t_1) + \sum_{i=j}^{|s_\mathrm{T}|} w(t_i)}{\sum_{i=1}^{|s_\mathrm{T}|} w(t_i)}. \tag{13}$$

Given a message m, assume s and m share only one common token in their spatial-oriented prefixes. The spatial-oriented prefix based method may take s as a candidate. However if $s_\delta * \frac{w(t_i)}{w(s_\mathrm{T})} + (1 - s_\delta) \cdot \textsc{sSim}(1, s)m < s_\tau$, s requires to share another common token t_j in $\textsc{Sig}_1^+(s)$ such that $\textsc{sSim}(s, m) \geq \mathsf{LSB}_1^+(s \mid t_j)$. If there is no such token in the spatio-textual prefix, we can prune s.

Example 4: Consider $s_2 = (\{t_4, t_3, t_1\}, l_2, 0.5, 0.7)$. $s_\tau^\mathsf{T} = \frac{0.7-(1-0.5)}{0.5} = 0.4$. The spatial-oriented prefix of s_2 is $\textsc{Sig}(s_0) = \{t_4, t_3\}$. $\mathsf{UTB}(s_2 \mid t_4) = 1$ and $\mathsf{LSB}(s_2 \mid t_4) = \frac{0.7-0.5}{0.5} = 0.4$. $\mathsf{UTB}(s_2 \mid t_3) = 0.5$. $\mathsf{LSB}(s_2 \mid t_3) =$

$\frac{0.7-0.5\cdot 0.5}{0.5}=0.9$. $\text{SIG}_1^+(s_2)$=$\{t_1\}$. $\mathsf{UTB}_1^+(s_2 \mid t_1)$=0.63 and $\mathsf{LSB}_1^+(s_2 \mid t_1)$=0.78. Given a message $m_1=(\{t_4,t_2\},l_{m_1})$, it only contains token t_4 in $\text{SIG}(s_2)$. As $\text{SSIM}(s_2,m_1)=0.6 \geq \mathsf{LSB}(s_2 \mid t_4)$=0.4, s_2 cannot be pruned by the spatial-oriented prefix and is taken as a candidate. As $0.5\cdot\frac{0.4}{0.8}$+0.5·0.6 =0.55<0.7, m_1 requires to share another token with s_2. As $m_1 \cap \text{SIG}_1^+(s_2)$=$\emptyset$, s_2 is pruned by the spatio-textual prefix.

Next we generalize this idea. We divide all the tokens in s_{T} into disjoint sets: $\text{SIG}(s)$, $\text{SIG}_1^+(s)$, $\text{SIG}_2^+(s)$, $\cdots$, where $\text{SIG}(s)=\{t_1,t_2,\cdots,t_{p-1}\}$ and $\text{SIG}_1^+(s)=\{t_p,\cdots,t_{q-1}\}$ are computed based on Equations 2 and 11 respectively. We compute $\text{SIG}_k^+(s)=\{t_{p_k},\cdots,t_{q_k-1}\}$ based on $\text{SIG}_{k-1}^+(s)=\{t_{p_{k-1}},\cdots,t_{q_{k-1}-1}\}$ for $k>1$, where $p_k=q_{k-1}$ ($p_1=p$ and $q_1=q$) and q_k is the minimum number such that

$$\frac{\sum_{i=1}^{k} w(t_i)+\sum_{j=q_k}^{|s_{\text{T}}|} w(t_j)}{w(s_{\text{T}})} < s_\tau^{\mathsf{T}}. \tag{14}$$

If $\frac{\sum_{i=1}^{k} w(t_i)+w(t_{|s_{\text{T}}|})}{w(s_{\text{T}})} \geq s_\tau^{\mathsf{T}}$, $\text{SIG}_k^+(s)=\{t_{p_k},\cdots,t_{|s_{\text{T}}|}\}$ and we terminate; otherwise we compute $\text{SIG}_{k+1}^+(s)$ based on $\text{SIG}_k^+(s)$. Similar to Equations 12 and 13, we compute textual upper bound and spatial lower bound as below:

$$\mathsf{LSB}_k^+(s \mid t_j)=\frac{s_\tau - s_\delta\cdot \mathsf{UTB}_k^+(s \mid t_j)}{1-s_\delta}, \tag{15}$$

$$\mathsf{UTB}_k^+(s \mid t_j)=\frac{\sum_{i=1}^{k} w(t_i)+\sum_{i=j}^{|s_{\text{T}}|} w(t_i)}{\sum_{i=1}^{|s_{\text{T}}|} w(t_i)}. \tag{16}$$

We can also integrate the spatio-textual prefix into the region-aware prefix by replacing s_τ^{T} with $s_\tau^{\mathsf{T}}(\tilde{\mathcal{R}})$ for a leaf region $\mathcal{R}$. Due to space constraints, we omit the details.

B. Spatio-Textual Prefix Filtering

To use the spatio-textual prefix, we make a minor modification on the hierarchical index. For leaf-regions, we add the spatio-textual index. Given a leaf region $\mathcal{R}$, for each subscription in $\mathcal{R}$, we add the tokens in its spatio-textual prefix into the spatio-textual lists. We use $\mathcal{L}(t,\mathcal{R})$ and $\mathcal{L}_k^+(t,\mathcal{R})$ to respectively denote the spatial-oriented list for $\text{SIG}(s)$ and the spatio-textual list for $\text{SIG}_k^+(s)$ in $\mathcal{R}$. As the spatial-oriented prefix and the spatio-textual prefix have no overlap, the index contains the tokens of each subscription and the space complexity is $\mathcal{O}(\mathcal{D}\cdot\sum_{s\in\mathcal{S}}|s|)$.

Given a message m, after generating candidate $\mathcal{C}$ using the spatial-oriented prefixes (as discussed in Section IV-C), there are two strategies to compute the answers.

Strategy 1. We directly verify the candidates in $\mathcal{C}$ by checking the tokens in the candidates.

Strategy 2. We use spatio-textual prefixes to prune irrelevant subscriptions from $\mathcal{C}$ and get a smaller candidate set $\mathcal{C}' \subseteq \mathcal{C}$.

Next we discuss how to implement the second strategy. We use a hash table $\mathcal{C}$ to keep the candidates and a hash table $\mathcal{A}$ to keep the answer. For each candidate s on the spatial-oriented list of $t_i \in m$,

(1) If s is already in $\mathcal{A}$, we skip s.

(2) If s is not in $\mathcal{A}$, (2.1) If s is not in $\mathcal{C}$, we compute $\text{SSIM}(s,m)$. If $\text{SSIM}(s,m)<\mathsf{LSB}(s \mid t_i)$, we skip s; otherwise if $s_\delta * \frac{w(t_i)}{w(s_{\text{T}})}+(1-s_\delta)\cdot\text{SSIM}(s,m)<s_\tau$, s requires to appear in a spatio-textual list, and we add $\langle s,\mathsf{Idx}(s,t_i)\rangle$ into $\mathcal{C}$; otherwise s is an answer, we add it into $\mathcal{A}$. (2.2) If s is already in $\mathcal{C}$, we remove it from $\mathcal{C}$ and verify the candidate by scanning tokens in s from $\mathsf{Idx}(s,t_i)$. If s is an answer, we add it into $\mathcal{A}$. After accessing all spatial-oriented lists, we get some answers in $\mathcal{A}$ and some candidates in $\mathcal{C}$. For candidates in $\mathcal{C}$, they share only one common token with the message. We have two strategies to verify them. The first one is to verify the candidate by directly checking tokens. The second one is to use the spatio-textual prefix to do further pruning. It first retrieves spatio-textual lists of tokens in m. For each subscription s in $\mathcal{L}_1^+(t\in m,\mathcal{R})$, if s appears in $\mathcal{C}$ and $\text{SSIM}(s,m)\geq\mathsf{LSB}_1^+(\mathcal{R} \mid t)$, we add it into a new candidate set $\mathcal{C}'$ ($\mathcal{C}'\subseteq\mathcal{C}$). Next we verify candidates in $\mathcal{C}'$ with two possible strategies: the first directly verifies $\mathcal{C}'$ and the other uses $\text{SIG}_2^+(s)$ to do further pruning.

Example 5: Consider message $m_1=(\{t_4,t_2\},l_{m_1})$. For $\mathcal{R}_9$, we retrieve $\mathcal{L}(t_4,\mathcal{R}_9)$. For $s_0\in\mathcal{L}(t_4,\mathcal{R}_9)$, as $\text{SSIM}(s_0,m_1)$=0.6>0.14, s_0 is a candidate. Similarly, $\langle s_1,1\rangle$ and $\langle s_2,1\rangle$ are candidates. Thus $\mathcal{C}=\{\langle s_0,1\rangle,\langle s_1,1\rangle,\langle s_2,1\rangle\}$. For the second strategy, we retrieve $\mathcal{L}_1^+(t_4,\mathcal{R}_9)$=$\{s_3\}$, $\mathcal{L}_1^+(t_2,\mathcal{R}_9)$=$\{s_0,s_1\}$. s_0 is accessed again in $\mathcal{L}_1^+(t_2,\mathcal{R}_9)$. We verify s_0 and it is an answer. s_1 is also on the list $\mathcal{L}_1^+(t_2,\mathcal{R}_9)$. As $\text{SSIM}(s_1,m_1)=0.7<\mathsf{LSB}_1^+(s_1 \mid t_2)=0.93$, s_1 is pruned. As s_2 does not appear in the spatio-textual lists, s_2 is pruned. Thus the second strategy checks 3 subscriptions on the lists. The first strategy verifies $\mathcal{C}=\{s_0,s_1,s_2\}$ directly and accesses $1+2+2$ tokens as shown in Figure 6. The second strategy is better for m_1. Consider message $m_2=(\{t_4,t_3,t_1\},l_{m_2})$. For region $\mathcal{R}_9$, we retrieve $\mathcal{L}(t_4,\mathcal{R}_9)=\{s_0,s_1,s_2\}$, $\mathcal{L}(t_3,\mathcal{R}_9)=\{s_2\}$. For s_2, when we access $\mathcal{L}(t_4,\mathcal{R}_9)$, we put $\langle s_2,1\rangle$ into $\mathcal{C}$. When scanning $\mathcal{L}(t_3,\mathcal{R}_9)$, we find s_2 again and it is an answer. Finally $\mathcal{C}=\{\langle s_0,1\rangle,\langle s_1,1\rangle\}$. The second strategy retrieves $\mathcal{L}_1^+(t_4,\mathcal{R}_9)=\{s_3\},\mathcal{L}_1^+(t_3,\mathcal{R}_9)=\{s_1\},\mathcal{L}_1^+(t_1,\mathcal{R}_9)=\{s_2,s_3\}$. As s_0 does not appear in the lists, it is pruned. As s_1 appears in $\mathcal{L}_1^+(t_3,\mathcal{R}_9)$, it generates a smaller candidate set $\mathcal{C}'=\{s_1\}$. Next it verifies $\mathcal{C}'$ by checking its third token. The second strategy accesses 4 subscriptions and verifies another token for s_1. The first strategy checks 1+2 tokens to verify the two candidates in $\mathcal{C}$ and thus is better for m_2.

C. Cost-based Method

The first strategy scans smaller numbers of inverted lists and the second strategy accesses more inverted lists but with larger pruning power. Inspired by adaptive prefix selecting [22], we develop a similar cost-based method to select the best filtering strategy to compute the answers. To determine whether to use $\text{SIG}_k^+(s)$ for further pruning (strategy 2), we estimate the cost of the two strategies when processing $\text{SIG}_{k-1}^+(s)$. We denote the cost of the first verification strategy by COST_{V}^k and the cost of the second filtering strategy by COST_{F}^k. If $\text{COST}_{\text{V}}^k\leq\text{COST}_{\text{F}}^k$, we select the first strategy and directly verify the candidates; otherwise we continue to use $\text{SIG}_k^+(s)$.

We start by considering $\text{SIG}_1^+(s)$. The first strategy needs to verify the candidates in $\mathcal{C}$ by scanning tokens and the complexity is $\text{COST}_{\text{V}}^1=\sum_{s\in\mathcal{C}}|s|-\mathsf{Idx}(s,t)$, where $\mathsf{Idx}(s,t)$ is the token order of t in m. Obviously the cost COST_{V}^1 can be exactly computed when scanning the spatial-oriented

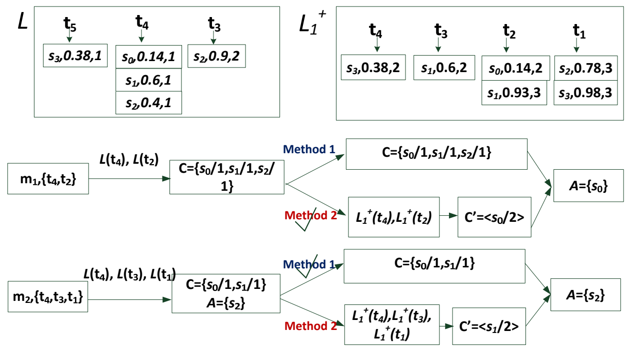


Fig. 6. An Example of Spatio-Textual Filter.

inverted lists. The second strategy includes two steps. The first step scans the spatio-textual lists and the time complexity is $\sum_{t\in m}|\mathcal{L}_1^+(t,\mathcal{R})|$ where $|\mathcal{L}_1^+(t,\mathcal{R})|$ is the size of spatio-textual list $\mathcal{L}_1^+(t,\mathcal{R})$, which can be materialized. The second step verifies the number of candidates in $\mathcal{C}'$. As $\mathcal{C}'$ is unknown, we need to estimate its size. Adaptive prefix selecting [22] adopted a sample based method to estimate candidate. However, it will invoke many additional sampling cost if we use the same method in every leaf nodes. To this end, we estimate the number based on appropriate proportions. The number of candidates should be in proportion to $\frac{\sum_{t\in m}|\mathcal{L}_1^+(t,\mathcal{R})|}{\sum_{t\in \mathcal{R}}|\mathcal{L}_1^+(t,\mathcal{R})|}$ and can be estimated as $|\mathcal{C}'| = |\mathcal{C}|\frac{\sum_{t\in m}|\mathcal{L}_1^+(t,\mathcal{R})|}{\sum_{t\in \mathcal{R}}|\mathcal{L}_1^+(t,\mathcal{R})|}$. The second strategy needs to verify candidates in $\mathcal{C}'$ and the complexity can be estimated by $\textsc{Cost}_{\mathtt{V}}^1\cdot\frac{|\mathcal{C}'|}{|\mathcal{C}|}-|\mathcal{C}'| = (\textsc{Cost}_{\mathtt{V}}^1-|\mathcal{C}|)\cdot\frac{\sum_{t\in m}|\mathcal{L}_1^+(t,\mathcal{R})|}{\sum_{t\in \mathcal{R}}|\mathcal{L}_1^+(t,\mathcal{R})|}$ The complexity of the second strategy can be estimate by $\textsc{Cost}_{\mathtt{F}}^1 = \sum_{t\in m}|\mathcal{L}_1^+(t,\mathcal{R})| + (\textsc{Cost}_{\mathtt{V}}^1 - |\mathcal{C}|)\cdot\frac{\sum_{t\in m}|\mathcal{L}_1^+(t,\mathcal{R})|}{\sum_{t\in \mathcal{R}}|\mathcal{L}_1^+(t,\mathcal{R})|}$. If $\textsc{Cost}_{\mathtt{V}}^1 \le \textsc{Cost}_{\mathtt{F}}^1$, we select the first strategy; otherwise, we select the second strategy.

Example 6: Consider m_1 in $\mathcal{R}_9$. $\mathcal{C} = \{s_0, s_1, s_2\}$. $\textsc{Cost}_{\mathtt{V}}^1 = \sum_{s\in\mathcal{C}}|s| - \mathsf{Idx}(s,t) = 1+2+2 = 5$. $\textsc{Cost}_{\mathtt{F}}^1 = |\mathcal{L}_1^+(t_4,\mathcal{R}_9)| + |\mathcal{L}_1^+(t_2,\mathcal{R}_9)| + (\textsc{Cost}_{\mathtt{V}}^1 - |\mathcal{C}|)\cdot\frac{|\mathcal{L}_1^+(t_4,\mathcal{R}_9)|+|\mathcal{L}_1^+(t_2,\mathcal{R}_9)|}{\sum_{t\in\mathcal{R}_9}|\mathcal{L}_1^+(t,\mathcal{R}_9)|} = 3+(5-3)*3/6{=}4$. Thus the second strategy is better. It probes $\mathcal{L}_1^+(t_4,\mathcal{R}_9)$ and $\mathcal{L}_1^+(t_2,\mathcal{R}_9)$. s_0 is a result. s_1 and s_2 are pruned. Consider m_2 in $\mathcal{R}_9$. $\mathcal{C} = \{s_0, s_1\}$. $\textsc{Cost}_{\mathtt{V}}^1 = \sum_{s\in\mathcal{C}}|s| - \mathsf{Idx}(s,t) = 1+2 = 3$. $\textsc{Cost}_{\mathtt{F}}^1 = |\mathcal{L}_1^+(t_4,\mathcal{R}_9)| + |\mathcal{L}_1^+(t_3,\mathcal{R}_9)| + |\mathcal{L}_1^+(t_1,\mathcal{R}_9)| + (\textsc{Cost}_{\mathtt{V}}^1 - |\mathcal{C}|)\cdot\frac{|\mathcal{L}_1^+(t_4,\mathcal{R}_9)|+|\mathcal{L}_1^+(t_3,\mathcal{R}_9)|+|\mathcal{L}_1^+(t_1,\mathcal{R}_9)|}{\sum_{t\in\mathcal{R}_9}|\mathcal{L}_1^+(t,\mathcal{R}_9)|} = 4+(3-2)*4/6 = 4.67$. The first strategy is better and it verifies candidates in $\mathcal{C}$.

After getting the candidate set $\mathcal{C}'$, we still need to decide which strategies should be used to compute answers based on $\mathcal{C}'$ and $\textsc{Sig}_k^+(s)$. Similar to $\textsc{Sig}_1^+(s)$, we compare the cost of the first strategy as $\textsc{Cost}_{\mathtt{V}}^k = \sum_{s\in\mathcal{C}^{k-1}}|s| - \mathsf{Idx}(s,t)$ and the cost of the second strategy is $\textsc{Cost}_{\mathtt{F}}^k = \sum_{t\in m}|\mathcal{L}_k^+(t,\mathcal{R})| + (\textsc{Cost}_{\mathtt{V}} - |\mathcal{C}^{k-1}|)\cdot\frac{\sum_{t\in m}|\mathcal{L}_k^+(t,\mathcal{R})|}{\sum_{t\in \mathcal{R}}|\mathcal{L}_k^+(t,\mathcal{R})|}$, where $\mathcal{C}^{k-1}$ is the candidate set computed by $\textsc{Sig}_{k-1}^+(s)$.

We devise a cost-based algorithm to automatically select the best strategy. Figure 7 shows the pseudo-code. We replace lines 9-12 in Algorithm 2 with Algorithm 3. We iteratively visit $\mathcal{L}, \mathcal{L}_1^+, \cdots, \mathcal{L}_k^+$. In each iteration, for each token t_i in m, we access subscription s on list $\mathcal{L}_k^+(t_i,\mathcal{R})$ (line 3). We skip s if it is in the answer set (line 5). Otherwise, we insert s into candidate set $\mathcal{C}$ and update $\textsc{Cost}_{\mathtt{V}}$ (lines 8-11) if s is not in $\mathcal{C}$. If s is already in $\mathcal{C}$ and cannot be pruned by the

Algorithm 3: Cost-based Algorithm

```
   Input: $m$: A message; $\mathcal{L}$: Spatial-oriented index on $\mathcal{S}$;
          Spatio-textual index $\mathcal{L}_k^+$ on $\mathcal{S}$;
   Output: $\mathcal{A}$: Answer to $m$
1  begin
      // Replace lines 9-12 in Algorithm 2
2     $\mathcal{L}_0^+ = \mathcal{L}$; $k = 0$; $\mathcal{C} = \emptyset$; $\textsc{Cost}_{\mathtt{V}} = \textsc{Cost}_{\mathtt{F}} = 0$;
3     while $\textsc{Cost}_{\mathtt{V}} \ge \textsc{Cost}_{\mathtt{F}}^k$ & $\mathcal{L}_k^+ \ne \phi$ do
4        for $t_i \in m$ do
5           for $s \in \mathcal{L}_k^+(t_i,\mathcal{R})$ & $s \notin \mathcal{A}$ do
6              if $s \in \mathcal{C}$ then
7                 if $c(s) >= k$ || $\textsc{sSim}(s,m) \ge \mathit{LSB}_k^+(s \mid t_i)$ then
                     $\textsc{Update}(s,\mathcal{C},\mathcal{A},\textsc{Cost}_{\mathtt{V}},t_i)$;
8              else if $k = 0$ & $\textsc{sSim}(s,m) \ge \mathit{LSB}(s \mid t_i)$ then
9                 $\textsc{Cost}_{\mathtt{V}} = \textsc{Cost}_{\mathtt{V}} + |s| - \mathsf{Idx}(s,t_i)$;
10                $\mathcal{C}.\mathsf{Add}(\langle s, \mathsf{Idx}(s,t_i), 0, 0\rangle)$;
11                $\textsc{Update}(s,\mathcal{C},\mathcal{A},\textsc{Cost}_{\mathtt{V}},t_i)$;
12       $\textsc{Cost}_{\mathtt{F}}^{k+1} = \sum_{t\in m}|\mathcal{L}_{k+1}^+(t,\mathcal{R})| + (\textsc{Cost}_{\mathtt{V}} - |\mathcal{C}|)\cdot\frac{\sum_{t\in m}|\mathcal{L}_{k+1}^+(t,\mathcal{R})|}{\sum_{t\in\mathcal{R}}|\mathcal{L}_{k+1}^+(t,\mathcal{R})|}$;
13    if $\mathcal{L}_k^+ \ne \phi$ then
14       for $\langle s, \mathit{Idx}(s,t), c(s), w(s)\rangle \in \mathcal{C}$ & $c(s) > k$ do
15          if $\textsc{Verify}^+(s, \mathit{Idx}(s,t), m, w(s))$ then
               $\mathcal{A}.\mathsf{Add}(s)$;
16    return $\mathcal{A}$;
17 end
```

Function $\textsc{Update}$ $(s,\mathcal{C},\mathcal{A},\textsc{Cost}_{\mathtt{V}},t_i)$

```
1  $w(s) = w(s) + w(t_i)$; $c(s)++$;
2  if $w(s) \ge \mathit{minw}(s)$ || $\mathit{Idx}(s,t_i) = |s_{\mathcal{T}}|$ then
3     $\mathcal{C}$.Remove$(s)$;
4     $\textsc{Cost}_{\mathtt{V}} = \textsc{Cost}_{\mathtt{V}} - (|s| - \mathsf{Idx}(s,t_i))$;
5     if $w(s) \ge \mathit{minw}(s)$ then
6        $\mathcal{A}$.Add$(s)$;
```

Function $\textsc{Verify}^+$ (s,Idx (s,t) ,m,w(s))

```
1  Same as Function Verify by replacing $w(s \cap m) = 0$ with
   $w(s \cap m) = w(s)$;
```

Fig. 7. Cost-based Algorithm.

prefix filter, we update $\mathcal{C}$ and $\textsc{Cost}_{\mathtt{V}}$ (lines 6-7). We keep four elements $\langle s, \mathsf{Idx}(s,t), c(s), w(s)\rangle$ in $\mathcal{C}$, where $c(s)$ is the current number of common tokens between s and m and $w(s)$ is the current common token weights, and update it with the $\textsc{Update}$ function. If the common token weight is larger than the required token weight (minw in Equation 5) or the current common token is the last token of s (lines 2-6 in the $\textsc{Update}$ function), s should be removed from $\mathcal{C}$. $\textsc{Cost}_{\mathtt{V}}$ is updated on each deletion of $\mathcal{C}$. After accessing the lists, we compute $\textsc{Cost}_{\mathtt{F}}^{k+1}$ and decide whether $\mathcal{L}_{k+1}^+$ should be used for further pruning (line 12). If $\textsc{Cost}_{\mathtt{V}} \le \textsc{Cost}_{\mathtt{F}}^{k+1}$, we stop the iteration and use the $\textsc{Verify}^+$ function to verify candidates (lines 13-15). Otherwise we use $\mathcal{L}_{k+1}^+$ to do further pruning. If all lists have already been visited, $\mathcal{C}$ is pruned.

VI. Discussion

A. Support Other Similarity Functions

Overlap. Overlap is a special case of our textual similarity function with each token weight of 1.

Weighted Jaccard. Suppose we use the weighted Jaccard function, i.e, $\textsc{Jac}(s,m) = \frac{\sum_{t\in s_{\mathcal{T}}\cap m_{\mathcal{T}}} w(t)}{\sum_{t\in s_{\mathcal{T}}\cup m_{\mathcal{T}}} w(t)}$. As $w(s_{\mathcal{T}}) =$

TABLE I. DATASETS.

Datasets	TWITTER	POI
Message (Avg Token Number)	12.1	41
Subscription (Number)	10M	10M
Subscription (Avg Token Number)	3	3
Subscription (Size, GB)	0.58	0.6

$\sum_{i=1}^{|s_{\mathrm{T}}|} w(t_i) \le \sum_{t \in s_{\mathrm{T}} \cup m_{\mathrm{T}}} w(t)$, $\textsc{Jac}(s,m) \le \textsc{tSim}(s,m)$. We can select the same spatial-oriented prefix $\textsc{Sig}(s)$ using the textual threshold s_{τ}^{T}. Similarly $\mathsf{UTB}(s \mid t_i)$ ($\mathsf{UTB}_k^{+}(s \mid t_i)$) is still an upper textual similarity bound, and $\mathsf{LSB}(s \mid t_i)$ ($\mathsf{LSB}_k^{+}(s \mid t_i)$) is still a lower spatial similarity bound.

Dice. Suppose we use the dice similarity function, i.e., $\textsc{Dice}(s,m) = \frac{2|s_{\mathrm{T}} \cap m_{\mathrm{T}}|}{|s_{\mathrm{T}}|+|m_{\mathrm{T}}|}$, where the weight of each token is 1. To select the spatial-oriented prefix, we have $\frac{2\sum_{t_i \in s_{\mathrm{T}} \cap m_{\mathrm{T}}} w(t_i)}{\sum_{t_i \in s_{\mathrm{T}}} w(t_i) + \sum_{t_i \in m_{\mathrm{T}}} w(t_i)} > 2 \cdot \textsc{tSim}(s,m)$. Thus, we generate $\textsc{Sig}(s)$ with textual threshold $\frac{s_{\tau}^{\mathsf{T}}}{2}$. We compute a similar lower spatial similarity bound, $\mathsf{LSB}(s \mid t_i) = \frac{s_{\tau} - 2s_{\delta} \cdot \mathsf{UTB}(s \mid t_i)}{1-s_{\delta}}$.

Cosine. Suppose we use the cosine function, i.e., $\textsc{Cos}(s,m) = \frac{|s_{\mathrm{T}} \cap m_{\mathrm{T}}|}{\sqrt{|s_{\mathrm{T}}|} \cdot \sqrt{|m_{\mathrm{T}}|}}$. We select $\textsc{Sig}(s)$ using threshold $\sqrt{|s_{\mathrm{T}}|} \cdot s_{\tau}^{\mathsf{T}}$ and compute a lower spatial similarity bound, $\mathsf{LSB}(s \mid t_i) = \frac{s_{\tau} - s_{\delta} \cdot \mathsf{UTB}(s \mid t_i)}{1-s_{\delta}}$, where $\mathsf{UTB}(s \mid t_i) = \frac{\sum_{j=i}^{|s_{\mathrm{T}}|} w(t_j)}{\sqrt{\sum_{j=1}^{|s_{\mathrm{T}}|} w(t_j)}}$.

B. Index Updates

First consider adding a new subscription. We insert it into the spatial index and locate the leaf region $\mathcal{R}$. We generate its spatial-oriented and spatio-textual prefix, compute the lower spatial bounds $\mathsf{LSB}(s \mid t)$ and $\mathsf{LSB}_k^{+}(s \mid t)$ for each t in the prefix, and insert it into the index. Next we generate its region-aware prefix using $\textsc{MaxSim}(s, \tilde{\mathcal{R}})$ and insert them into the index. We also update the lower spatial similarity bound for $\mathcal{R}$ (and $\mathcal{R}$'s ancestor regions). For each region-aware token t, if the lower spatial bound $\mathsf{LSB}(s \mid t)$ is smaller than the current lower bound $\mathsf{LSB}(\mathcal{R}' \mid t)$, we update the new lower bound as $\mathsf{LSB}(\mathcal{R}' \mid t) = \mathsf{LSB}(s \mid t)$. The worst-case complexity is $\mathcal{O}(\mathcal{D} \cdot |s_{\mathrm{T}}|)$. Second consider deleting a subscription s. We locate the leaf region and remove s. As deleting a subscription, the bound $\mathsf{LSB}(\mathcal{R} \mid t)$ on non-leaf regions will not decrease, thus we will not remove it from non-leaf nodes. Instead we use a widely-adopted delay manner to support deletions.

VII. EXPERIMENTAL STUDY

In this section we conduct extensive experiments to evaluate the efficiency and quality of our proposed techniques.

A. Experimental Setup

Datasets. We used two real datasets: TWITTER and POI. The TWITTER dataset was collected from *twitter.com*, which had 10 million tweets with locations. The POI dataset contained 10 million points of interests in USA. We randomly selected 1-5 tokens from each tweet/POI to generate subscriptions. We also randomly selected 2000 tweets/POIs as messages. To generate long messages, we combined 10 POIs as a single message. The datasets are shown in Table I.

Parameters. There are four main parameters. (1) Preference s_{δ}: It varied from 0 to 1. (2) Threshold s_{τ}: It varied from 0.5 to 1. (3) Message token number: It varied from 1 to 50. (4) Subscription token number: It varied from 1 to 5. When we varied a parameter, other parameters will be in the default range. We used idf to generate token weight.

TABLE II. USER STUDY.

(a) User Votes

s_{δ} \ s_{τ}	0.2	0.8	s_{τ}
1	2	2	4
0	0	1	2
0.5	5	4	10
s_{δ}	8	7	**55**

(b) Effectiveness

p/r/f s_{δ} \ s_{τ}	0.2	0.8	s_{τ}
1	0.40/0.90/0.55	0.51/0.20/0.29	0.50/0.69/0.58
0	0.37/0.95/0.53	0.48/0.23/0.31	0.47/0.80/0.59
0.5	0.41/0.88/0.56	0.58/0.20/0.30	0.53/0.83/0.65
s_{δ}	0.45/0.90/0.60	0.63/0.23/0.34	0.59/0.83/**0.69**

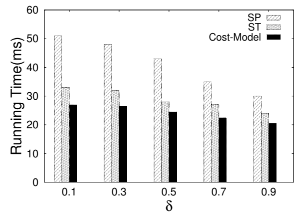
Baselines. Existing approaches cannot be directly applied to our problem on parameterized spatio-textual subscriptions. We extended two related state-of-the-art methods to support our problem. (1) IR-tree based methods. We extended spatial keyword search method IR-tree [6, 16] to support our problem. We kept the minimum and maximum preference parameters, minimum thresholds and minimum total weight of a subscription's token set in each IR-tree node, and used the bounds to do pruning. We further improved the method by separately constructing multiple IR-tree indexes based on different possible s_{δ} and s_{τ}. As s_{δ} and s_{τ} are consecutive, we keep 10 discrete points $(0.1, 0.2, \cdots, 1)$ for each parameter. Thus we built 10×10 IR-tree indexes and inserted subscriptions to corresponding indexes based on s_{δ} and s_{τ}. We call the basic IR-tree IRTREE-1 and the improved one IRTREE-100. (2) R^{t++} based method. We extended state-of-the-art location-aware publish/subscribe method R^{t++} [15] to solve our problem. R^{t++} assigned the tokens of each subscription into different ancestors of the leaf node where the subscription is located. To support ranking queries, we extended the method by keeping a textual similarity upper bound $\mathsf{UTB}(s \mid t)$ for each token t of subscription s and a spatial similarity lower bound $\mathsf{LSB}(s) = \frac{s_{\tau} - s_{\delta} \cdot \mathsf{UTB}(s \mid t)}{1-s_{\delta}}$ on node n, and kept the minimum bound $\mathsf{LSB}(n)$ for all subscriptions on node n.

Experimental Setting. All the algorithms were implemented in C++, using R-tree as the spatial index. We used in-memory setting. All the experiments were run on a computer with 40GB RAM, Intel Xeon CPU 2.93GHz, running Linux Ubuntu.

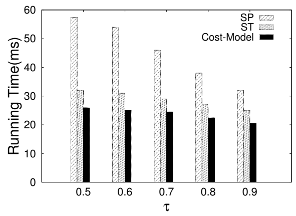
B. Effectiveness Study

To validate the utility of our parameterized solution, we deployed a system to conduct a user study. The system asked users to select parameter values between 0 and 1 using sliders (default 0.5) based on their own interests. We used four settings for s_{δ}: (1) textual similarity only ($s_{\delta} = 1$), (2) spatial similarity only ($s_{\delta} = 0$) (the results must contain at least one query token), (3) fixed parameter $s_{\delta} = 0.5$, and (4) user defined s_{δ}, and three settings for s_{τ}: (i) a small threshold ($s_{\tau} = 0.2$), (ii) a large threshold ($s_{\tau} = 0.8$), (iii) user defined s_{τ}. Thus there were 12 combinations for the two parameters.

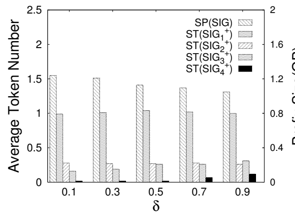
We conducted two experiments. First, we asked 100 different users to select their most satisfied setting, and the results are shown in Table II(a). We can see that the parameterized setting is most attractive and most of users prefer to define their own parameters. For example, 86% users voted the setting with a user-defined parameter and 55% users selected the setting with two user-defined parameters. Second, we evaluated the precision, recall, and F-measure. For each subscription, we returned a set of relevant results (i.e the messages contained at least one token and regardless of spatial constraint) and asked the user to mark their interested results. Taking the marked results as ground truth, we can compute the precision, recall and F-measure. Table II(b) shows the results. We had the following observations. First, s_{τ} can affect the result quality. A small threshold ($s_{\tau} = 0.2$) leaded to high recall but low



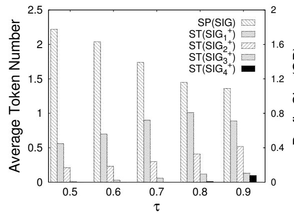
(a) Filter Performance(s_δ)



(b) Filter Performance(s_τ)

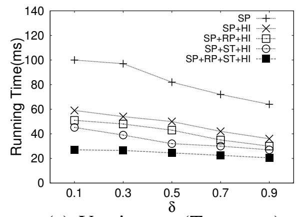


(c) Prefix Size (s_δ)

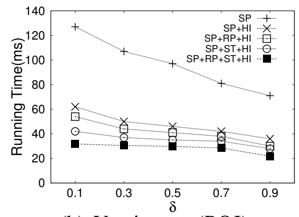


(d) Prefix Size (s_τ)

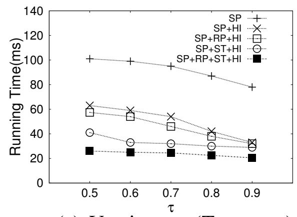
Fig. 8. Evaluation on Cost Model (TWITTER).



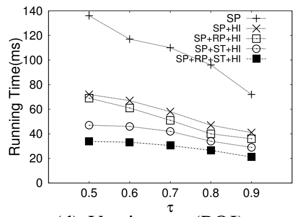
(a) Varying s_δ (TWITTER)



(b) Varying s_δ (POI)

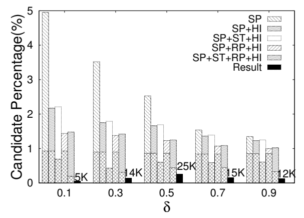


(c) Varying s_τ (TWITTER)

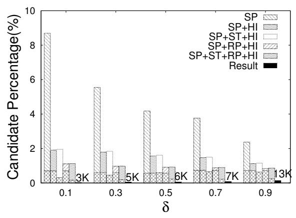


(d) Varying s_τ (POI)

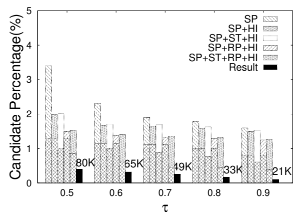
Fig. 9. Filtering Performance on Different Filters.



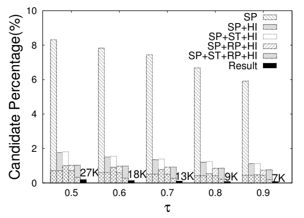
(a) Varying s_δ(TWITTER)



(b) Varying s_δ(POI)



(c) Varying s_τ (TWITTER)



(d) Varying s_τ (POI)

Fig. 10. Pruning Ability on Different Filters(bottom:candidates, top:accessed subscriptions).

precision as a small threshold returned more results. On the contrary, a large threshold ($s_\tau = 0.8$) had low recall but high precision. Second, s_δ can also influence the quality. Using both spatial and textual similarity ($s_\delta = 0.5$) had better quality than simply using one factor of them ($s_\delta = 1, 0$), and parameterized s_δ always achieved the best precision under different thresholds as parameterized setting allowed users to select their preference. Third, the parameterized setting on both s_δ and s_τ outperformed other settings. To summarize, *the parameterized setting can help users generate more related and accurate results and is better than other settings.*

Help Users Understand The Parameters. We used two useful strategies to help users understand the parameters. (1) Explanations of the parameters. We used previous subscriptions as examples to help users understand the the parameters. (2) User-friendly parameter tuning. We used sliders to help users select preference among textual and spatial relevancy and users can select parameter values using the sliders easily.

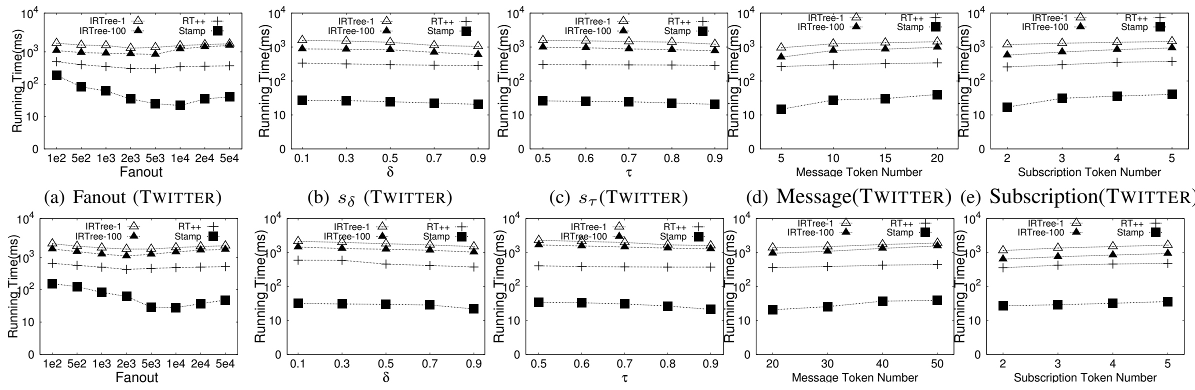
C. Efficiency Study

1) Evaluation on Cost Model: We evaluated our cost-based method on the TWITTER dataset. We compared three methods. (1) SP: using spatial-oriented prefix to generate candidates and then directly verifying all candidates; (2) ST: using spatial-oriented prefix to generate candidates and then always using spatio-textual prefixes to do further pruning; (3) Cost Model: adaptively selecting the best filtering strategy. Figure 8 shows the average filtering performance. We can see that ST was more efficient than SP as spatio-textual prefixes had larger textual pruning power than spatial-oriented prefixes. Among the three methods, the cost-based method performed the best as it adaptively selected the best filtering strategy.

Figure 8 also shows the average token number in the prefixes and the prefix size. ST achieved higher performance at the expense of involving a bit larger prefix sizes, because SP only maintained the spatial-oriented prefixes (SIG) and ST also required to keep additional spatio-textual prefixes ($\text{SIG}_1^+, \text{SIG}_2^+, \cdots$). It is worth noting that the average token number in all prefixes (including both SIG and SIG_k^+) was exactly the average token number in all subscriptions and the prefix size was the same as the data size, because each token in every subscription appeared in at most one prefix.

2) Evaluation on Different Filters: We compared spatial-oriented prefix (SP), spatial-oriented prefix with hierarchical index (SP+HI), region-aware prefix with hierarchical index (SP+RP+HI), spatio-textual prefix with hierarchical index (SP+ST+HI), and spatio-textual and region-aware prefix with hierarchical index (SP+RP+ST+HI) where we used the cost-based algorithm for the methods using the spatio-textual prefixes. Here the fanout of R-tree was 5000 as it achieved the highest performance at this point. We will show more results by varying the fanout later. We varied s_δ and s_τ on the two datasets. Figure 9 shows the filtering performance of different filters and Figure 10 shows the number of real results (5k-80k) and the ratios of the numbers of candidates (bottom bar) and accessed subscriptions (top bar) to the number of total subscriptions, where the candidates refer to subscriptions that were verified using the VERIFIY function and accessed subscriptions refer to subscriptions that were accessed on the inverted lists for different algorithms.

We made the following observations. First SP+HI outperformed SP as SP+HI used the hierarchical index to improve the pruning power which can avoid visiting the inverted lists of a whole region. SP+RP+HI further improved the performance by using tighter bounds. SP+RP+ST+HI achieved the highest performance, because SP+RP+ST+HI had the largest pruning power by integrating all the proposed techniques. For example, in Figure 9(a), on the Twitter dataset, for $s_\delta = 0.5$, SP took 82 ms, SP+HI improved it to 50 ms, SP+RP+HI further



(a) Fanout (TWITTER) (b) s_δ (TWITTER) (c) $s_\mathcal{T}$(TWITTER) (d) Message(TWITTER) (e) Subscription(TWITTER)

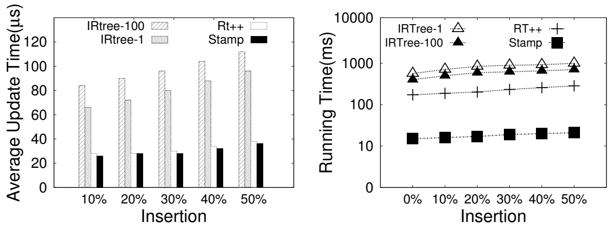
(f) Fanout (POI) (g) s_δ (POI) (h) $s_\mathcal{T}$ (POI) (i) Message(POI) (j) Subscription(POI)

Fig. 11. Comparison with State-of-the-art Algorithms.

improved to 43ms and SP+RP+ST+HI only took 25ms. Second, the region-aware prefix and the hierarchical index can reduce the number of accessed subscriptions using the spatial-pruning techniques. Thus SP, SP+HI, SP+RP+HI had the same number of candidates, however SP+RP+HI accessed smaller numbers of subscriptions than SP+HI which in turns visited less subscriptions than SP. Third, the spatio-texutal prefix accessed more subscriptions but reduced the number of candidates. For example, although SP+RP+ST+HI (SP+ST+HI) accessed many more subscriptions than SP+RP+HI (SP+HI), SP+RP+ST+HI (SP+ST+HI) involved less candidates as they utilized spatio-texutal prefixes to do pruning. Fourth, with the increase of $s_\mathcal{T}$, the performance increased, because for larger $s_\mathcal{T}$ there were smaller numbers of subscriptions required to be visited and verified, and we had greater opportunity to prune more irrelevant subscriptions. Fifth, with the decrease of s_δ, SP, SP+HI, SP+ST+HI and SP+RP+HI took much longer time, because for smaller s_δ, the spatial similarity is more important and they cannot estimate accurate prefix bounds. SP+RP+ST+HI was slightly affected as it selected the best strategy to prune many irrelevant subscriptions.

3) Comparison with State-of-the-art algorithms: We compared our best algorithm SP+RP+ST+HI, denoted by STAMP (spatio-textual prefix for parameterized subscriptions), with state-of-the-art algorithms IRTREE-1, IRTREE-100 and R^{t++}. We first compared the performance by varying R-tree fanout from 100 to $50,000$. Figure 11 shows the result. With increase of the fanout, the performance first increased and then decreased, and the best fanout for R-tree was around 5000. This is because for smaller fanout, the depth of the R-tree become larger and these methods took higher cost to traverse the index; for larger fanout, a region contained many more subscriptions and these methods had lower pruning power on each region. STAMP always outperformed existing algorithms with different fanouts, because we integrated prefix-based filters and spatial-pruning techniques together and achieved much larger pruning power than existing methods.

We then varied different parameters to compare these algorithms. As R^{t++} reached the best performance on the fanout of 5000, we reported performance on this point in other experiments. We had the following observations. First, IRTREE-100 was always better than IRTREE-1 since the pruning ability of each IR-tree node was improved after separating different s_δ and $s_\mathcal{T}$ values and IRTREE-100 can get much tighter bounds. Second, STAMP significantly outperformed IRTREE-100 by 2 orders of magnitude and was 10-50 times better than R^{t++} on different parameters and datasets. Third, as the increase of threshold $s_\mathcal{T}$, the performance increased as it was much easier to support larger thresholds. Fourth, with the increase of the message token numbers, the running time sightly increased as long messages contained more tokens and involved more answers. Fifth, with the increase of subscription token numbers, the running time also slightly increased, because more tokens in subscriptions generated longer prefixes.

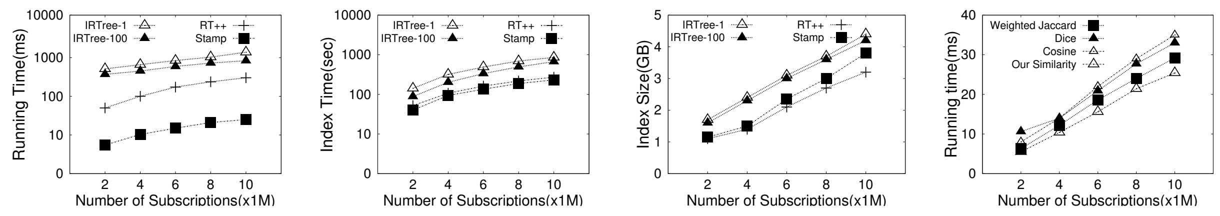


(a) Update Cost (b) Filtering Performance

Fig. 12. Update (TWITTER).

4) Update: We evaluated the update cost and filtering performance after updates. We first created indexes for 5 million subscriptions, then inserted 10% (0.5 million) subscriptions each time and reported the average update cost (Figure 12(a)) and the average filtering performance (Figure 12(b)) after updates. We can see with increase of updates, the average update time and average filtering time increased. However STAMP achieved higher filtering performance than existing methods as our prefix filters had larger pruning ability. STAMP had similar update cost as R^{t++} and lower update cost than IRTREE-1 and IRTREE-100, as STAMP and R^{t++} kept each token in a single R-tree node while IRTREE-1 and IRTREE-100 assigned a token to multiple R-tree nodes.

5) Scalability: We evaluated the scalability by varying the numbers of subscriptions on the TWITTER dataset. Figure 13 shows the results. We can see our method scaled much better than existing methods, because IR-tree added many tokens into R-tree nodes which were space consuming. With the increase of the numbers of subscriptions, the filtering performance of STAMP increased sublinearly. This is because STAMP can prune more dissimilar subscriptions on more subscriptions. In addition, STAMP and R^{t++} had smaller index sizes and time than IRTREE-1 and IRTREE-100, because IRTREE-1 and IRTREE-100 assigned tokens to multiple R-tree nodes, and STAMP and R^{t++} assigned tokens to a single R-tree node. We also evaluated other similarity functions and Figure 13(d) shows the results. We see that our method also achieved high performance on other similarity functions, because our method can easily adapt to different functions.



(a) Filtering Perforamnce (b) Index Time (c) Index Size (d) Other Similarity Functions

Fig. 13. Scalability.

VIII. Related Work

To the best of our knowledge, this is the first study on the location-aware publish/subscribe problem for parameterized spatio-textual subscriptions. Most related studies are location-aware filtering [15, 3]. Li et. al. [15] designed a location-aware publish/subscribe system which used spatial overlap to evaluate spatial similarity and the "AND" semantics to evaluate textual relevancy. Chen et al. [3] proposed an efficient index to match a stream of boolean range continuous queries over a stream of geo-textual objects. They focused on the "AND" and "OR" semantics to evaluate textual relevancy. Their problems are different from ours, and we generalized the problem and provided a fundamental technique to address this problem.

Spatial Keyword Search. There are many studies on spatial keyword search [9, 6, 20, 25, 5, 8, 18, 2, 14, 17, 26, 11, 13]. Chen et. al. [4] provided a survey of 12 traditional geo-textual indices and compared spatial keyword queries. Felipe et. al. [9] integrated text signatures into R-tree. Cong et. al. [6] and Rocha et. al. [20] combined inverted lists and R-tree. Zhang et. al. [25] combined Quadtree and inverted lists with text schema constraints. A region-based keyword query returns the objects relevant to the keywords within a query region [5, 11, 13]. A general solution is also to integrate textual information into some spatial indexes. Fan et. al. [8] studied a kind of searching problem, which finds similar regions by considering spatial overlap and textual similarity. Other research on spatial keyword search includes reverse spatial and textual knn query [18], collective keyword search [2] and moving top-k spatial keyword query [23, 12]. Michael et.al. [17] implemented a spatio-textual search engine. Different from spatial keyword search, we adopt a publish/subscribe model while it is a traditional search problem.

Information Filtering & Publish/Subscribe Services. Benjamin et.al. [21] built an information system to display spatial related news, but it did not provide publish/subscribe service. Foltz et. al. [10] studied how to deliver interest information to different users by a space vector model and a latent semantic indexing. Yan et. al. [24] provided an information filtering tool. Fabret et. al. [7] and Aguilera et. al. [1] studied the publish/subscribe problem, focusing on subscriptions with conjunction predicates. Traditional publish/subscribe methods only consider textual description [24, 19] while we consider both spatial information and textual descriptions.

IX. Conclusion

We have studied the location-aware publish/subscribe problem on parameterized spatio-textual subscriptions. We proposed a filter-verification framework and developed three effective filters. The spatial-oriented prefix filter utilized the maximum spatial similarity to generate the prefix. The region-aware prefix filter improved the spatial-oriented prefix filter using tighter bounds. The spatio-textual prefix filter utilized multiple tokens to do pruning. We integrated three filters into the hierarchical spatial index. Experimental results showed that our method significantly outperformed baseline approaches in terms of both performance and result quality.

Acknowledgement This work was partly supported by the 973 Program of China (2015CB358700 and 2011CB302206), and the NSFC project (61373024 and 61422205), Beijing Higher Education Young Elite Teacher Project (YETP0105), China Information Technology Security Evaluation Center, Huawei, Tencent, SAP, the "NExT Research Center" (WBS:R-252-300-001-490), and the FDCT/106/2012/A3.

References

[1] M. K. Aguilera, R. E. Strom, D. C. Sturman, M. Astley, and T. D. Chandra. Matching events in a content-based subscription system. In *PODC*, pages 53–61, 1999.

[2] X. Cao, G. Cong, C. S. Jensen, and B. C. Ooi. Collective spatial keyword querying. In *SIGMOD Conference*, pages 373–384, 2011.

[3] L. Chen, G. Cong, and X. Cao. An efficient query indexing mechanism for filtering geo-textual data. In *SIGMOD Conference*, pages 749–760, 2013.

[4] L. Chen, G. Cong, C. S. Jensen, and D. Wu. Spatial keyword query processing: An experimental evaluation. *PVLDB*, 6(3):217–228, 2013.

[5] Y.-Y. Chen, T. Suel, and A. Markowetz. Efficient query processing in geographic web search engines. In *SIGMOD Conference*, pages 277–288, 2006.

[6] G. Cong, C. S. Jensen, and D. Wu. Efficient retrieval of the top-k most relevant spatial web objects. *PVLDB*, 2(1):337–348, 2009.

[7] F. Fabret, H.-A. Jacobsen, F. Llirbat, J. Pereira, K. A. Ross, and D. Shasha. Filtering algorithms and implementation for very fast publish/subscribe. In *SIGMOD Conference*, pages 115–126, 2001.

[8] J. Fan, G. Li, L. Zhou, S. Chen, and J. Hu. Seal: Spatio-textual similarity search. *Proceedings of the VLDB Endowment*, 5(9):824–835, 2012.

[9] I. D. Felipe, V. Hristidis, and N. Rishe. Keyword search on spatial databases. In *ICDE*, pages 656–665, 2008.

[10] P. W. Foltz and S. T. Dumais. Personalized information delivery: An analysis of information filtering methods. *Commun. ACM*, 35(12):51–60, 1992.

[11] R. Hariharan, B. Hore, C. Li, and S. Mehrotra. Processing spatial-keyword (sk) queries in geographic information retrieval (gir) systems. In *SSDBM*, page 16, 2007.

[12] W. Huang, G. Li, K.-L. Tan, and J. Feng. Efficient safe-region construction for moving top-k spatial keyword queries. In *CIKM*, pages 932–941, 2012.

[13] P. Jin, H. Chen, S. Lin, X. Zhao, and L. Yue. Hybrid index structures for temporal-textual web search. In *APWeb*, pages 271–277, 2011.

[14] G. Li, J. Feng, and J. Xu. Desks: Direction-aware spatial keyword search. In *ICDE*, pages 474–485, 2012.

[15] G. Li, Y. Wang, T. Wang, and J. Feng. Location-aware publish/subscribe. In *KDD*, pages 802–810, 2013.

[16] Z. Li, K. C. K. Lee, B. Zheng, W.-C. Lee, D. L. Lee, and X. Wang. Ir-tree: An efficient index for geographic document search. *IEEE Trans. Knowl. Data Eng.*, 23(4):585–599, 2011.

[17] M. D. Lieberman, H. Samet, J. Sankaranarayanan, and J. Sperling. STEWARD: architecture of a spatio-textual search engine. In *ACM-GIS 2007, November 7-9, 2007, Seattle, Washington, USA, Proceedings*, page 25, 2007.

[18] J. Lu, Y. Lu, and G. Cong. Reverse spatial and textual k nearest neighbor search. In *SIGMOD Conference*, pages 349–360, 2011.

[19] K. Mouratidis and H. Pang. Efficient evaluation of continuous text search queries. *IEEE Trans. Knowl. Data Eng.*, 23(10):1469–1482, 2011.

[20] J. B. Rocha-Junior, O. Gkorgkas, S. Jonassen, and K. Nørvåg. Efficient processing of top-k spatial keyword queries. In *SSTD*, pages 205–222, 2011.

[21] B. E. Teitler, M. D. Lieberman, D. Panozzo, J. Sankaranarayanan, H. Samet, and J. Sperling. Newsstand: a new view on news. In *ACM-GIS 2008, November 5-7, 2008, Irvine, California, USA, Proceedings*, page 18, 2008.

[22] J. Wang, G. Li, and J. Feng. Can we beat the prefix filtering?: an adaptive framework for similarity join and search. In *SIGMOD Conference*, pages 85–96, 2012.

[23] D. Wu, M. L. Yiu, C. S. Jensen, and G. Cong. Efficient continuously moving top-k spatial keyword query processing. In *ICDE*, pages 541–552, 2011.

[24] T. W. Yan and H. Garcia-Molina. Index structures for selective dissemination of information under the boolean model. *ACM Trans. Database Syst.*, 19(2):332–364, 1994.

[25] D. Zhang, K.-L. Tan, and A. K. H. Tung. Scalable top-k spatial keyword search. In *EDBT*, pages 359–370, 2013.

[26] R. Zhong, J. Fan, G. Li, K. Tan, and L. Zhou. Location-aware instant search. In *CIKM*, pages 385–394, 2012.