IJRECS @ Sep – Oct 2014, V-2, I-1
ISSN-2321-5485 (Online) ISSN-2321-5784 (Print)

A Proxy Based Approach in Mobile Environments for Online Social Network

Ravi Ravindranath Tagore[1], S. V. Kishore Babu[2]
[1]M.Tech, Dept of CSE, Chirala Engineering College, Chirala, AP, India
[2]Assistant Professor, Dept of CSE, Chirala Engineering College, Chirala, AP, India.

ABSTRACT: In location-based services, users with location-aware mobile devices are able to make queries about their surroundings anywhere and at any time. While this ubiquitous computing paradigm brings great convenience for information access, it also raises concerns over potential intrusion into user location privacy. To protect location privacy, one typical approach is to cloak user locations into spatial regions based on user-specified privacy requirements, and to transform location-based queries into region-based queries. In this paper, we identify and address three new issues concerning this location cloaking approach. First, we study the representation of cloaking regions and show that a circular region generally leads to a small result size for region-based queries. Second, we develop a mobility-aware location cloaking technique to resist trace analysis attacks. Two cloaking algorithms, namely Max Accu Cloak and Min Comm Cloak, are designed based on different performance objectives. Finally, we develop an efficient polynomial algorithm for evaluating circular-region-based kNN queries. Two query processing modes, namely bulk and progressive, are presented to return query results either all at once or in an incremental manner. Experimental results show that our proposed mobility-aware cloaking algorithms significantly improve the quality of location cloaking in terms of an entropy measure without compromising much on query latency or communication cost. Moreover, the progressive query processing mode achieves a shorter response time than the bulk mode by parallelizing the query evaluation and result transmission.

KEYWORD: Mobile Social Behavior Motivation and Challenge, Region-Based Query Processing, Default Parameter Settings.

I. INTRODUCTION

Location-based services (LBS) are emerging as a major application of mobile geospatial technologies [2], [10], [12], [16]. In LBS, users with location-aware mobile devices are able to make queries about their surroundings anywhere and at any time. Spatial range queries and k-nearest-neighbor (kNN) queries are two types of the most commonly used queries in LBS. For example, a user can make a range query to find out all shopping centers within a certain distance of her current location, or make a kNN query to find out the k nearest gas stations. In these queries, the user has to provide the LBS server with her current location. But the disclosure of location information to the server raises privacy concerns, which have hampered the widespread use of LBS [7], [8], and [17]. Thus, how to provision location-based services while protecting user location privacy has recently become a hot research topic [1], [3], [5], [13], [14], [15].

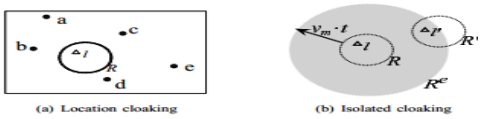


Fig.1: Dynamic Location Cloaking

Location cloaking is one typical approach to protecting user location privacy in LBS [3], [4], [5], [15]. Upon receiving a location-based spatial query (e.g., a range query or a kNN query) from the user, the system cloaks the user's current location into a cloaking region based on the user's privacy requirement. The location-based spatial query is thus transformed into a region-based spatial query before being sent to the LBS server. The LBS server then evaluates the region-based query and returns a result superset, which contains the query results for all possible location points in the cloaking region. Finally, the system refines the result superset to generate the exact results for the query location. Figure 1a shows a sample NN query. Instead of providing the exact location l, the system submits a cloaking region Rto the LBS server, which then returns the set of objects {b, c, d}that are the nearest neighbors of at least one point in R. Finally, among {b, c, d}, the system finds out the true nearest neighbor bof query location. Throughout this query processing procedure, the LBS server knows only the region R in which the user is located, not the exact location l. In the literature, a variety of cloaking algorithms based on snapshot user locations have been developed for different privacy metrics (e.g., [3], [4], [13], and [15]).

In this paper, we identify and address three new issues concerning the location cloaking approach. We first show that the representation of a cloaking region has an impact on the result superset size of the region-based query. In general, a small result superset is preferred for saving the cost of data transmission and reducing the workload of the result refinement process (especially if this process is implemented on the mobile client). We find that, given privacy requirement, representing the cloaking region with a circle generally leads to a smaller result superset than using other shapes. Second, we consider the location cloaking problem for continuous LBS queries. In such scenarios, trace analysis attacks are possible by linking historical cloaking regions with user mobility patterns. Assume that in our previous example, the user issues a second query at location l with a cloaking region R (see Figure 1b). If the LBS server somehow learns the user's maximum possible moving speed vm, the server can draw a region Re (the shaded area in Figure 1b) expanded from the last cloaking region R based nomad the interval t between the two queries. The server is then able to infer that the user must be located in the intersection area of Re and R′, which degrades the quality of location cloaking and may fail to meet the expected privacy requirement. The cloaking quality will further deteriorate with the analysis of more successive queries and cloaking regions. To address this issue, we develop a mobility-aware location cloaking technique that resists trace analysis attacks. Given that the server observes a cloaking region together with any series of historical cloaking regions, our proposed technique makes equal the derivable probability that the user will be located at any one point within the cloaking region. To achieve this, we leverage the probability theory to control the generation of cloaking regions and design two cloaking algorithms, namely MaxAccu Cloak and MinComm Cloak, based on different performance objectives.

MaxAccu Cloak is designed to maximize the accuracy of query results, while MinComm Cloak attempts to reduce the network communication cost. Finally, we investigate how to evaluate efficiently circular-region-based spatial queries on the LBS server. While the evaluation of circular-region-based range queries is straight-forward, we develop an efficient O (kM3) algorithm for evaluating circular-region-based kNN queries, where M is the cardinality of the spatial object set. In addition, we present two query processing modes, namely bulk and progressive, which return query results either all at once or in an incremental manner.

We conduct simulation experiments to evaluate the performance of the proposed location cloaking and query processing algorithms. The results show that the proposed mobility-aware cloaking algorithms outperform an isolated cloaking algorithm by up to 34% in terms of an entropy measure of cloaking quality, without compromising much on query latency or communication cost (sometimes performing even better). Regarding the end-to-end system performance, MaxAccu Cloak results in very high query accuracy, while MinComm Cloak achieves a good balance between communication cost and query accuracy. When the result superset size is small, the bulk and progressive modes of query progressing perform similarly.

For large result sets that require a long time to evaluate and transmit, the progressive mode achieves a shorter user-perceived response time than the bulk mode by parallelizing the query evaluation and result transmission. The rest of this paper is organized as follows. Surveys the related work on location privacy protection and spatial query processing gives an overview of our system model and location privacy metrics. Studies the representation of cloaking regions, followed by which presents the mobility-aware location cloaking algorithms the processing of circular-region-based queries is discussed in experimentally evaluates the proposed location cloaking and query processing algorithms. Finally, concludes this paper.

2. MOBILE SOCIAL BEHAVIOR MOTIVATION AND CHALLENGE

Nowadays, due to the wide use of mobile devices, more and more web applications have been expanded to mobile platforms, as have OSN ser-vices. We believe that it is the right time to high-light the importance of mobile social networks (MSNs). In MSNs, mobile users can publish and share information based on the social connections among them. On one hand, most major OSN platforms such as Face book, Twitter, and LinkedIn release mobile applications to allow users to access their services through mobile devices. On the other hand, more mobile-centric functions have been integrated into OSNs, such as location-based services and mobile communication. Understanding the user behavior in MSNs is very helpful for the design and implementation of MSN systems, improving the sys-

tem efficiency in mobile environments or supporting better mobile-centric functions. In this section, we focus on studies of user behaviors in MSNs.

3. EXISTING SOLUTIONS ANDDISCUSSION

Mobile Social Application — A large number of interesting and useful mobile social applications have been proposed. Social Serendipity [10] is a mobile-phone-based system that combines widely used mobile phones with the functionality of online introduction systems to cue informal face-to-face interactions between nearby users who do not know each other but probably should.

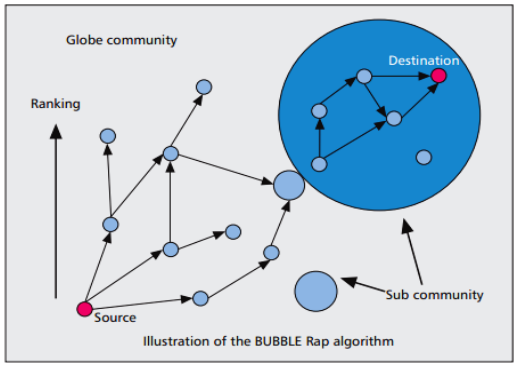


Figure 1.Illustration of the BUBBLE Rap algorithm.

Serendipity uses Bluetooth to sense nearby people and utilizes a centralized server to decide whether two users should be introduced to each other. The system calculates a similarity score by extracting the commonalities between two proximate users' profiles and behavioral data, and sums them according to user-defined weights. If the score is higher than the threshold set by both users, the system will inform them that someone nearby might be interested in them. For instance, internal collaboration in large companies can be facilitated by Serendipity for introducing people who are working on similar projects. It is emphasized that privacy issues are important and fundamental in Serendipity, and privacy-protecting tools should be designed carefully.

Geographical Prediction in OSN — Geography and social relationships are inextricably intertwined. As people spend more time online, data regarding these two dimensions are becoming increasingly precise, allowing building reliable models to describe their interaction. In [11], the study of user-contributed address and association data from Face book shows that the addition of social information produces improvement in accuracy of predicting physical location. First, friendship as a function of distance and rank is analyzed. It is found that at medium to long-range distances, the probability of friendship is roughly proportional to the inverse of distance. However, at shorter ranges, distance does not influence much. Then the maximum likelihood approach is presented to predict the physical location of a user, given the known location of her friends. This method predicts the physical location of 69.1 percent of the users with 16 or more located friends to within 25 mi, compared to only 57.2 percent using IP-based methods.

Friendship and Mobility in LBSN — although human movement and mobility patterns have a high degree of freedom and variation, they also exhibit structural patterns due to geographic and social constraints. Using cell phone location data, as well as data from two online location-based social networks, Cho et al. [12] aim to understand the basic laws that govern human motion and dynamics. It is found that humans experience a combination of strong short-range spatially and temporally periodic movement that is not impacted by the social network structure, while long-distance travel is more influenced by the social network ties. Furthermore, it is shown that social relationships can explain about 10 to 30 percent of all human movement, while periodic behavior explains 50 to 70 percent. Based on these findings, a model of human mobility is proposed that combines periodic short-range movements with travel due to the social network structure and gives an order of magnitude better performance than previous models.

Social-Based Routing in PSNs— widely used smart devices with networking capability form novel networks, such as pocket switched network (PSN). Due to the mobility of devices, PSNs are intermittently connected, and effective routing protocols are essential in such networks. Previous methods relied on building and updating routing tables to deal with dynamic conditions. Actually, the social structure and the interaction of users of smart devices have a great influence on the performance of routing protocols. BUB-BLE Rap [3] is a social-based forwarding method for PSNs. Two social and structural metrics, centrality and community, are used to effectively enhance delivery performance. As shown in Fig. 2, BUBBLE Rap first uses a centrality metric to spread out the messages (i.e., sending messages to more popular nodes), and then uses a community metric to identify the destination community and focus the messages to

the destination. The evaluation shows that BUBBLE Rap has a similar delivery ratio, but much lower resource utilization than flooding, control flooding, and other social-based forwarding schemes.

Content Distribution in MSN — Ioannidis et al. [4] study the dissemination of dynamic con-tent, such as news and traffic information, over an MSN. In this application, mobile users sub-scribe to a dynamic-content distribution service offered by their service provider. To improve coverage and increase capacity, it is assumed that users share any content updates they receive with other users they meet. Reference 14 deter-mines how the service provider can allocate its bandwidth optimally to make the content at users as "fresh" as possible. Moreover, there is a condition under which the system with high scalability is specified: even if the total bandwidth dedicated by the service provider remains fixed, the expected content age at each user grows slowly (as log(n)) with the number of users n.

The first objective function MaxAccu Cloak attempts to minimize the outer query blocking probability for i = K + 1, K + 2, ⋯ , L, thereby maximizing the query accuracy. In contrast, the second MinComm Cloak trades query accuracy for communication cost. It also aims to maximize the inner query blocking probability for i = 1, 2, ⋯ , K to increase the result reuse rate and save remote queries. The performance of these two cloaking algorithms will be evaluated by experiments in Section VII-C.On solving the linear program and obtainingv1, v2,· · · , v L–K+1, we can compute P(j|i)'s using Eq. (13). Then, given a new query with user location in ring i, the query has a probability of (1–PjP(j|i)) to be blocked. If the query is not blocked, the distance between the new cloaking region and the old cloaking region can be randomly generated based on the probabilities of P(j|i)'s. Given the distance, the center of the new cloaking region can be randomly generated on the corresponding arc. A summary of the optimal mobile-aware cloaking technique is described in Algorithm 1.

Algorithm 1 Overview of Mobility-Aware Location Cloaking

Input: mobility pattern U (·), old cloaking region centered at O, new user location in ring i

Output: the center of the new cloaking region

Procedure:

1: compute Q(i|j)'s using Eqs. (11) and (12)

2: construct a linear program formed by Eqs. (15) And (16) (for MaxAccu Cloak), or Eqs. (15) And (17) (for Min-Comm Cloak), depending on the performance objective

3: solve the linear program to get v_j 's 4: compute P (j|i)'s usin g Eq. (13)

5: determine whether the query is blocked based on theP probability of (1 –6

6. **if** the query is not blocked **then**

7. generate the distance of the new cloaking region from O by following P (j|i)'s

8. randomly generate the center of the new region on the corresponding arc

4. REGION-BASED QUERY PROCESSING

This section discusses how to process circular-region-based queries on the server side. The evaluation of a region-based range query is straightforward since it is still a range query (with an extended range), which simply retrieves all the objects within the spatial range. Thus, we focus on the evaluation of circular-region-based kNN queries (hereafter called kCRNN queries) in this section. Following Theorem 1, the results of a kCRNN query include all the objects in the circular region and the kNNs of the points on the perimeter of the circle (denoted by Ω).

In the following, we propose two kCRNN processing al-gorithms: a *bulk* algorithm that generates the query results all at once at the end of query evaluation and a *progressive* algorithm that produces the results incrementally during query evaluation.

A. Bulk Query Processing of k*CRNN*

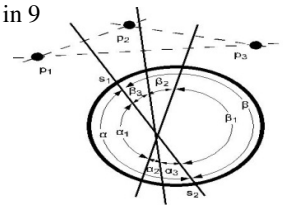
Denote the set of spatial objects by $\{p_1, p_2, \cdots, p_M\}$. The basic idea is to scan the objects one by one, and during each scan we maintain the set of arcs on Ω for each object to which this object is the 1st, 2nd, ⋯ , and k-th NN. The example shown in Figure 12a is used to illustrate the idea for a 2CRNN query. Suppose that there are three objects p_1, p_2, and p_3. Initially p_1 is the 1st NN to the circumference Ω. Then, p_2 is scanned, and the perpendicular bisector of p_1 and p_2 splits Ω into arcs α and β. As a result, p_2 takes over p_1 to be the 1st NN to β — p_1 is the 1st NN to α and the 2nd NN to β;

p_2 is the 1st NN to β and the 2nd NN to α. Next, p_3 is scanned and we check it against p_1 and p_2. The perpendicular bisectors further split α into α_1, α_2, and α_3, and split β into β_1, β_2, and β_3. Now p_3 takes over p_1 to be the 1st NN for α_3 and the 2nd NN for β_2; p_3 takes over p_2 to be the 2nd NN for α_2 and the 1st NN for β_1. In general, when object p_i is scanned, initially we assume that p_i is farther away from any arc than any candidate kCRNN result. Afterwards, we check p_i against each p_j in the candidate kCRNN result set. Given a p_j , the perpendicular bisector of p_j and p_i splits the existing arcs at two points at most. For each of the arcs located on the p_i side of the perpendicular bisector, p_j moves backward in the kNN list (e.g., the 2nd NN becomes the 3rd NN), while p_i advances in the kNN list (e.g., the 3rd NN becomes the 2nd NN). After each scan, those objects which have at least one l-th-NN arc ($l \leq k$) constitute the candidate set of kCRNN results; and if the order of some object p to an arc exceeds k, this arc is removed for p. The algorithm works by scanning the entire set of objects to obtain the final kCRNN results of Ω. Recall that the results of a kCRNN query also include all the objects in the circular region. Thus, when scanning the objects, the algorithm also checks whether they are in the circular region and if so, includes them in the final kCRNN results. Furthermore, in order to speed up the convergence of the kCRNN candidate set, we sort the objects and apply a heuristic (Heuristic 1) to scan the objects closest to Ω first because they are most likely to appear in the final kCRNN results.

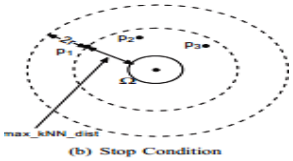
Heuristic 1: The objects are sorted and scanned in the increasing order of their minimum distances to Ω. And those objects whose minimum distances to Ω are 2r (r is the radius of Ω) farther than the k-th NN of some arc are removed from scanning.

The second statement of Heuristic 1 sets a stop condition for the scan, because those objects that are 2r farther from Ω than the current k-th NN of some arc must be farther away from any arc of Ω than the current kNNs of this arc. This can be explained in Figure 12b for a 3CRNN query. For the moment, p_1, p_2, and p_3 are the 3NNs of an arc of Ω and p_1 is the third NN. The minimum distance from p_1 to Ω, denoted by max kN N dist, is used to prune faraway objects in future search. More specifically, if any object is 2r+max kN N dist away from any point on Ω (i.e., outside the outermost circle in Figure 12b), the object does not need to be scanned since it must be farther away from any arc of Ω than p_1, p_2, and p_3.

The complete query processing algorithm is described in 9



(a) Query Processing



(b) Stop Condition

Finding kCRNN Results

Algorithm 2, where the data structure F(p_i, **ab**) maintains the order of object p_i to arc **ab**. We call it a *bulk* algorithm as all the candidate kCRNN results are finalized at the end of query evaluation. We now analyze the complexity of Algorithm 2. Given a kCRNN query with M objects, the while loop iterates through at most M scans. Since each scan increases the number of arcs by 2M in the worst case, the total number of arcs for all candidate kCRNN results is bounded by $O(M^2)$. Each arc can appear in the arc sets of at most k objects. Thus, the worst-case storage complexity is $O(kM^2)$. For the time complexity, each F(p_j , **ab**) entry may be updated at most M times, once in each scan. Hence, the worst-case time complexity is $O(kM^3)$. Nevertheless, in practice the cost would be far less because the candidate set of kCRNN results is normally not large and the scanning may terminate early with the stop condition proposed in Heuristic 1.

B. Progressive Query Processing of kCRNN

The bulk query processing algorithm generates the kCRNN results at the end of query evaluation. Therefore, the server cannot start transmitting the results to the client until the end of query evaluation. We now propose an alternative *progressive* query processing algorithm to parallelize the query evaluation and result transmission.

The idea is to determine whether an object will be a final kCRNN result earlier. The query progressing procedure remains the same as in Algorithm 2 except that i) any object in in circle results is immediately

returned to the client when it is scanned, and ii) after the scan of each object, we add a checking procedure (see Algorithm 3). We randomly pick an

Algorithm 2 Bulk Query Processing of kCRNN

Input: query circle Ω with radius r, spatial object set $\mathcal{S}$
Output: the kCRNN results of Ω
Procedure:

1: enqueue the objects in $\mathcal{S}$ into a priority queue in increasing order of their (minimum) distances to Ω (denoted by $minDist(p_i, \Omega)$)
2: dequeue the first object p_1
3: $\mathcal{F}(p_1, \Omega) := 1$ // $\mathcal{F}(p_i, \widehat{ab})$ maintains the order of p_i to $\widehat{ab}$
4: $cand\_kCRNN\_results := \{p_1\}$
5: $max\_kNN\_dist := \infty$
6: dequeue the next object p_i
7: **while** $minDist(p_i, \Omega) < 2r + max\_kNN\_dist$ **do**
8: **if** p_i is inside Ω **then**
9: $in\_circle\_results := in\_circle\_results \cup p_i$
10: initialize $\mathcal{F}(p_i, \widehat{ab}) := |cand\_kCRNN\_results| + 1$
for any arc $\widehat{ab}$ // initially assuming p_i is farther
// than any candidate kCRNN result
11: **for** each object p_j in $cand\_kCRNN\_result$ **do**
12: split existing arcs by $\perp p_i p_j$ — the perpendicular bisector of p_i and p_j
13: **for** any arc $\widehat{ab}$ located on p_i side of $\perp p_i p_j$ **do**
14: **if** entry $\mathcal{F}(p_j, \widehat{ab})$ exists **then** $\mathcal{F}(p_j, \widehat{ab})$++
// move p_j backward in the kNN list
15: $\mathcal{F}(p_i, \widehat{ab})$-- // move p_i forward in the kNN list
16: let S be the set of scanned objects including p_i;
$cand\_kCRNN\_results :=$
$\{p \mid p \in S, \exists$ an arc $\widehat{ab}, \mathcal{F}(p, \widehat{ab}) \leq k\}$
17: remove $\mathcal{F}()$ entries with $\mathcal{F}(p, \widehat{ab}) > k$ for $p \in S$
18: $max\_kNN\_dist := \min\{max\_kNN\_dist,$
$\min_{\forall \widehat{ab}}\{minDist(p, \Omega) \mid p$ is the k-th NN of arc $\widehat{ab}\}\}$
19: dequeue the next object p_i
20: return $in\_circle\_results \cup cand\_kNN\_results$ as the final results

Unchecked split point on Ω as the check point, and go through the list of unscented objects to compute its full kNN results. If any of the kNN results has not been returned to the client, it is output for immediate transmission. For our running example shown in Figure 12a, after scanning p_2, we may select s_1 as the check point. We then compute s_1's 2NN results as p_1 and p_2 and return them immediately. Compared to the bulk query processing, since the checking procedure here incurs extra overhead, the overall query processing time would be in-creased. Nevertheless, the worst-case time complexity remains $O(kM^3)$ since the checking procedure adds a complexity of $O(kM^2)$ only. On the other hand, the progressive algorithm can start returning the kCRNN results earlier and, hence, likely to result in a shorter user-perceived response time, as will be demonstrated in Section VII-D.

5. PERFORMANCE EVALUATION

A. Experiment Setup

We have developed a tested [9] to evaluate the performance of the proposed location cloaking and query processing algorithms. The client-side query interface and location cloaking 10

Algorithm 3 Checking Procedure in Progressive Query Processing of kCRNN

Input: $\mathcal{F}(p_i, \widehat{ab})$ entries, the queue of unscanned objects
Output: the kNN results of a check point
Procedure: // this procedure is added to between lines
// 17 and 18 of Algorithm 2

1: randomly select an unchecked split point s as the check point
2: retrieve the tentative kNN results of s: $\{p \mid \mathcal{F}(p, \widehat{ab}) \leq k, s \in \widehat{ab}\}$
3: $current\_kNN\_distance :=$ distance of the current k-th NN
4: dequeue the next object p_i
5: **while** $minDist(p_i, \Omega) < current\_kNN\_distance$ **do**
6: **if** $Dist(p_i, s) < current\_kNN\_distance$ **then**
7: update the tentative kNN results
8: update $current\_kNN\_distance$
9: dequeue the next object p_i
10: return the final kNN results if not yet

Parameter	Setting
kNN Query (k)	5
Query Interval (I)	4 *min*
Privacy Requirement (r)	0.001
Spatial Object Set Size	2,249,727 objects
Object Record Size	1 *kb*
Query Size	20 *bytes*
Data Transfer Rate	114 *kbps*

TABLE I

DEFAULT PARAMETER SETTINGS

Agent was implemented on an O2 Xda Atom Exec PDA with Intel PXA 27x 520 MHz Processor and 64 MB RAM. The PDA supports GSM/GPRS/EGDE and WiFi communications. The LBS server was implemented on a Redhat 7.3 Linux server with Intel Xeon 2.80 GHz processor and 2 GB RAM. We assume that the client and the server communicate through a wireless network at a data transfer rate of 114 kbps.

The spatial object set used in the experiments contains 2,249,727 objects representing the centroids of the street segments in California [29]. We normalize the data space to a unit space and index the spatial objects by an R-tree (with a page fanout of 200 and a page occupancy of 70%) [6]. the size of an object record is set at 1 kb. To process a kCRNN query of a circle Ω, we first use our previously developed disk-based access method [9] to retrieve the kNN results for the minimum

bounding rectangle of Ω. By definition, this set of kNN results is a superset of Ω's kCRNN results. The kCRNN processing algorithms developed in Section VI are then applied on this superset to get the kCRNN results of Ω.

We simulate a well-known random walk model [11], in which the user moves in steps. In each step, the user selects a speed and travels along an arbitrary direction for a duration of 2 min. We test two speed settings: 1) constant speed: the moving speed is fixed at 0.0003 / min; 2) random speed: for each step, a speed is randomly selected from a range of [0.0001 /min, 0.0005 /min]. By default, the random speed setting is adopted. The user makes privacy-conscious kNN queries from time to time. The query interval I is set at 4 min by default. The user specifies the privacy requirement by a radius r (i.e., the minimum acceptable cloaking area is πr^2), with a default setting of 0.001. The size of a kNN query message is set at 20 bytes. For the numerical method of optimal location cloaking, is set at 0.0001, and the Simplex algorithm is employed to solve the linear program. The default parameter settings are summarized in Table I. The experimental results reported below are averaged over 1,000 randomly generated queries.

B. Effectiveness of Mobility-Aware Cloaking

In this section, we compare the proposed optimal mobility-aware cloaking technique (Algorithm 1) against the isolated cloaking scheme (described at the beginning of Section V). For both the optimal and isolated cloaking techniques, initially a cloaking region is randomly generated based on the user location. In other words, the user is equally likely to be at any location in the cloaking region. We measure the quality of the cloaking region for a subsequent query in terms of entropy based on 1,500 sample locations and 1,000 random queries. As shown in Figures 13a and 13b, when the query interval is small (i.e., 1 min), the entropy of isolated cloaking is nearly 20% lower than that of optimal cloaking for all queries tested. With increasing query interval, the average entropy of isolated cloaking improves (see Figure 13c) but is still far lower than that of optimal cloaking. When the query interval is 8 min, Figure 13a and 13b show that the entropy of isolated cloaking is 40% lower than that of optimal cloaking for over 15% of the queries tested and 20% lower for over 40% of the queries tested. Note that the results shown here are for one successive query only with more successive queries; the quality of isolated cloaking would further degrade.

To highlight the benefit of achieving higher entropy, we conduct two possible attacks. Recall that through trace analysis attacks, the LBS server can derive the probabilities of the user being at different locations in a cloaking region. The first attack attempts to limit the possible user location to a sub-region with 95% confidence. The second attack calculates s the highest aggregate probability for any sub-region with size equal to 5% of the cloaking region. Figures 14a and 14b show the results when the number of cloaking regions used in trace analysis attacks is increased from 1 to 10. We can see from the results that our optimal cloaking is robust against the attacks: for example, with the first attack (Figure 14a), the sub-region size is as large as 95% of the cloaking region since the derivable probability for the user to be at any location is uniform across the region. In contrast, with the same level of confidence, the sub-region size under isolated cloaking could be much smaller due to a skewed probability distribution (see Figure 14c for a sample distribution we observed in the experiment). Similarly, with the second attack (Figure 14b), under isolated cloaking, the server would be able to derive the probability for the user to be in a sub-region of 5% size of the cloaking region with a confidence of 16%-99%. The confidence for the same sub-region is only 5% under optimal cloaking.

C. Comparison of Mobility-Aware Cloaking Algorithms

This section compares the two cloaking algorithms devel-oped in Section V based on the optimal cloaking technique, namely MaxAccu Cloak (abbreviated as *MaxAccu*) and Min-Comm Cloak (abbreviated as *MinComm*). Recall that Max-Accu aims at higher query accuracy by minimizing the outer query blocking probability while MinComm attempts to achieve a balance between communication cost and query accuracy by maximizing the inner query blocking probability at the same time.

As shown in Figure 15a, MaxAccu has an outer query blocking probability of zero. Hence, its query results are 100% accurate as shown in Figure 15b. In contrast, MinComm has an outer query blocking probability of about 15%. For those blocked outer queries, approximate results are obtained based on cached result supersets. Figure 15b shows that the average error (measured by the ratio of the distance of an approximate kNN result to the actual kNN distance) is pretty small. In the worst case, the approximate kNN

distance is no more than 2.3 times of the actual distance.

Figure 15a also shows that MinComm has a much higher inner query blocking probability than MaxAccu. Recall that inner queries are sent to the server for evaluation merely for the purpose of optimal cloaking. They do not affect query accuracy but communication cost. With more queries (including both inner and outer queries) being blocked, the communication cost incurred by MinComm is about half that of MaxAccu (see Figure 15c).

D. Comparison of k*CRNN Query Processing Algorithms*

In this section, we evaluate the performance of the *bulk* and *progressive* kCRNN query processing algorithms developed in Section VI. Figure 16 shows the user-perceived response time for both algorithms. When k or r is small, the bulk and progressive algorithms perform similarly. However, when k or r is large, the progressive algorithm clearly outperforms the bulk algorithm. To explain, we show in Figures 17a and 17b the timeline performance of two sample queries with $r=0.25\times10^{-3}$ and $r=4\times10^{-3}$, respectively. For the query with $r=0.25\times10^{-3}$ (Figure 17a), both the bulk and progressive algorithms took a short time to process. Thus, parallelizing the query evaluation and result transmission does not help a lot in user-perceived response time. On the other hand, when $r=4\times10^{-3}$ (Figure 17b), the result superset size is large and the query requires a long time (over 1000 ms) to evaluate; then by returning the kCRNN results incrementally, the progressive algorithm completes the result transmission earlier.

6. CONCLUSION

This paper has presented a complete study on processing privacy-conscious location-based queries in mobile environments. The technical contributions made in this paper are summarized as follows:

• We have studied the representation of cloaking regions and showed that a circular region generally leads to a small result superset.

• We have developed an optimal mobility-aware location cloaking technique to resist trace analysis attacks. Two cloaking algorithms, namely MaxAccu Cloak and Min Comm Cloak, have been designed to favor different performance objectives.

• We have developed two efficient polynomial algorithms, namely bulk and progressive, for processing circular-region-based kNN queries. We have also conducted simulation experiments to evaluate the proposed algorithms. Experimental results show that the optimal mobility-aware cloaking algorithms is robust against trace analysis attacks without compromising much on query latency or communication cost. MaxAccu Cloak gets 100% query accuracy while MinComm Cloak achieves a good balance between communication cost and query accuracy. It is also shown that the progressive query processing algorithm generally achieves a shorter user-perceived response time than the bulk algorithm.

As for future work, we are going to extend the mobility-aware location cloaking technique to other privacy metrics (e.g., the k-anonymity model and the l-diversity model) and road networks. We are also interested in investigating mobility-aware peer-to-peer cloaking techniques.

REFERENCES

[1] R. Cheng, Y. Zhang, E. Bertino, and S. Prabhakar Preserving user location privacy in mobile data management infrastructures Privacy Enhancing Technology Workshop, Cambridge, UK, June 2006

[2] K. Cheverst, N. Davies, K. Mitchell, and A. Friday Experiences of developing and deploying a context-aware tourist guide: the GUIDE project.ACM MobiCom, 2000.

[3] M. Gruteser and D. Grunwald Anonymous usage of location-based services through spatial and temporal cloaking ACM MobiSys, 2003

[4] B. Gedik and L. Liu. A customizable k-anonymity model for protecting location privacy ICDCS, 2005

[5] B. Gedik and L. Liu. Protecting location privacy with personalized k-anonymity: Architecture and algorithms IEEE Transactions on Mobile Computing, 7(1):1–18, 2008.

[6] A. Guttman. R-trees: A dynamic index structure for spatial searching. SIMGOD1984.

[7] J. I. Hong and J. A. Landay. Anarchitecture for privacy-sensitive ubiquitous computing ACM MobiSys, 2004

[8] Y. C. Hu and H. J. Wang Location privacy in wireless networks.ACM SIGCOMM Asia Workshop, April 2005

[9] H. Hu and D. Lee Range nearest neighbor queries TKDE, 18(1):78–91, 2006.

[10] H. Hu, J. Xu, W. S. Wong, B. Zheng, D. L. Lee, and W.-C Lee. Proactive caching for spatial queries in mobile environments ICDE, 2005

[11] Barry D. Hughes. Random Walks and Random Environments. Oxford University Press 1996

[12] X. Liu, M. D. Corner, and P. Shenoy Ferret: RFID locationing for pervasive ultimedia. Proc. Ubicomp, Irvine, CA, Sept. 2006.

[13] P. Kalnis, G. Ghinita, K. Mouratidis, and D Papadias Preserving anonymity in location based services. Technical Report, School of Computing, the National University of Singapore, 2006

[14] H. Kido, Y. Yanagisawa, and T. Satoh An anonymous communication technique using dummies for location-based services Intl. Conf. on Pervasive Services (ICPS), 2005

[15] M. F. Mokbel, C. Y. Chow and W. G. Aref. The New Casper: Query procesing for location services without compromising privacy. VLDB 2006

[16]. J. Xu, B. Zheng, W. C. Lee, and D. L. Lee. Energy-efficient index for querying location-dependent data in mobile broadcast environments ICDE, 2003

[17]. B. Schilit, J. Hong, and M. Gruteser. Wireless location privacy protection. IEEE Computer, Dec. 2003