Efficient Algorithms for K-Anonymous Location Privacy in Participatory Sensing

Khuong Vu and Rong Zheng
Department of Computer Science
University of Houston
Houston, TX 77204
E-mail: {*khuong.vu, rzheng*}@*cs.uh.edu*

Jie Gao
Department of Computer Science
Stony Brook University
Stony Brook, NY 11794
jgao@cs.stonybrook.edu

***Abstract*—Location privacy is an important concern in participatory sensing applications, where users can both contribute valuable information (*data reporting*) as well as retrieve (location-dependent) information (*query*) regarding their surroundings. K-anonymity is an important measure for privacy to prevent the disclosure of personal data. In this paper, we propose a mechanism based on locality-sensitive hashing (LSH) to partition user locations into groups each containing at least K users (called spatial cloaks). The mechanism is shown to preserve both locality and K-anonymity. We then devise an efficient algorithm to answer kNN queries for any point in the spatial cloaks of arbitrary polygonal shape. Extensive simulation study shows that both algorithms have superior performance with moderate computation complexity.**

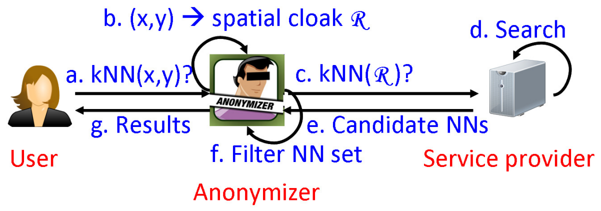


Fig. 1: Framework for K-anonymous location privacy. kNN stands for the k-nearest neighbor query. NN stands for "nearest neighbor".

I. Introduction

With the proliferation of mobile devices loaded with rich sensory peripherals, participatory sensing – outsourcing sensing tasks to a large group of mobile users (a crowd) – has gained much attention in a variety of applications including, real-time traffic and road monitoring, reporting spots of oil spill, finding the best biking routes, and scoring 3G broadband services, etc. In participatory sensing, a user can both contribute valuable information (*data reporting*) as well as retrieve (location-dependent) information (*query*). Privacy is an important issue in data sharing. In participatory sensing, privacy concerns arise from two aspects. The first is in the data reporting process. It is often desirable to build an understanding/a model of the sensed environment without the precise knowledge of individual's information. Many techniques have been proposed in literature by transforming the data (e.g., adding noise [2], fitting [3] etc.) The second is in the query process, where a user sends location sensitive queries regarding his surroundings (e.g., "where is the closest pub?"). Location privacy mainly concerns with two objectives: hide user locations, and hide user identities, which avoids association of users with their activities (e.g., "who is requesting the nearest pub?"). Our work deals with the latter.

K-anonymity is a measure of privacy first introduced by Sweeney et al. [20] to prevent the disclosure of personal data. A table satisfies K-anonymity if every record in the table is indistinguishable from at least $K-1$ other records with respect to every set of quasi-identifier attributes. In the context of location privacy, the location attribute can be viewed as a quasi-identifier. K-anonymous location privacy thus implies that the user's location is indistinguishable from at least $K-1$ other users. To achieve K-anonymous location privacy, one common approach is to incorporate a trust server, called *the anonymizer* who is responsible for removing the user's ID and selecting an anonymizing spatial region (ASR) containing the user and at least $K-1$ users in the vicinity (Figure 1(b)). Another purpose of ASR is to reduce the commnication cost between the anonymizer and the service provider, and the processing time at service provider side. This process is also called "cloaking" as it constructs a spatial cloak around the user's actual location. The anonymizer forwards the ASR along with the query to the (untrusted) location based service (LBS) (Figure 1(c)), which processes the query and returns to the anonymizer a set of candidate point of interests (POIs) (Figure 1(e)). The anonymizer removes the false hits and forwards the actual result to the user (Figure 1(f)(g)).

As shown in Figure 1, in achieving K-anonymous location privacy, it is crucial to devise quality spatial cloaks at the anonymizer and efficient searching algorithms at the LBS. Intuitively, the cloaks produced should be *locality preserving* – close to the user location, and small in size since both the computational complexity of the search algorithms and the number of POIs returned increases with the size of the cloak. In this paper, we make the following contributions in K-anonymous location privacy for participatory sensing applications:

- **Locality-preserving cloaking:** We utilize locality-sensitive hashing (LSH) [5] to project the location data to a high-dimension space, which is then partitioned into cells that contain at least K users. LSH has the property that location proximity is preserved during the mapping.

- **Efficient and flexible search algorithm:** We devise a search algorithm for finding k-nearest POIs of simple polygonal cloaks that takes $O(\log n + Kn + m)$ worst-case running time where m is the number of vertices in the polygonal cloak, n is the number of the POIs, and K is the anonymity level. (In fact, the exact running time is $O(\log n + e + m)$, where $e << O(Kn)$, as shown later.) Contrary to the general belief that complex cloak shapes drastically increase the running time of the k-nearest neighbor (kNN) search, the complexity of our proposed algorithm depends only linearly on m.

The rest of the paper is organized as follows. In Section II, we introduce the terminology used and the attacker model. The LSH-based cloaking algorithm is described in Section III, and the search algorithm is presented in Section IV. In Section V, we present evaluation results. In Section VI we provide a review of related work for spatial anonymization. Finally, we conclude the paper in Section VII with the list of future work.

II. Background

In this section, we first introduce necessary terminologies used throughout the paper and the attacker model, then we give an overview of Voronoi diagram (VD), which is used in the proposed searching algorithm.

A. Attacker model

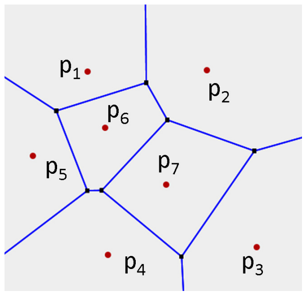
Similar to [11], we assume that an attacker can i) intercept the ASR, ii) know the cloaking algorithm used by the anonymizer, and iii) obtain the up-to-date locations of all users. The first assumption implies that either the LBS is not trusted, or the communication channel between the anonymizer and the LBS is not secure. The second assumption is common in the literature since the data security techniques are typically public. The third assumption is motivated by the fact that users often issue queries from the same locations (home, office), which could be identified through physical observation, triangulation, telephone catalogs, etc.

B. K-anonymity and reciprocity

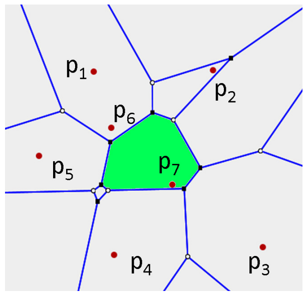
We consider N users distributed on a 2-D bounded area $\mathcal{B}$. The set of user locations is denoted by S. The proposed methodologies can be easily extended to higher-dimensional space. We assume queries are one-time (or snapshot queries) such that the attackers cannot utilize historical data to make further inference. Privacy in publishing trajectory data has been considered in [21] and is out of the scope of this paper.

K-anonymity is satisfied if the attacker can identify the user that issues a query with probability not exceeding $1/K$. Reciprocity is introduced by Kalnis [11] as a sufficient condition for K-anonymity as follows:

Definition 1: Consider a user U issuing a query with anonymity degree K, and anonymizing spatial region ASR. ASR satisfies reciprocity if (i) it contains U and at least $K-1$ additional users, and (ii) every user in ASR also generates the same ASR for the given K.



(a) Order-1 VD.



(b) Order-2 VD. Shaded is the Voronoi cell corresponding to sites p_6 and p_7.

Fig. 2: An example of high order VDs.

Reciprocity necessarily implies a fixed partition of $\mathcal{B}$ such that every partition contains at least K users forming the cloak of the associated users. Though reciprocity is not necessary to ensure K-anonymity, it is easy to verify and has been widely adopted in literature.

C. High-order point Voronoi diagrams

The order-1 VD of a set of points, also known as sites, in the plane is a tessellation that divides the plane into non-overlapping regions called *Voronoi cells* (or cells for short), each corresponding to a site. A Voronoi cell is the locus of points that are closer to the corresponding site than to the others. Similarly, if we divide the plane into regions, each being the locus of points closer to a set of k sites than to the others, we have an order-k VD. Figure 2 gives examples of order-1 and order-2 VDs of 7 sites. As shown in Figure 2b, the shaded area is an order-2 Voronoi cell corresponding to sites p_6 and p_7. The distance from any point p in this cell to p_6 and p_7 is smaller than those to other sites. In other words, $d(p, p_6) \leq d(p, p_i)$ and $d(p, p_7) \leq d(p, p_i)$, where $p_i \neq p_6, p_7$, and $d(\cdot, \cdot)$ is the Euclidean distance between two points.

Construction of high order VDs requires identifying groups of sites, whose high order Voronoi cells are not empty. In the example shown in Figure 2, there exists no point p such that $d(p, p_2) \leq d(p, p_i)$ and $d(p, p_5) \leq d(p, p_i)$ for $p_i \neq p_2, p_5$. Therefore, the order-2 cell corresponding to $\{p_2, p_5\}$ does not exist. In his seminal paper [14], Lee proposed an incremental approach to construct order-k VDs of n sites in $O(k^2 n \log n)$ time. Due to its incremental nature, Lee's algorithm in fact constructs k diagrams from order-1 to order-k.

Order-k VDs can be used to answer kNN query efficiently. Given a query point q, the k nearest neighbors correspond to sites of the order-k Voronoi cell that contains q. Locating a point in a Voronoi diagram of n sites takes $O(\log n)$ time by various techniques [6], [13].

III. LSH based cloaking

As discussed in Section II, given a query from location q, the anonymizer needs to construct a spatial cloak that contains q and $K-1$ other user locations. To achieve reciprocity, the anonymizer first partitions all user locations into non-overlapping buckets each containing at least K users. Then, user locations in the bucket containing q are enclosed in a geometric shape that is locality-preserving. In this section, we first show by an example that satisfying both locality-preservation and K-anonymity is not trivial. Then, we propose a LSH based approach for cloaking.

A. A naive approach to cloaking

In order-K VDs, each cell corresponds to a set of K sites that are spatially close. Therefore, order-K VD cells appear to be natural candidates for cloaking. Specifically, we first construct an order-K VDs of all user locations S. A cloak can then be formed from a set of sites H of a non-empty cell, which is randomly chosen such that H contains the query location. Consider the example shown in Figure 3. There are 11 user locations in the field. Assume user location p_3 requires 2-anonymity. The order-2 VD of S is given in Figure 3. A cell is randomly chosen whose sites contain p_3. Without loss of generality, let the resultant cell be $H = \{p_6, p_3\}$[1]. H constitutes a cloak for p_3. Unfortunately, this approach does not satisfy K-anonymity if the attacker knows all user locations and the cloaking algorithm. In this case, the probability that the attacker can correctly guess p_3 can be derived using the Bayes formula:

$$\begin{aligned}\mathbb{P}(p_3|H) &= \frac{\mathbb{P}(H|p_3)\times\mathbb{P}(p_3)}{\mathbb{P}(H)}\\ &= \frac{\mathbb{P}(H|p_3)\times\mathbb{P}(p_3)}{\sum_{p_i\in S}\mathbb{P}(H|p_i)\times\mathbb{P}(p_i)}\end{aligned}$$

The numbers of cells whose sites contain p_3 and p_6 are 4 and 5, respectively. Therefore, $\mathbb{P}(H|p_3) = 1/4$ and $\mathbb{P}(H|p_6) = 1/5$. Additionally, $\mathbb{P}(H|p_i) = 0$ for $p_i \neq p_3, p_6$. We further assume that each user has equal probability to issue the query, and thus $\mathbb{P}(p_3) = 1/11$. Hence,

$$\begin{aligned}\mathbb{P}(p_3|H) &= \frac{\frac{1}{4}\times\frac{1}{11}}{\frac{1}{11}\times(\frac{1}{4}+\frac{1}{5})}\\ &> 1/2\end{aligned}$$

Clearly, 2-anonymity is violated. The main reason is that different user locations may contribute to different numbers of cells in the order-K VD. Thus, although the approach is locality-preserving, it does not satisfy K-anonymity. This motivates us to seek for a method that is both locality-preserving and K-anonymous.

B. Locality-sensitive hashing based approach to reciprocity

A straightforward approach to reciprocity is to partition the user locations into buckets of adjacent points. For instance, given S, we randomly choose a point p and group it with its $(K-1)$ nearest neighbors. The process continues until all points are assigned to some buckets. This approach has two disadvantages. First, it fragments the dataset and is not locality

[1]For the ease of presentation, we abuse the notation and use H to refer to both the set of sites and the Voronoi cell.

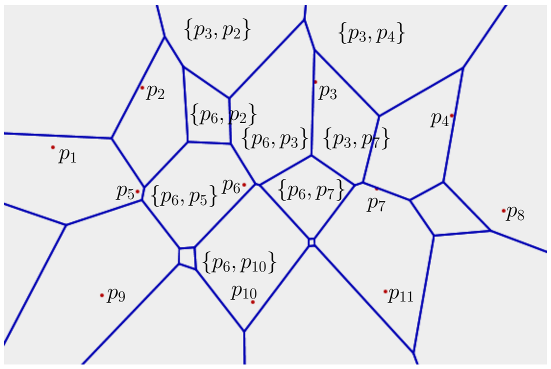


Fig. 3: The order-2 Voronoi diagram of 11 locations. $\{\cdot,\cdot\}$ shows the locations corresponding to a cell.

preserving. Points in the same bucket may be not neighbors in the original dataset, especially for large K's. This is illustrated in Figure 4(a). In the example, $K = 4$ and point p is chosen initially. Three neighbors of p then form a cloak, shown in the rectangle. After the four points are removed from the dataset, the cloak of the remaining points is large. In this example, the partition as shown in Figure 4(b) is clearly more desirable. Second, the time complexity is high. It is easy to see that the method takes $O(\frac{n^2}{K}\log(\frac{n^2}{K}))$ running time. We show shortly an approach that achieves desirable partitions with lower time complexity thanks to locality-sensitive hashing (LSH) [5].

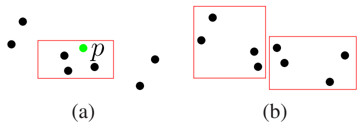


Fig. 4: Fragmentation in the naive nearest neighbor partition.

LSH hashes the input data so that similar points are mapped to the same buckets with high probability. Formally, for a domain S, a function family $\mathcal{H} = \{h : S \to U\}$ is called (r_1, r_2, p_1, p_2) sensitive with distance measure D if for any $v, q \in S$: if $d(v, q) < r_1$ then $\mathcal{P}_{\mathcal{H}}[h(q) = h(v)] \geq p_1$; if $d(v, q) > r_2$ then $\mathcal{P}_{\mathcal{H}}h(q) = h(v) \leq p_2$, where $p_1 > p_2$, $r_1 < r_2$, and $d(v, q)$ is the distance between v and q in D. There are several LSH families. In this work, we use the LSH family based on p-stable distributions [24] proposed in [5]. The idea of the LSH scheme is as follows. Consider two input vectors v_1 and v_2, and a vector a whose entries are chosen independently from a p-stable distribution $\mathbf{X}$. The distance between their projections on a, $(a \cdot v_1 - a \cdot v_2)$, is distributed as $||v_1 - v_2||_p\mathbf{X}$. By dividing the projected points into equal-width buckets of size r, vector a gives rise to a locality-sensitive hash function, where $h_a(q) = q \cdot a$. However, though similar points are hashed into the same bucket with high probability, the reverse does not hold, i.e., a bucket may contain faraway points. A solution to this problem is to use multiple hash functions. That is, q is hashed by L functions

$g_l(q) = \langle h_1(q) \rangle, l = 1, 2, \ldots, L$.[2] Multiple hash functions lead to a better separation of the data points as illustrated in the example in Figure 5. The projections of points p_1, p_2, p_3, and p_4 in the plane onto line a are close, while those corresponding to line b are more separate. The use of the two vectors a and b maps p_1 and p_2 into the same bucket, and p_3 and p_4 to another bucket. It has been shown in [5] that given an error rate ε, L can be chosen such that r-near neighbor queries are answered correctly with the error rate lower than ε.

LSH is useful in devising spatial cloaks due to its locality-preserving property. Instead of finding r-near neighbors, the canonical applications of LSH, we wish to partition the data set into groups of at least K elements. Since the partition using a single hash function may contain many distant points, we use L hash functions $g_1, g_2, \ldots, g_L$ instead. The LSH-based partitioning algorithm is summarized in Algorithm 1. We first build sorted lists $l_1, \ldots, l_L$ of hashed values of S from the L hash functions. Then, each list is partitioned into buckets each containing K elements (the last one may contain more than K elements). To avoid fragmentation, we always start from the first available point q on l_1. q's $(K-1)$ nearest neighbors are extracted from $U(q)$ the union of the L buckets containing q in the respective lists. Due to the properties of LSH, q's nearest neighbors are in $U(q)$ with high probability, and, moreover, the size of $U(q)$ is not high. In our implementation, the 2-stable Gaussian distribution proposed in [5] is adopted for 2-D location data.

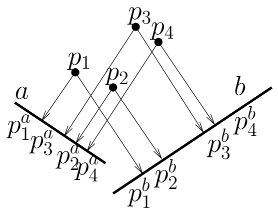


Fig. 5: Hashing 4 points in the plane with 2 hash functions, a and b.

We now analyze Algorithm 1's running time. Lines 1 to 3 sort the data set in L lists, which costs $O(L\,n \log n)$, where $n = |S|$ is the cardinality of the dataset. Line 12 requires sorting elements from L buckets, which takes at most $O(L\,k \log Lk)$. (Note that the exact number of distinct elements is generally lower). While loop (lines 4 to 14) executes $\lfloor n/k \rfloor$ times. Therefore, the worst-case running time complexity of Algorithm 1 is $O(L\,n \log n)$. Since number of distinct elements sorted in line 12 is low, Algorithm 1's running time is expected to be inversely proportional to anonymity level K.

User locations produced by Algorithm 1 are then used to form spatial cloaks. The search algorithm introduced in Section IV allows simple polygon cloaks as input. Convex polygonal cloaks are popular since they are locality-preserving, and more importantly, the complexity of kNN queries of convex polygonal cloaks is roughly proportional to the number of edges. However, forming the convex cloak of k users takes $O(K \log K)$ time, which only needs to be done one time as long as the user locations do not changes. Alternatively, the minimum bounding rectangle can be constructed in $O(K)$ time to form the spatial cloaks [22].

[2]In the original LSH scheme, each hash function maps a data point in $\mathcal{R}^d$ to the $\mathcal{R}^k$. Here we choose $k = 1$.

Algorithm 1: LSH-based location partitioning

```
Data: A set of points $S$, $K$-anonymity
Result: $T$, a partition of $S$ into groups of size $K$, except for
        the last one
1  generate $L$ hash functions, each is a vector whose entries are
   chosen from a Gaussian distribution;
2  compute and maintain $L$ sorted lists $\{l_1, \ldots, l_L\}$ of hash values
   of $S$;
3  $T \leftarrow \emptyset$;
4  while sorted lists are not empty do
5      Partition $l_i$ into buckets of size $K$, $i = 1, \ldots, L$;
6      $q \leftarrow$ the first element of $l_1$ ;
7      $\Omega \leftarrow \emptyset$;
8      for $i = 1$ to $L$ do
9          $b \leftarrow$ the bucket containing $q$ in $l_i$;
10         $\Omega \leftarrow \Omega \bigcup b$;
11     end
12     $NNs \leftarrow q \bigcup (K-1)$ nearest neighbors of $q$ in $\Omega$;
13     $T \leftarrow T \bigcup NNs$;
14     remove elements of $NNs$ from the $L$ sorted lists ;
15 end
16 return $T$;
```

IV. k-Nearest Neighbor Search for Polygonal Cloaks

In this section, we first give the necessary and sufficient condition for determining the kNN of a polygonal cloak; and then propose a kNN search algorithm. The algorithm can be easily extended to circular cloaks and be omitted due to space limit.

A. Necessary and sufficient conditions for the kNN set

A spatial region C is said to *intersect* with a cloak $\mathcal{R}$ if there exists a point p that is interior to both $\mathcal{R}$ and C, and a point p' that is interior to C but exterior to $\mathcal{R}$. In other words, C intersects with $\mathcal{R}$ iff $C \not\subset \mathcal{R}$ and $\mathcal{R} \cap C \neq \emptyset$. C is *inside* $\mathcal{R}$ if $C \subset \mathcal{R}$. We use the term *overlap* when C intersects with $\mathcal{R}$ or lies completely inside $\mathcal{R}$.

Given a cloak $\mathcal{R}$, we search for the set $\mathcal{P}$ of POIs that contains the set of k-nearest POIs of any location in the cloak, i.e., $\mathcal{P}$ should be *sufficient*. Moreover, $\mathcal{P}$ should be *necessary*, that is, any POI in $\mathcal{P}$ must be in the set of k nearest POIs of some location in the cloak. By the definition of order-k VDs (Section II), the set of sites associated with the Voronoi cells that intersect with $\mathcal{R}$ must be in $\mathcal{P}$. Next, we show formally it is both necessary and sufficient to consider these order-k VD cells.

Lemma 1: Consider a spatial cloak $\mathcal{R}$ of a query point q. Let U be the set of POIs associated with the Voronoi cells of the order-k VD of POIs that intersect with $\mathcal{R}$. U uniquely characterizes the candidate set for the k nearest POIs of q.

Proof: To show that U is necessary, we prove by contradiction that no POI can be removed from U. Assume $u \in U$ can be removed from U. By the construction of U, there exists a cell C of sites H such that $u \in H$ and $C \cap \mathcal{R} \neq \emptyset$. Choose any point $p \in C$. Clearly, u is one of the k-nearest neighbor of p, a contradiction.

To show that U is sufficient, consider a POI $u \notin U$ and is one of the k-nearest neighbor of some point $p \in \mathcal{R}$. By the definition of order-k VDs, there exists a cell C corresponding to sites H such that $u \in H$ and $p \in C$. Thus, we have a contradiction. ∎

B. Search algorithm

Lemma 1 establishes that to determine the kNN of a cloak $\mathcal{R}$, it suffices to identify the order-k cells that overlap with $\mathcal{R}$, and take the union of their corresponding POIs. Next, we first give a straw-man approach that has high computation complexity and then present a more efficient algorithm that utilizes order-1 VDs.

To find the order-k cells overlapping with $\mathcal{R}$, one can start with one such cell, say, C and iteratively explores its neighboring cells that overlap with $\mathcal{R}$. The procedure stops when no such neighboring cells can be found. To find C, we choose an arbitrary point p in $\mathcal{R}$ and query the cell that contains p. Given an order-k VD of n sites, this takes $O(\log n)$. Testing the overlap of a cell with the cloak involves checking the relative position of the cell's edges with respect to the cloak. Let $T(m)$ be the running time of testing whether an edge intersects with $\mathcal{R}$, where m is the number of vertices of $\mathcal{R}$. This procedure takes $O(\log n + e_k \cdot T(m) + s)$, where e_k is the number of Voronoi edges of the order-k cells that *overlap* with the cloak, and s is the number of POIs returned. Clearly, e_k increases as $\mathcal{R}$ gets larger. In Section V, we implement this method (called naive search) as a baseline for comparison purposes.

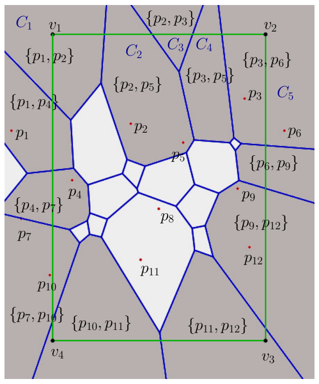
In our proposed approach, we reduce the complexity of the above procedure by considering cells that intersect with $\mathcal{R}$ and cells that are inside $\mathcal{R}$ separately. Furthermore, we show that in the latter case, it suffices to examine order-1 cells, which is generally k times less than the number of order-k cells. To see so, let us consider an example in Figure 6, which asks for the 2-nearest POIs of a rectangle cloak (in green). Figures 6(a) and (b) show respectively order-2 and order-1 Voronoi diagrams of 12 POIs, $\mathcal{P} = \{p_1, \ldots, p_{12}\}$, which are also the candidate POIs of the cloak. In Figure 6a, the POIs given by the cells intersecting with the cloak (darker) is $U_1 = S \backslash \{p_8\}$, while the 11 cells inside $\mathcal{R}$ (white) only contribute one additional POI, namely, p_8. On the other hand, the set of POIs associated with the two order-1 cells inside the cloak is $U_1 = \{p_5, p_8\}$. Obviously, $U_1 \bigcup U_2$ gives all candidate 2-nearest POIs. However, we observe the number of order-1 cells inside the cloak is much smaller (and thus requires less time to identify). We state the results formally as follows:

Lemma 2: Given a simple polygon cloak $\mathcal{R}$ and the order-k VD of a set $\mathcal{P}$ of POIs, $V^k(\mathcal{P})$. Let $I \subset \mathcal{P}$ be the set of sites whose order-1 cells are inside $\mathcal{R}$. The following holds:

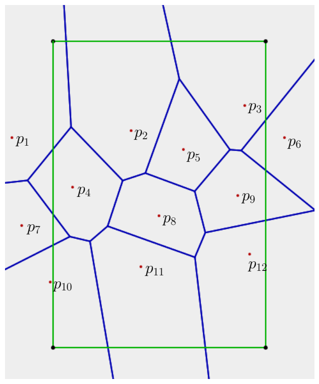
$$\bigcup_{C_j^k \ inside\ \mathcal{R}} H_j^k \subset I \cup \left(\bigcup_{C_i^k \ intersectwith\ \mathcal{R}} H_i^k \right)$$

where C_i^k and H_i^k denote the order-k Voronoi cell i in $V^k(S)$, and the set of associated sites, respectively.

Proof: We prove by induction on k. Clearly, the claim is true when $k = 1$. Assume it holds when $k = l$. We prove that it holds when $k = l + 1$. From [14], $V^{l+1}(S)$ can be constructed from $V^l(S)$ by tessellating order-l cells C_i^l's using the sites associated with C_i^l's neighbors. Thus, the Voronoi cell C_i^{l+1} inside $\mathcal{R}$ must be created by tessellating cells C_j^l's inside $\mathcal{R}$ or intersecting with $\mathcal{R}$ with sites corresponding to C_j^l's neighbors, which are either inside $\mathcal{R}$ or intersect with $\mathcal{R}$. Therefore, $\left(\bigcup_{C_j^{l+1} \ inside\ \mathcal{R}} H_j^{l+1}\right) \subset \left(\bigcup_{C_l^m \ inside\ \mathcal{R}} H_l^l \cup \bigcup_{C_i^l \ intersect\ \mathcal{R}} H_i^l\right) \subset \left(I \cup \left(\bigcup_{C_i^m \ intersect\ \mathcal{R}} H_i^l\right)\right)$. The cells of order-$(m+1)$ that intersect with $\mathcal{R}$ are created by tessellating order-m cells that intersect with $\mathcal{R}$ or are inside $\mathcal{R}$. Thus, $\left(\bigcup_{C_i^l \ intersect\ \mathcal{R}} H_i^l\right) \subset \left(\bigcup_{C_i^{l+1} \ intersect\ \mathcal{R}} H_i^{l+1}\right)$. This implies that $\left(\bigcup_{C_j^{l+1} \ inside\ \mathcal{R}} H_j^{l+1}\right) \subset \left(I \cup \left(\bigcup_{C_i^{l+1} \ intersect\ \mathcal{R}} H_i^{l+1}\right)\right)$. ∎



(a) Order-2 Voronoi diagram of 12 POIs. Cells intersecting with the cloak is shown darker.



(b) The corresponding order-1 Voronoi diagram of the 12 POIs and the cloak.

Fig. 6: Illustration of Lemma 2: finding 2-nearest POIs. The cloak is presented by the green rectangle $v_1v_2v_3v_4$. $\{\cdot, \cdot\}$ presents the POIs associated with a cell. $C_1, \ldots, C_4$ show four of the order-2 cells intersecting with the cloak.

Therefore, in determining the k-nearest POIs of cloak $\mathcal{R}$, we devise procedures to find POIs associated with order-k cells intersecting with $\mathcal{R}$ and order-1 cells inside the cloak, respectively.

Evaluating candidate k-nearest POIs from cells intersecting with the cloak: First, we identify the cells intersecting with the cloak $\mathcal{R}$ by tracing $\mathcal{R}$'s boundary. W.l.o.g., we assume that vertices $v_1, v_2, \ldots, v_m$ of the cloak, and Voronoi vertices of each cell are in a counter-clockwise order. We start from a vertex v_1 of $\mathcal{R}$ and find the Voronoi cell containing v_1. Let it be C_1. Note that, since Voronoi cells are convex, they intersect with a line segment at most twice. A Voronoi cell that has less than two intersections with a line segment must contain at least one of its endpoints. As a result, we can test if v_2 is in C_1. If yes, we conclude that the line segment $\overline{v_1v_2}$ is in C_1 and continue to the next line segment $\overline{v_2v_3}$. Otherwise, $\overline{v_1v_2}$ must intersect with one unique edge of C_1, which is a bisector of C_1 and one of its neighboring Voronoi cell, say C_2. C_2 and C_1 only differ in one site as given in the following lemma [14]:

Lemma 3: Let H_1 and H_2 be the sets of sites corresponding to two adjacent cells C_1 and C_2 in an order-k Voronoi diagram. Then, $H_1 = H \bigcup \{s_1\}$ and $H_2 = H \bigcup \{s_2\}$. Furthermore, s_1 and s_2 construct the bisector that contains the edge of C_1 and C_2.

In other words, as we move among neighboring cells intersecting with the cloak, only one POI is added at a time. The above procedure is repeated until the first vertex v_1 is encountered again. Let the resulting set of POIs be U. Consider the example in Figure 6a. Assume we start from cell C_1, POIs p_1 and p_2 are included in U. As we traverse from v_1 to v_2, we move to C_2, which differs from C_1 by p_5. Similarly, C_3 differs from C_2 by p_3, and so on.

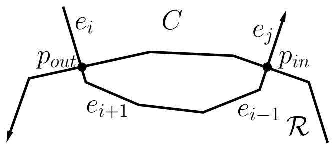


Fig. 7: Evaluation of the position of edges of cells in B regarding $\mathcal{R}$. C denotes a cell in B.

Evaluating candidate k-nearest POIs from cells inside the cloak: The next step is to retrieve the order-1 cells inside $\mathcal{R}$ and compute the corresponding candidate POIs.

We discuss the generalized problem of identifying the order-k cells I that are inside the cloak. The idea is to find the cells B intersecting with $\mathcal{R}$, which act as a boundary for the cells inside $\mathcal{R}$. We divide I into those that are adjacent to cells in B (I_1), and the rest, which are not (I_2). The cells in I_1 must share edges with cells in B, and these edges must be inside $\mathcal{R}$. Identifying edges of cells in B that are inside $\mathcal{R}$ can be done as follows. Consider a cell C that intersects with $\mathcal{R}$. As we move along $\mathcal{R}$'s boundary in a counter-clockwise order, we enter C at point p_{in} and exit at point p_{out}. Although C may intersect with $\mathcal{R}$ at more than 2 points, we can always identify such pairs (albeit multiple of them). Now, we observe that, as one moves along C's boundary from p_{out} to p_{in} in the counter-clockwise direction, the edges of C encountered that do not include p_{out} or p_{in} must be inside $\mathcal{R}$. This allows us to identify the cells of I_1. An illustrative example is given in Figure 7. Starting from point p_{out} in edge e_i where the cloak $\mathcal{R}$ exits from C, we trace along C in the counter-clockwise direction (arrow) until we reach point p_{in} on edge e_j where $\mathcal{R}$ enters C. Edges $e_{i+1}, \ldots, e_{j-1}$ are inside $\mathcal{R}$.

To find I_2, we simply iterate through the neighbors of cells that are inside $\mathcal{R}$ until no new inner cells are encountered. Consider the example shown in Figure 8. The cloak $\mathcal{R}$ is given by the green polygon. We start at vertex v_1 of $\mathcal{R}$, which lies in cell C_1. Applying the procedure described in the previous section, we can retrieve the shaded cells $C_1, \ldots, C_4$ that intersect with the boundary of $\mathcal{R}$. Next, cells inside $\mathcal{R}$ that are adjacent to cells of B, I_1, are identified, shown in white cells. As shown in the figure, those cells are adjacent to cells in B at the edges that lie completely inside $\mathcal{R}$. Finally, remaining cells inside $\mathcal{R}$ are identified by retrieving neighbors of cells in I_1, shown in red.

We summarize the procedure in Algorithm 2.

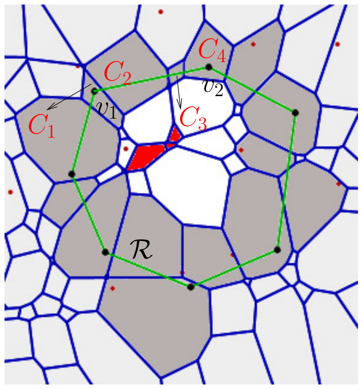


Fig. 8: Evaluation of the position of edges of cells in B regarding $\mathcal{R}$. C denotes a cell in B.

Algorithm 2: Computing the cloak's candidate K nearest POIs.

```
Data: A Voronoi diagram $V$, convex polygon cloak $\mathcal{R}$
Result: The set $U$ of $K$-nearest POIs associated with cells
        overlapping $\mathcal{R}$
/* POIs associated with order-$k$ cells
   intersecting with $\mathcal{R}$                              */
1  $B \leftarrow$ order-$K$ cells intersecting with $\mathcal{R}$;
2  $U \leftarrow$ POIs associated with $B$;
/* POIs associated with order-1 cells
   inside $\mathcal{R}$                                         */
3  $B' \leftarrow$ order-1 cells intersecting with $\mathcal{R}$;
4  $curr\_cells \leftarrow$ order-1 cells next to $B'$ that are inside $\mathcal{R}$;
5  $U' \leftarrow$ POIs associated with $curr\_cells$ ;
6  $B' \leftarrow B' \cup curr\_cells$;
7  while $curr\_cells \neq \emptyset$ do
8      $curr\_cells \leftarrow curr\_cells$'s neighbors inside $\mathcal{R}$ that are
       not in $B'$ ;
9      $U' \leftarrow$ POIs associated with $curr\_cells$ ;
10     $B' \leftarrow B' \cup curr\_cells$;
11 end
12 $U \leftarrow U \cup U'$;
```

Complexity analysis: We now analyze the running time of Algorithm 2. Let m be the number of $\mathcal{R}$'s vertices. Lines 1 and 3 compute the cells of order-k (B) and order-1 (B') intersecting with $\mathcal{R}$. It first locates the cell containing a cloak's vertex, which costs $O(\log n)$. Then, it iterates through all line segments of the cloak $\mathcal{R}$ and all edges of the order-K and order-1 cells intersecting with $\mathcal{R}$, thus costs $O(e_K + e_1 + m)$, where e_k is the number of edges of order-k cells intersecting with $\mathcal{R}$. Line 4 computes order-1 cells I_2 that are interior to $\mathcal{R}$ and adjacent to B', which iterates all edges of cells in B'. Line 8 computes the other order-1 cells that are inside $\mathcal{R}$, which costs the number of their edges. Let $\mathbf{C}$ be the set of order-k cells intersecting with $\mathcal{R}$ and order-1 cells overlapping $\mathcal{R}$ Computing U (lines 2, 5, 9) takes $O(s + c)$, where s is the number of POIs returned by the algorithm, and $c = |\mathbf{C}|$. (Note that $(e_K + e_1)$ is smaller than the number of $\mathbf{C}$'s edges). Therefore, Algorithm 2 costs $O(\log n + s + e + m)$ running time, where e is the number of $\mathbf{C}$'s edges. Since the number of edges in an order-K Voronoi diagram is $O(Kn)$ ([14]), e is bounded by $O(Kn)$. In practice, $e << O(Kn)$ due to the fact that the cloak is small and usually convex.

V. Evaluation

Results of hashing-based cloaking and POI search algorithms have been implemented in CGAL [1], a computational geometry library. All simulations run on a Core2 Duo 1.7Ghz Linux workstation.

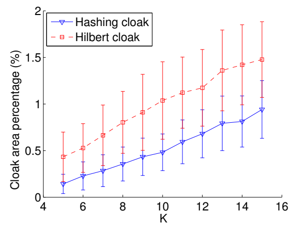
A. Cloaking

In the first set of experiments, we compare the performance of the proposed cloaking method with the Hilbert cloaking method. In the simulations, n user locations are randomly placed in a 1000×1000 area.

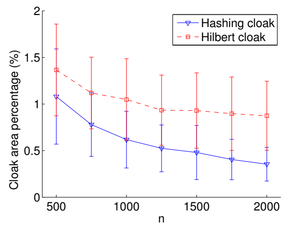
Figure 9 compares the performance of the two methods with respective to the level of K-anonymity where the number of users where n fixed at 1000 and K fixed at 10, respectively. The y-axis gives the size of the cloak as the percentage of the size of the area. Also shown in the figures are the error bars corresponding to the maximum and minimum cloak sizes. As can be observed in Figures 9a and 9b, the LSH-based approach is significantly better than the Hilbert curve-based method. As K increases, the cloak size increases roughly linearly in both methods. With more users, it is expected that the cloak size decreases linearly. This trend is more prominent with the proposed LSH-based method.

The hashing-based method is efficient computationally. Figure 10 plots the running time of the hashing-based cloaking method when the number of users varies from 1000 to 10000. As shown in the figure, the hashing-based method's running time is low even with a large number of users. Somehow, counter-intuitively, the running time of the algorithm decreases as the anonymity level K increases since it is inversely proportional to the anonymity level as indicate in Section III.

To evaluate the impact of the number of hash functions used, we randomly generate 1000 locations, and issue 10-nearest POI and 20-nearest POI queries. The number of hash functions, L, varies from 2 to 45. As shown in Figure 11, the higher the number of hash functions, the smaller the cloak areas produced. However, the cloak areas slightly decrease as the number of hash functions increases from 10 to 45. Similar to the results in Figure 9a, higher K implies large cloaks.



(a) The number of hash functions L is 20. n is fixed at 1000. K varies from 5 to 15.



(b) The number of hash functions L is 10. K is fixed at 10. n varies from 500 to 2000.

Fig. 9: Performance of Hilbert curve-based method and hashing-based method on various levels of K-anonymity and number of users (n).

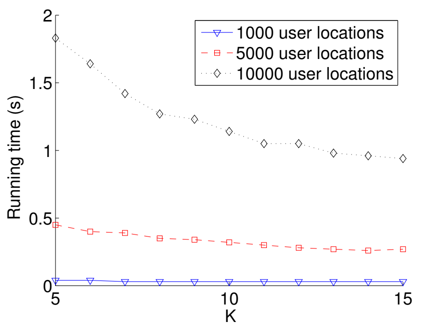


Fig. 10: Running time of hashing-based cloaking method. The number of hash functions L is 20.

We compare the resultant candidate POI set of the proposed search algorithm corresponding to hashing and Hilbert cloaks. We vary anonymity level K from 5 to 17 and make queries with 500 and 1000 POIs. As shown in Figure 12, the number of POIs corresponding to LSB-based cloaks is smaller than that of Hilbert cloaks. Moreover, the difference between the two methods increases as the number of POIs increases from 500 to 1000.

B. kNN Search

Next, we evaluate the performance of the proposed kNN search algorithm. In the simulations, 1000 POIs are randomly placed in a 1000×1000 area. We compare the running time with different K-anonymity (also K nearest POIs) and cloak areas (as the percentage of the total area). The results are shown in Figure 13. As seen in the figure, the running time is negligible. The running time increases only moderately as K or the size of the cloak area increases, which is sharp contrast with the results in [11].

Figure 14 shows the number of candidate k-nearest POIs. We evaluate the number of candidate POIs with different cloak areas and values of k. As shown in the figure, the number of candidate k-nearest POIs increases approximately linearly as the cloak area increases since POIs are evenly distributed.

Lastly, we compare the complexity of the proposed algorithm with the naive search algorithm that scans order-k Voronoi cells. We vary the cloak area and plot the number of cells processed. As shown in Figure 15, the number of cells

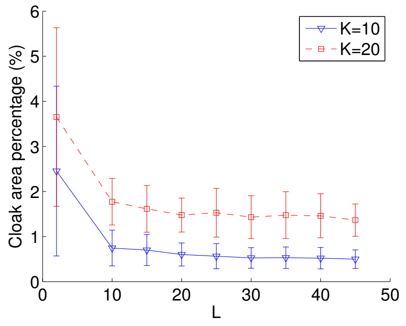


Fig. 11: Hashing-based cloaking performance with different number of hash functions.

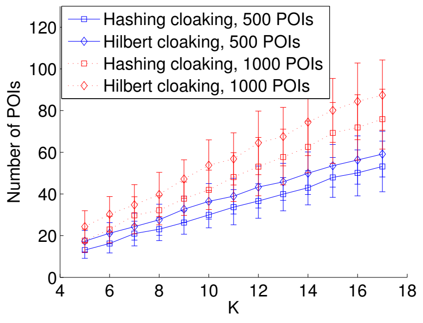


Fig. 12: Number of candidate POIs corresponding to Hilbert and hashing cloaks.

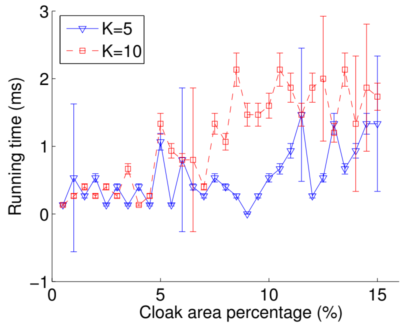


Fig. 13: Running time of the proposed search algorithm.

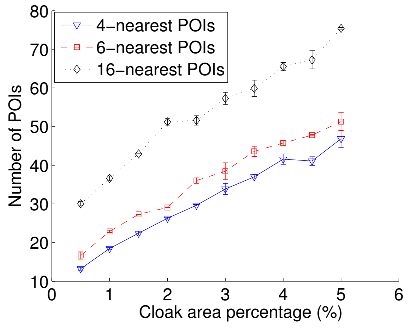


Fig. 14: Number of candidate k-nearest POIs.

processed in the proposed method is much lower. Furthermore, as the cloak area increases, the number of cells processed by our method increases much slower than that in the naive search. Figure 15 corroborates the low running time of the proposed algorithm in Figure 13.

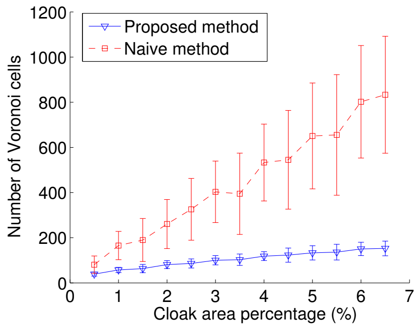


Fig. 15: Number of cells processed by the proposed method VS. the naive (only considers order-k cells).

VI. Related work

Location privacy in participatory sensing campaign is intensively studied in literature ([4], [17], [18], [9], [19]). In the scenario proposed in [12], users have access to a list of data collection sites to and choose the site closest to them. In the work, Kazemi et al. proposed PiRi, a privacy framework that utilizes representative participants for range queries independent of query issuers' location. PiRi assumes that participants can trust one another, and thus may subject to insider attacks.

Privacy in location-based services has drawn much attention in the database and data mining community in recent years. An excellent survey can be found in [7]. Existing approaches broadly fall into two categories: user-side approaches and approaches that require a trusted server. In the first category, users anonymize their location-based queries by adding noise to the location attributes or generating multiple decoys at different locations. One such approach is called SpaceTwist [23]. In SpaceTwist, starting with a location different from the user's actual location, the nearest neighbors are retrieved incrementally until the query is answered correctly. The uncertainty of the user location is roughly the distance from the initial location to the user's actual location. SpaceTwist requires implementation of incremental k-NN query on the server sides. Furthermore, it does not guarantee K-anonymity if the resulting uncertain area contains less than $K-1$ other users.

With a trusted anonymizer, more sophisticated spatial cloaking mechanisms can be devised. In Casper [15], the anonymizer maintains the locations of the clients using a pyramid data structure, similar to a Quad-tree. Upon reception of a query, the anonymizer first hashes the user's location to the leaf node and then move up the tree if necessary until enough neighbors are included. Hilbert cloaking [11] uses the Hilbert space filling to map 2-D space into 1-D values. These values are then indexed by an annotated B+-tree, which supports efficient search by value or by rank (i.e., position in the 1-D sorted list). The algorithm partitions the 1-D sorted list into groups of K users. Hilbert cloaks though achieving K-anonymity does not always preserve locality, which leads to large cloak size and high server-side complexity. Recognizing that Casper does not provide K-anonymity, Ghinita et al. proposed a framework for implementing reciprocal algorithms using any existing spatial index on the user locations [8]. Once the anonymous set (AS) is determined, the cloak region can be represented by rectangles, disks or simply the AS itself.

Specialized (LBS-side) algorithms have been proposed for identifying a candidate set that includes the k nearest neighbors for any location in a convex m-vertex polygon [10]. The authors proposed a sweep-line-based algorithm with $O(mk^2 n \log n)$ worst-case time complexity, which incurs a higher complexity than the proposed Voronoi diagram based approach. In [11], the authors proposed an algorithm for circular cloaked region with worst-case exponential time complexity. Different from aforementioned work, we propose a search algorithm with any simple polygonal shape cloak with improved running time complexity. The algorithm can be easily extended to handle circular cloaks. Finally, cryptographic approaches have been applied to location privacy, where one-way hash functions are used to encode user and POI locations [16]. Both exact and approximate nearest-neighbor search can be supported at the expense of higher computation complexity.

VII. Conclusion and Future work

In this paper, we proposed computationally efficient algorithms for constructing spatial cloaks and kNN search of POIs with the dual objectives of locality preservation and K-anonymity. We showed through extensive simulations that the algorithms achieve superior performance with moderate time complexity are scale well with large input size. The proposed algorithms are central to protect identity privacy in participatory sensing applications.

As part of the future work, we plan to devise efficient cloaking algorithms that tighten the conditions for K-anonymity. In particular, as discussed in Section II, reciprocity is a sufficient but not necessary condition for K-anonymity. In fact, users may be assigned to multiple cloaks as long as the probability of each user in the same cloak is the same. Also of interest are incremental algorithms that handle gracefully location updates.

VIII. Acknowledgement

This work is funded by the National Science Foundation (NSF) under award CNS-0832089.

References

[1] CGAL - computational geometry algorithms library.

[2] Dakshi Agrawal and Charu C. Aggarwal. On the design and quantification of privacy preserving data mining algorithms. In *PODS '01: Proceedings of the twentieth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 247–255, New York, NY, USA, 2001. ACM.

[3] Hossein Ahmadi, Nam Pham, Raghu Ganti, Tarek Abdelzaher, Suman Nath, and Jiawei Han. Privacy-aware regression modeling of participatory sensing data. In *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*, SenSys '10, pages 99–112, New York, NY, USA, November 2010. ACM.

[4] J. Burke, D. Estrin, M. Hansen, A. Parker, N. Ramanathan, S. Reddy, and M. B. Srivastava. Participatory sensing. In *In: Workshop on World-Sensor-Web (WSW06): Mobile Device Centric Sensor Networks and Applications*, pages 117–134, 2006.

[5] Mayur Datar and Piotr Indyk. Locality-sensitive hashing scheme based on p-stable distributions. In *In SCG 04: Proceedings of the twentieth annual symposium on Computational geometry*, pages 253–262. ACM Press, 2004.

[6] Herbert Edelsbrunner, Lionidas J Guibas, and Jorge Stolfi. Optimal point location in a monotone subdivision. *SIAM J. Comput.*, 15:317–340, May 1986.

[7] Gabriel Ghinita. Private queries and trajectory anonymization: a dual perspective on location privacy. *Trans. Data Privacy*, 2:3–19, April 2009.

[8] Gabriel Ghinita, Keliang Zhao, Dimitris Papadias, and Panos Kalnis. A reciprocal framework for spatial k-anonymity. *Inf. Syst.*, 35:299–314, May 2010.

[9] Peter Gilbert, Landon P. Cox, Jaeyeon Jung, and David Wetherall. Toward trustworthy mobile sensing. In *Proceedings of the Eleventh Workshop on Mobile Computing Systems and Applications*, HotMobile '10, pages 31–36, New York, NY, USA, 2010. ACM.

[10] Haibo Hu and Dik Lun Lee. Range nearest-neighbor query. *IEEE Trans. on Knowl. and Data Eng.*, 18:78–91, January 2006.

[11] Panos Kalnis, Gabriel Ghinita, Kyriakos Mouratidis, and Dimitris Papadias. Preventing location-based identity inference in anonymous spatial queries. *IEEE Trans. on Knowl. and Data Eng.*, 19:1719–1733, December 2007.

[12] Leyla Kazemi and Cyrus Shahabi. Towards preserving privacy in participatory sensing. In *2011 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, pages 328–331. IEEE, March 2011.

[13] David G. Kirkpatrick. Optimal search in planar subdivisions. Technical report, Vancouver, BC, Canada, Canada, 1981.

[14] Der-Tsai Lee. On k-nearest neighbor voronoi diagrams in the plane. *IEEE Trans. Comput.*, 31(6):478–487, 1982.

[15] WG Aref MF Mokbel, CY Chow. The new casper: A privacy-aware location-based database server. *2007 IEEE 23rd International Conference on Data Engineering*, pages 1499–1500, 2007.

[16] Arvind Narayanan, Narendran Thiagarajan, Michael Hamburg, Mugdha Lakhani, and Dan Boneh. Location privacy via private proximity testing. *NDSS'10*, 2011.

[17] Sasank Reddy, Jeff Burke, Deborah Estrin, Mark Hansen, and Mani Srivastava. A framework for data quality and feedback in participatory sensing. In *Proceedings of the 5th international conference on Embedded networked sensor systems*, SenSys '07, pages 417–418, New York, NY, USA, 2007. ACM.

[18] Sasank Reddy, Deborah Estrin, and Mani Srivastava. Recruitment framework for participatory sensing data collections. In Patrik Floréen, Antonio Krüger, and Mirjana Spasojevic, editors, *Proceedings of the 8th International Conference on Pervasive Computing*, pages 138–155, Berlin, Heidelberg, May 2010. Springer Berlin Heidelberg.

[19] Jing Shi, Rui Zhang, Yunzhong Liu, and Yanchao Zhang. Prisense: Privacy-preserving data aggregation in people-centric urban sensing systems. In *INFOCOM*, pages 758–766, 2010.

[20] Latanya Sweeney. k-anonymity: a model for protecting privacy. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, 10:557–570, October 2002.

[21] Manolis Terrovitis and Nikos Mamoulis. Privacy preservation in the publication of trajectories. In *Proceedings of the The Ninth International Conference on Mobile Data Management*, pages 65–72, Washington, DC, USA, 2008. IEEE Computer Society.

[22] Godfried T. Toussaint. Solving geometric problems with the rotating calipers. In *Proceedings of IEEE MELECON*, 1983.

[23] Man Lung Yiu, Christian S. Jensen, Xuegang Huang, and Hua Lu. Spacetwist: Managing the trade-offs among location privacy, query performance, and query accuracy in mobile services. In *In ICDE*, 2008.

[24] V. M. Zolotarev. One-dimensional stable distributions. In *Translations of Mathematical Monographs*, volume 65. American Mathematical Society, 1986.