Local Differential Perturbations: Location Privacy Under Approximate Knowledge Attackers

Rinku Dewri, *Member, IEEE*

Abstract—Location privacy research has received wide attention in the past few years owing to the growing popularity of location-based applications, and the skepticism thereof on the collection of location information. A large section of this research is directed towards mechanisms based on location obfuscation enforced using cloaking regions. The primary motivation for this engagement comes from the relatively well researched area of database privacy. Researchers in this sibling domain have indicated multiple times that any notion of privacy is incomplete without explicit statements on the capabilities of an adversary. As a result, we have started to see some efforts to categorize the various forms of background knowledge that an adversary may possess in the context of location privacy. Along this line, we consider some preliminary forms of attacker knowledge, and explore what implication does a certain form of knowledge has on location privacy. Continuing on, we extend our insights to a form of adversarial knowledge related to the geographic uncertainty that the adversary has in correctly locating a user. We empirically demonstrate that the use of cloaking regions can adversely impact the preservation of privacy in the presence of such approximate location knowledge, and demonstrate how perturbation based mechanisms can instead provide a well-balanced trade-off between privacy and service accuracy.

Index Terms—location privacy, differential privacy, query approximations

1 Introduction

Location based services (LBS) form a large class of applications in modern day mobile systems. These applications utilize the positioning capabilities of a mobile device to determine the current location of the user, and customize query results to include neighboring points-of-interest (POI). Wide acceptance of personal digital assistants and the advancements in wireless cellular technology have opened up countless possibilities in this business paradigm. Potential applications can range from proximity based notifications to tracking business resources. A wireless carrier typically serves as a channel between the user and the location content provider.

The potential advantages of location based applications is not difficult to realize. However, location knowledge is often perceived as personal information. It remains an open question whether the benefits of these applications can outweigh the underlying privacy risks. Drawing inspiration from efforts in database privacy, location based applications have been argued to be usable without communicating precise location data to the content provider.

Location obfuscation is a widely researched technique to achieve location privacy. The fundamental idea here is to process location based queries relative to a sufficiently larger region, also known as a cloaking region, compared to one where a user can be uniquely located. For instance, a cloaking region can be generated to include k users, including the one making the query [1]. Multiple algorithms have been proposed to generate such a k-anonymous cloaking region [2], [3]. However, obfuscating private data without understanding the capabilities of the adversary can be unproductive. The background knowledge of the adversary must be known (or at least assumed) in order to demonstrate the privacy guarantees of a mechanism.

We begin this work by identifying the primary form of attacker knowledge targeted by most location obfuscation techniques. This knowledge relates to an adversary being able to determine the true locations of a certain subset of users. Using a case by case analysis of what this adversary can achieve from queries made using true locations and queries made using cloaking regions, we argue that "location privacy" is a misused term in this context. The use of cloaking regions is motivated by the need to introduce ambiguity in correlating a user to a query. If an adversary does not have any form of location knowledge on the users, then location information in a query cannot be used to map it to a user. The adversary must posses at least approximate location knowledge about the user, to be able to exploit the location information in a query. On the other hand, if true location knowledge is present, then there is no location privacy. In fact, what is being offered is query privacy. We treat the two forms of privacy differently – location privacy meaning hiding the location and query privacy meaning preventing the mapping of a query to a user.

In one of our earlier works, we highlighted the inadequacy of cloaking regions in preventing location privacy breaches when approximate location knowledge is available to an adversary [4]. No privacy mechanism should enable an adversary to infer exact knowledge using the existing background information. Towards this end, we explored the possibility of using perturbed

R. Dewri is with the Department of Computer Science, University of Denver, Denver, CO 80208, USA. Email: rdewri@cs.du.edu

locations to issue queries, and proposed a perturbation method based on local enforcement of differential privacy. Differential privacy works under the principle that the chances of being a victim of a privacy breach should not increase substantially due to the inclusion of ones private information in a shared data set [5]. However, the privacy level was assumed to be a function of the anonymity set size during the local differential privacy enforcement. The work here provides a robust evaluation of the privacy levels enforced by such differentially-private perturbations, in terms of a pair of correctness and uncertainty measures. In terms of privacy assessment, correctness measures the probability with which an adversary can correctly guess the location of a user, and uncertainty measures how skewed (or uniform) is the adversary's guess across other likely locations. We provide an algorithm to efficiently check the amount of uncertainty that still remains in correctly associating the true location to a user. Last, we empirically validate that, on both synthetic and real-world points-of-interest, perturbed locations can in fact be used to retrieve a significantly large subset of the actual query results. This strengthens the feasibility of using perturbed locations in a real-world setting.

The remainder of the paper is organized as follows. We begin with a restatement of our earlier work in Sections 2 and 3. Section 2 initiates our discussion on attacker capabilities, and the affect on location and query privacy. Section 3 presents our approach to address a form of attacker knowledge based on approximate locations of the users. Section 4 provides an evaluation of the proposed approach in terms of the correctness and uncertainty in determining the true location of a user. Section 5 presents some empirical results on the effectiveness of the approach in generating useful query results, as well as a comparison with the breach possibilities in a cloaking based approach. Section 6 lists some related work in this area, followed by references to future work in Section 7.

2 Attacker Class

Classification of attacker knowledge is crucial in order to provide a comprehensive statement on the privacy preserving properties of an obfuscation technique. To consider the extremes, location obfuscation in the presence of an "oracle" or an adversary with effectively no background knowledge, is only going to degrade the quality of service. Other intermediate scenarios also exist where location obfuscation cannot achieve one or both of location and query privacy.

Background knowledge in a LBS setting has earlier been classified into classes involving the user set, the connection between users and places, and prior events [6]. For instance, an adversary may posses knowledge of the entire, or a subset, of users that form the active user-set. Further, the pseudonyms used by the users may also be known. In terms of location knowledge, the adversary may be able to correlate some users to certain places (say home or work locations) at given time instances. Prior knowledge may also help the adversary in eliminating the possibility of a certain user being at a certain place. Mobility profiles of the user may also have been gathered by observing prior events. These knowledge forms are the basis for attacks that seek to correlate events at multiple points in time to make location inferences. We direct our focus to the knowledge form pertaining to the user-to-location mapping. Note that this mapping is often fuzzy. Hence, the error in tracking a user is not always the distance between an estimated and the actual location, but can also be the area of a geographic region surrounding the actual location.

An adversary that has information on the locations of any individual(s) is referred to here as a *locator*. Further, a *perfect* locator knows exact coordinates of the users, while an *approximate* locator has approximate knowledge (an area instead of exact coordinates) on the locations. We also consider another form of knowledge that is related to the identity of users issuing the queries. We refer to any adversary that has access to the query database as a *holder* (also refereed to as a *simple* holder). A *perfect* holder in this case would be an adversary who knows the identity of the person who issued a query. There are multiple permutations in which these two forms of knowledge may be present in an adversary. While each form in itself states how much an adversary knows about the locations or queries of the users, respectively, the objective is to avoid the inference or significant improvement of one form of knowledge using existing knowledge of the other form.

Location based service users communicate location information as part of their queries. The location information can be in the form of precise GPS coordinates, the resulting query being processed thereafter with respect to a point in space. Such queries are also referred to as *point queries*. However, due to the implications on privacy, precise locations are obfuscated using a cloaking region. Queries in this case are processed on a geographic range, therefore referred to as *range queries*. We begin with point queries and put the two forms of attacker knowledge in perspective with respect to such queries. Some of the following observations related to privacy violations in point queries are well-known in the community. We present them here for the sake of completeness.

2.1 Point Queries

A point query is where exact geographic coordinates are communicated along with the query. A query database in this case contains the precise location of users, among other parameters of the queries. It is a straightforward observation that no location privacy can be achieved in the presence of a perfect locator, and no query privacy can be achieved in the presence of a perfect holder. Nonetheless, query privacy is preserved in the case of a

perfect locator. However, as an immediate consequence of point queries, location privacy is violated even when the adversary is only a perfect holder. A perfect holder in this case performs an identity to location mapping using the location information in the query database. A perfect locator must also be at least a holder to effectuate a breach of query privacy. In this case, the adversary uses the location knowledge to determine the corresponding query of the user in the database. The perfect locator here covers situations such as restricted space identification and observation based identification [1]. A simple holder with access to the query database alone is no threat to either the location or query privacy of the users.

The effectiveness of point queries in the presence of approximate locators has not been evaluated yet. Point queries can be potentially harmless depending on the extent of the adversary's approximation. For instance, an approximate locator with an approximation of a few hundred meters is stronger than one with an approximation of a city block. The exact extent of knowledge is difficult to estimate. We shall discuss later how point queries can still be effectively generated in the presence of approximate locators.

2.2 Range Queries

A range query is where a query region is associated with the query. Query results are generated assuming that the user may be located anywhere inside the region. The query region serves as a cloak for the user, and is generated following some established privacy principle. For instance, a k-anonymous cloaking region would encompass at least k users inside it. An obfuscation algorithm tries to achieve the privacy principle within the smallest possible area. In the following, we present a case by case overview of which privacy aspect does a range query help preserve, and under what form of adversarial knowledge.

2.2.1 Locator

Since a perfect locator knows the location of a user, use of a cloaking region does not help hide the location of the user. Query privacy is preserved in the absence of access to the query database. This implies that no privacy breach (in the sense of gaining additional knowledge) can occur in the presence of a perfect locator. Point queries can in fact be used instead of a range query, in order to improve the quality of service. Cloaking regions also do not help achieve better location privacy from an approximate locator. The approximation of the adversary on the user's location is what determines the location privacy level. Point queries can again be used here, given that the adversary has no access to the query database. In other words, no location privacy violation can occur as a side-effect of the user using the service.

2.2.2 (Perfect) Holder

No query privacy is possible in the presence of a perfect holder. Location privacy violation is certain since the cloaking regions present in the queries provide approximate location knowledge to the adversary. The cloaking regions can potentially reveal more precise information as well. Note that a privacy principle such as k-anonymity is meant to prevent the association of a user to the issued query – any of the k users could have issued the query. However, such a principle is irrelevant in the case of a perfect holder. A better principle to enforce would be location diversity [7], [8]. This would guarantee that zones with multiple levels of sensitivity are present within the cloaking region, thereby preventing further location based inferences. Request locality is another issue to address. This situation occurs when different likelihoods can be estimated for issuing the query from different areas within the cloaking region. Both location and query privacy are preserved in the case of a simple holder.

2.2.3 Perfect Locator and Holder

The location of a user is already known to this kind of an adversary. It is easy to determine the set of queries that could have potentially originated from a certain user. However, query privacy violation can be prevented if the cloaking region can generate an ambiguous mapping between a query and the user. This is achieved by anonymity principles such as k-anonymity. In fact, obfuscation methods that generate minimal k-anonymous cloaking regions assume the existence of a perfect locator with precise location knowledge of at least k users. This assumption implies that location obfuscation is used here to preserve query privacy, and not necessarily any form of location privacy. Query privacy, however, can also be preserved by issuing a point query using the true location of one of the k users. This can produce a relatively accurate result set if the bounding rectangle of the k users is not excessively large. The result sets would differ much for larger bounding rectangles, in which case the costs may itself be too high for acceptable range query processing.

2.2.4 Approximate Locator and (Perfect) Holder

While cloaking regions are sufficient (although perhaps not always required) to handle a perfect locator and holder, their use starts to have a detrimental effect in the presence of approximate locators. The inadequacy of cloaking regions in providing location privacy has earlier been highlighted by Shokri et al. [9]. The authors demonstrate that generating minimum-area cloaking regions can reveal the active user-set, and thereby violate location privacy. In our context, when location knowledge is approximate, a cloaking region can enhance the precision. As depicted in Fig. 1, a k-anonymous cloaking region may allow an approximate locator to improve upon the location knowledge of more than just the query issuer. The problem is eliminated only if the cloaking region is guaranteed to encompass the approximated regions corresponding to each of the k users. Unfortunately, it is difficult to judge the extent of knowledge

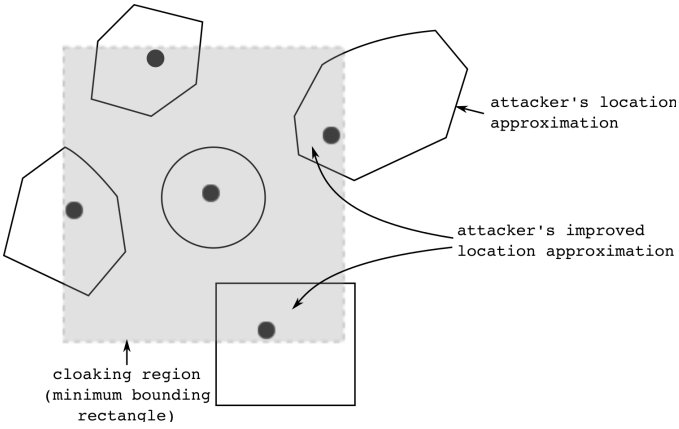

Fig. 1. Location privacy breach as a result of using cloaking regions.

that an adversary possesses. This case presents us with a situation where the obfuscation method helps preserve query privacy (if adversary is not a perfect holder), but can potentially lead to a breach in location privacy.

We summarize below the conclusions that can be drawn from the discussion in the preceding sections.

1) Neither location privacy nor query privacy can be preserved in the presence of a perfect locator and a perfect holder.
2) Point queries pose privacy threats in the presence of a perfect locator and holder.
3) Cloaking regions help preserve query privacy in the presence of a perfect locator and holder.
4) Cloaking regions can provide query privacy in the presence of an approximate locator and holder, but do not guarantee protection against a location privacy breach.

3 Differential Perturbations

Current location obfuscation techniques based on cloaking regions are insufficient, and undesirable, in location privacy preservation. This arises from the fact that perfect locators represent a very strong class of adversaries. For instance, acquiring the exact geographic coordinates of a user would require resources of the likes of a RFID tracking tag and satellite based monitoring. Further, not much can be done with location obfuscation once an adversary gains access to such information. A more plausible form of adversary is represented by an approximate locator. Approximate location knowledge can be obtained by a variety of means – device communication logs such as cell towers used, public records such as parking violations, or social engineering methods such as during a casual conversation. Unless regulated by legislations, the approximate location can more simply be inferred directly from the information broadcast from cell towers and wireless access points. The "blue circle" we often see in existing mobile applications is a manifestation of this technique. Preserving location privacy in this context dwells upon the problem of preventing an adversary from reducing the margin of location error using external references of a user's activities (such as in a location based service log).

Although point queries are potentially risky, queries from perturbed user locations remain a possibility. Perturbations may be generated such that they do not bear any direct linkage to a single user. The downside is that queries based on perturbed locations can result in an inaccurate result set. However, if the perturbations are reasonably close to the actual location, then the query results can also be assumed to be close enough to the true set. There is definitely an inherent trade-off involved between the accuracy of the result set and the location perturbations. We postpone the analysis of this trade-off for a later stage and focus on the generation of the perturbations themselves.

Trivial methods such as using the centroid of a k-anonymous cloaking region as the perturbed location are susceptible to an inversion attack [4]. Our approach is motivated by the requirement to provide probabilistic bounds on what an adversary can learn from the perturbed location.

3.1 Location Perturbation

Each user query in a LBS application is tagged with a geographic location. Assuming that users are not willing to reveal their true coordinates, the objective is to compute a perturbed location for use in the query. The perturbation can be generated by adding a random noise (to the true location) drawn from a standard probability distribution. However, the credibility of this method is questionable if the adversary knows the distribution and a set of likely positions (including the true location) for the user. With such information, the adversary can compute the probability of generating the observed perturbation from each of the likely positions. The adversary will confidently infer the user position if the probability is significantly high for the true location. Therefore, our objective is to perform the perturbation in a manner such that these probabilities are within a small constant factor of each other. As shown below, the addition of noise from a carefully selected Laplace distribution can fulfill this requirement.

Let l_p be the perturbed location corresponding to a true location l_t, denoted as $l_t \rightarrow l_p$. A location is assumed to have two components, denoted by the x and y coordinates. Let $l_1, ..., l_k$ be a set of k points, one of which is l_t. The method of choosing these k points is discussed in the next section. We would generate the perturbed location $l_p = (x_p, y_p)$ such that, for any two locations l_i and l_j,

$$Pr(x_i \rightarrow x_p) \leq e^{\epsilon} Pr(x_j \rightarrow x_p) \text{ and}$$

$$Pr(y_i \rightarrow y_p) \leq e^{\epsilon} Pr(y_j \rightarrow y_p),$$

where $\epsilon \geq 0$ and $i, j \in \{1, ..., k\}$. We achieve this property by using a Laplace distribution with scale $\lambda > 0$ to perturb a location $l_i = (x_i, y_i)$ such that

$$Pr(x_i \rightarrow x_p) = \frac{1}{2\lambda} e^{-\frac{|x_i - x_p|}{\lambda}} \text{ and}$$

$$Pr(y_i \rightarrow y_p) = \frac{1}{2\lambda} e^{-\frac{|y_i - y_p|}{\lambda}}.$$

The amount of noise to be added to a component is given as $-\lambda \text{sign}(rnd) \ln(1 - 2|rnd|)$, where rnd is a uniform random value in $]-\frac{1}{2}, \frac{1}{2}]$. Based on the following observation, λ is set at $(\max_n x_n - \min_n x_n)/\epsilon$ to generate x_p, and set at $(\max_n y_n - \min_n y_n)/\epsilon$ to generate y_p. l_p is obtained as (x_p, y_p).

Observation: Without loss of generality, let c denote a generic component of a location. Using the triangle inequality, we can write $|c_j - c_p| \leq |c_j - c_i| + |c_i - c_p|$. After rearrangement, dividing by λ, raising as a power of e and multiplying by $1/2\lambda$, we get

$$\begin{aligned} \frac{1}{2\lambda} e^{-\frac{|c_i - c_p|}{\lambda}} &\leq \frac{1}{2\lambda} e^{-\frac{|c_j - c_p|}{\lambda}} e^{\frac{|c_j - c_i|}{\lambda}}, \text{ or} \\ Pr(c_i \rightarrow c_p) &\leq Pr(c_j \rightarrow c_p) e^{\frac{|c_j - c_i|}{\lambda}}. \end{aligned}$$

We therefore have

$$\begin{aligned} Pr(x_i \rightarrow x_p) &\leq Pr(x_j \rightarrow x_p) e^{\frac{|x_j - x_i|}{\lambda}} \text{ and} \\ Pr(y_i \rightarrow y_p) &\leq Pr(y_j \rightarrow y_p) e^{\frac{|y_j - y_i|}{\lambda}}, \end{aligned}$$

and the power of the exponent is bounded as

$$\begin{aligned} Pr(x_i \rightarrow x_p) &\leq Pr(x_j \rightarrow x_p) e^{\frac{\max_n x_n - \min_n x_n}{\lambda}} \text{ and} \\ Pr(y_i \rightarrow y_p) &\leq Pr(y_j \rightarrow y_p) e^{\frac{\max_n y_n - \min_n y_n}{\lambda}}. \end{aligned}$$

Hence, the probability of a location coordinate generating a certain perturbed value is always within a factor e^{ϵ} of the probability of some other location (in the set of k points) generating the same perturbed value.

3.2 Selecting a Perturbation

A perturbed location for a query point can be chosen using the above method. However, the distribution of the k points can affect the proximity of the perturbed location to the true coordinates. For example, if locations are perturbed based on the coordinates of every known user within a city, the scale parameter in the noise distribution will become significantly high, thereby resulting in heavy noise addition. One method to resolve the problem is to compute the perturbation from a restricted set of k points, k now being an argument of the perturbation mechanism. Further, the k points should be chosen to preserve reciprocity [3], [10]. In other words, the same set should be chosen irrespective of which of the k locations is the query point. This is achieved by dividing the users into buckets of size k, the set being chosen as the bucket to which the query point belongs. The buckets are formed based on the Hilbert indices of the users. The locality preserving properties of Hilbert curves ensure (although not necessarily optimal) the formation of buckets with users that are at close proximity to each other. The bounding box of the k points corresponds to the cloaking region used in the HilbertCloak algorithm [3], and is used during the privacy evaluation steps later in Section 5.

Each of the k points is then subjected to perturbation, and the one having the minimum average distance to all points in the set is chosen as the location to use in the query. Note that our probabilistic guarantee is only local to a bucket. Given a perturbed location, the k points are probabilistically identical (within a factor of e^{ϵ}) irrespective of which one was used to perform the perturbation. Hence, choosing the one with minimum average distance to all points does not risk an inversion attack. Note that the context of the application still plays a crucial role. If the user base is relatively sparse, i.e. the k users are distributed over a significantly large area, then the generated perturbation will still be far away from the true location. A cloaking region would also be unacceptably large in this case.

3.3 Evaluating the Perturbation

Cloaking regions guarantee that the results generated for a location based query will contain the results corresponding to the location of the user. Such a claim cannot be made for queries issued with a perturbed location. However, differences in the result may or may not exist depending on the density of the queried objects, and the distance of the perturbed location from the true one. Consider a *Knn*-query that retrieves the K nearest objects with respect to a given location. Such a query is likely to generate a larger subset of common results on sparsely distributed objects (e.g. police stations). On the other hand, for densely distributed objects (e.g. cafe), this likelihood reduces. Note that we use a lower case k for the computation of a perturbed location.

Result set similarity can also be measured with respect to the distances to the retrieved objects. Under this measure, two result sets are considered similar if, corresponding to every object in one set, there exists an object in the other set that is equidistant from the queried location. This perspective of result similarity applies well to proximity based queries – nearest gas stations, nearest restaurants, nearest friends – where the distance to the object carries more weight than attributes of the objects. Result set similarity using common subsets is relevant in queries where the retrieved objects must be ordered using user-stated preferences – nearest K cheapest gas stations.

A third measure is also possible using the distance of the perturbed location from the true location. Assuming that the service provider guarantees that the result set is accurate relative to the query point, a user wanting complete accuracy will have to travel from the current

location to the perturbed point. It is therefore worth investigating how far is the generated perturbation from the current location of the user.

Although we are not stating any theoretical bounds on these metrics at this stage, intuition says that query processing relative to well-formed perturbed locations will not be futile. As the first step, the following three metrics are used to evaluate the effectiveness of our approach [11].

1) *Nearness:* The nearness value signifies the fraction of perturbations at close proximity to the true location.
2) *Resemblance:* Let $\mathcal{O} = \{o_1, ..., o_K\}$ be the objects retrieved by a *Knn*-query relative to the true location of user $\mathcal{U}$, and $\mathcal{O}' = \{o'_1, ..., o'_K\}$ be the objects retrieved relative to the perturbed location. The resemblance is the fraction of common objects between $\mathcal{O}$ and $\mathcal{O}'$, given as
$$\frac{|\mathcal{O} \cap \mathcal{O}'|}{|\mathcal{O}|}.$$
3) *Displacement:* The displacement is measured as the average difference in object distance (from the true location) across the set of mismatched objects, given as
$$\begin{cases} \frac{\sum_{i=1}^{K} dist(o'_i, \mathcal{U}) - \sum_{i=1}^{K} dist(o_i, \mathcal{U})}{|\mathcal{O}| - |\mathcal{O} \cap \mathcal{O}'|}, & \mathcal{O} \neq \mathcal{O}' \\ 0, & \mathcal{O} = \mathcal{O}' \end{cases},$$
$dist(\cdot)$ being a distance value between an object's location and the true location of a user.

4 Privacy Evaluation

Privacy evaluation involves reverse computations performed by an adversary with background knowledge about the underlying protection mechanism and user locations. A separate line of research in analyzing anonymous location traces have revealed that user locations are heavily correlated, and knowing a few frequently visited locations can easily identify the user behind a certain trace [12], [13]. Note that the privacy breach in these cases occurs because the location to identity mapping results in a violation of user anonymity. The proposal in this work attempts to prevent the reverse mapping—from user identity to user location. A known user may be generating a trace (of perturbed locations) over a period of time, however by virtue of the local differential privacy enforcement, the true location can be any of the coordinates from a set of k points.

True locations of users are geographic points. However, we take a granular approach where approximation of the true location to a small geographic area is a privacy violation of the same magnitude as determining the true location itself. Henceforth, we consider that geographic areas are divided into *cells* of a relatively smaller area, say $100m \times 100m$. We change the notation (x_i, y_i) to denote the cell where user i is located. Similarly, (x_p, y_p) is the cell of the perturbed location. We then define the approximate location knowledge of an adversary, corresponding to a user, as a collection of cells where the user is believed to be present. We assume an adversary with rather accurate knowledge, such that the collection of cells is centered at the true cell of a user. The adversary's location knowledge with respect to user i is then specified as $\mathcal{LK}_i = \{(x, y) | x \in [x_i - r_i, .., x_i + r_i], y \in [y_i - r_i, .., y_i + r_i]\}$. The notation $[a, .., b]$ implies an integer interval. The set of x and y values is denoted by $\mathcal{LK}_i.x$ and $\mathcal{LK}_i.y$ respectively. r_i is also referred to as the *radius* of the location knowledge. A radius of one signifies that the user is equally likely to be in the true cell and the eight neighboring ones. Hence the probability of associating a user to the correct location is $1/|\mathcal{LK}_i|$. It can be argued that the uniform probability assumption is not always valid. However, this possibility is more likely when the set of cells considered in $\mathcal{LK}_i$ is large, say crossing city limits, thereby making some cells more probable than others. Hence, in our experiments, we have assumed a considerably strong adversary that can place a user within a kilometer of the true location. An adversary who can further specialize the probability distribution can probably track the user in real time. Further, we assume a uniform knowledge adversary, implying that the radius is same (r) for every user, i.e. $r_i = r, \forall i$, unless stated otherwise.

4.1 Correctness and Uncertainty

An approximate knowledge adversary begins with a set of equally likely positions (cells) for a user. However, certain positions can become more probable after the adversary observes the perturbed location used by the user. This probability shift occurs since certain positions in the adversary's approximate knowledge is highly unlikely to generate the observed perturbation under the used noise distribution. Thereafter, the adversary can make a probabilistic guess on the user position using the newly learned distribution. Under such a mechanism, the user's true location is more protected if the adversary's guess is more skewed towards other positions. This notion of privacy was first introduced in the *correctness* measure proposed by Shokri et al. [14], where location privacy is argued to be a characteristic of the adversary's probability of associating a user to her true location, instead of the size of the anonymity set. In addition, as noted in Shokri et al.'s work, the strength of the adversarial knowledge can be measured in terms of the "confusion" about the user's likely location. This is called the *(un)certainty* measure, and can also be used to evaluate the privacy preserving potential of the underlying protection mechanism. This uncertainty is maximum when each cell in the adversary's background knowledge is equally likely—maximum confusion—and minimum when one cell is singled out as the user's position. Entropy is often the preferred metric to quantify uncertainty. Note that minimum uncertainty does not imply maximum correctness.

Let $(X,Y) = (\langle x_1^{'}, x_2^{'}, ..., x_n^{'}\rangle, \langle y_1^{'}, y_2^{'}, ..., y_n^{'}\rangle)$ collectively denote the cell vectors of n users, where $(x_i^{'}, y_i^{'})$ is the cell associated to user i by the adversary. We shall use (X_i, Y_i) to denote a cell vector where the cell corresponding to user i is the true cell, i.e. $x_i^{'} = x_i$ and $y_i^{'} = y_i$. The set of all cell vectors is denoted by $V = \{(X,Y)|\forall j, (x_j^{'}, y_j^{'}) \in \mathcal{LK}_j\}$. Given a user i, a subset V_i of V contains all vectors of the form (X_i, Y_i). In the discussion below, the suffix $.X$ and $.Y$ are used to imply the set of X and Y components of the cell vectors respectively. For example, $V.X = \{X|(X,Y) \in V\}, V.Y = \{Y|(X,Y) \in V\}, V_i.X = \{X_i|(X_i,Y_i) \in V_i\}, V_i.Y = \{Y_i|(X_i,Y_i) \in V_i\}$.

We illustrate the above terminology using an example. Consider a system with two users, currently located at cell $(2,2)$ and $(3,4)$ respectively. Assuming location knowledge radius of one, an adversary's location knowledge will be:

$$\mathcal{LK}_{user1} = \{(1,1),(1,2),(1,3),(2,1),(2,2),(2,3),(3,1),(3,2),(3,3)\}$$

and

$$\mathcal{LK}_{user2} = \{(2,3),(2,4),(2,5),(3,3),(3,4),(3,5),(4,3),(4,4),(4,5)\}.$$

We can create a possible cell vector (X,Y) by picking one element from $\mathcal{LK}_{user1}$ and another from $\mathcal{LK}_{user2}$. For example, choosing the first element from $\mathcal{LK}_{user1}$ and the last from $\mathcal{LK}_{user2}$ gives us a cell vector $(X,Y) = (\langle 1,4\rangle, \langle 1,5\rangle)$. A different choice will give us another cell vector. A cell vector is therefore one possible assignment of cells to each user by the adversary. The collection of all such cell vectors is the set V. The notation (X_{user1}, Y_{user1}) means a cell vector with the correct indices, i.e. $(2,2)$, for *user1*; for example, $(\langle 2,3\rangle, \langle 2,4\rangle)$ or $(\langle 2,4\rangle, \langle 2,5\rangle)$, or V_{user1} as a representation of all possibilities. In fact, $(\langle 2,3\rangle, \langle 2,4\rangle)$ also fits the notation (X_{user2}, Y_{user2}) since $(3,4)$ are the correct indices for *user2*.

The cardinality of set V is $(2r+1)^{2n}$ and that of set V_i is $(2r+1)^{2(n-1)}$. The correctness in associating a user i to its true cell is then given as,

$$Pr[(X,Y) = (X_i,Y_i)] = \frac{(2r+1)^{2(n-1)}}{(2r+1)^{2n}} = \frac{1}{(2r+1)^2}.$$

In the context of the previous example, an adversary would correctly guess the correct cell for *user1* if the cell vector is of the form $(\langle 2,*\rangle, \langle 2,*\rangle)$—the number of possibilities in which this will happen is 9; hence the probability of correctly guessing *user1*'s location is $\frac{1}{9} (= \frac{1}{(2\times 1+1)^2})$. Note that the probability is equal to $1/|\mathcal{LK}_i|$. This probability signifies the privacy level of a user in the absence of any additional knowledge. However, we aim to evaluate the probability when knowledge about the location obfuscation mechanism (or the obfuscated location thereof) is also available to the adversary. We seek to compute the following correctness and uncertainty measures.

$$Correctness(i) = Pr[(X,Y) = (X_i,Y_i)|(x_p,y_p)], \text{ and}$$

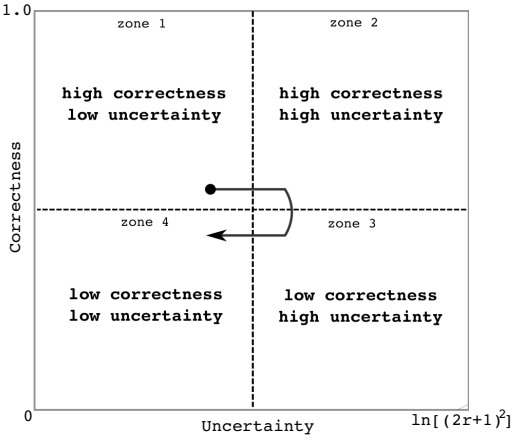

Fig. 2. Correctness and uncertainty space. Privacy mechanisms should avoid zone 1. Arrow indicates privacy improvement.

$$Uncertainty(i) = \sum_{(x,y)\in\mathcal{LK}_i} p(x,y) \ln \frac{1}{p(x,y)},$$

where $p(x,y)$ is the probability of associating user i to cell (x,y).

Correctness when using a cloaking region can be computed as the inverse of the number of cells common to $\mathcal{LK}_i$ and the cloaking region. Uncertainty in this case is $\ln[1/Correctness(i)]$. Fig. 2 shows the correctness versus uncertainty space for an adversary with knowledge radius r. A robust privacy mechanism should prevent the occurrence of high correctness and low uncertainty (zone 1) values. Points in this zone imply that the adversary has been able to eliminate a significant number of cells from $\mathcal{LK}_i$, with some cells being more probable to be the adversary's best guess. On the other hand, a high uncertainty (zone 2) implies the possibility of a uniform likelihood across the remaining cells. Most preferably, a privacy mechanism should create points having low correctness values (zones 3 and 4). The high uncertainty in zone 3 indicates that likelihoods are almost equal across the remaining cells. However, as the true cell has a low probability (low correctness), most other cells must also have similar likelihoods, implying that the adversary may not have been able to eliminate a majority of the cells in $\mathcal{LK}_i$. Zone 4 is an interesting scenario, where the adversary strongly believes (low uncertainty) that the user is at a location different from the true one (low correctness).

4.2 Computing Correctness

Given the location perturbation mechanism in Section 3.1, the probability ratio of generating the perturbed location (x_p, y_p) by any two users in an anonymity set (the set of k users) is bounded by e^{ϵ}. The Laplace noise added to a location depends on the component-wise maximum distance between two users. Note that (x_p, y_p) will satisfy the probability ratio as long as the scale parameters use these maximum distances. An adversary

can exploit this characteristic feature of the perturbation mechanism to reduce the cardinality of set V. Thereafter, a choice is made from the reduced set and used as the adversary's best guess on the locations of the users. The guess is correct if the chosen cell vector is of the form (X_i, Y_i), provided that the perturbed location was generated for user i. The correctness metric measures the chances of making a correct guess.

In order to eliminate cell vectors from V, the adversary must make an assumption on the maximum distances. Assuming maximum distances d_x and d_y, for components x and y respectively, a cell vector (X, Y) can be eliminated if $\exists i, j$ s.t. $|x_i^{'} - x_j^{'}| > d_x$ or $\exists i, j$ s.t. $|y_i^{'} - y_j^{'}| > d_y$. Alternatively, d_x is a candidate for the maximum distance in the x component if and only if the set $V(d_x) = \{(X, Y) | \forall i, j, |x_i' - x_j^{'}| \leq d_x\}$ is non-empty. Consequently, a correct association happens when the adversary's guess is also in $V_i(d_x)$. A similar condition applies to d_y. Note that the maximum x distance between two cells in any two $\mathcal{LK}_i$ provides an upped bound for d_x. Similarly, the minimum distance gives us a lower bound. Bounds on d_y can be obtained in a similar manner. Also, since the two components are independently perturbed, the correctness can be computed as the product of $Pr[X = X_i | x_p]$ and $Pr[Y = Y_i | y_p]$. Owing to the similar nature of computation, we use the generic notation C (or c) to symbolically represent either of the two components (read c as x or y, and C as X or Y). Hence, we seek to compute,

$$\begin{aligned} & Pr[C = C_i | c_p] \\ = & \sum_{d_c \text{ s.t. } |V(d_c).C| \neq 0} Pr[C = C_i | d_c] Pr[d_c] \\ = & \sum_{d_c \text{ s.t. } |V(d_c).C| \neq 0} \frac{|V_i(d_c).C|}{|V(d_c).C|} Pr[d_c]. \end{aligned}$$

Given a certain value of d_c, a brute force implementation would involve iterating through all possible vectors in the c component, verifying their membership in $V(d_c).C$, and checking if $C = C_i$. The number of possible vectors to explore in this case is $(2r + 1)^n$. We provide below an algorithm to avoid this expensive exploration for large values of r and (or) n.

4.3 A Sliding Window Approach

We present a sliding window based approach to efficiently compute the size of the sets $V(d_c).C$ and $V_i(d_c).C$. Let $c_1^{'}, c_2^{'}..., c_m^{'}$ represent the ordered domain of values for the component c, i.e. the set $\cup_j \mathcal{LK}_j.c$ arranged in increasing order. We consider a window $w_0 = [c_1^{'}, .., c_1^{'} + d_c]$ of length d_c, starting at $c_1^{'}$, and count the number of distinct values in $\mathcal{LK}_j.c$ (corresponding to user j) that appear in this window. The counts are denoted by $t_1^0, ..., t_n^0$ for users $1, ..., n$. The number of vectors contributed to $V(d_c).C$ by this window is then $\prod_j t_j^0$. Given user i, if the window includes the value c_i (the true c component of i), then the contributed number of vectors to $V_i(d_c).C$, i.e. with $c_i^{'} = c_i$, is $\prod_{j \neq i} t_j^0$.

The window is next slid over to start at $c_1^{'} + 1$. This gives us the window $w_1 = [c_1^{'} + 1, .., c_1^{'} + 1 + d_c]$. However, a simple counting as before would result in repeated vectors being counted more than once. Consider an example with $n = 3$. Assuming that the values corresponding to the users for the window $[1, .., 11]$ are $\{1, 2, 3\}_{user1}$, $\{10, 11\}_{user2}$ and $\{11\}_{user3}$, vectors $\langle 1, 10, 11\rangle, \langle 1, 11, 11\rangle, ..., \langle 3, 11, 11\rangle$ (a total of $3 \times 2 \times 1 = 6$ vectors) would contribute to $V(10).C$. When the window is moved next to $[2, .., 12]$, say the new set of possible values become $\{2, 3\}_{user1}$, $\{10, 11, 12\}_{user2}$ and $\{11, 12\}_{user3}$. In this case, all previous vectors except $\langle 1, 10, 11\rangle$ and $\langle 1, 11, 11\rangle$ would get recounted. The problem can be remedied by considering what new values are added as a result of the shifted window – $\{\}_{user1}$, $\{12\}_{user2}$ and $\{12\}_{user3}$ – and what old values are retained – $\{2, 3\}_{user1}$, $\{10, 11\}_{user2}$ and $\{11\}_{user3}$. Vectors generated from old values have already been accounted for in a previous window. New vectors must use new values for at least one user. Therefore, the number of newly added vectors is the number of vectors possible using new and old values, minus the number of vectors from old values only. In the example, this would be $(2 \times 3 \times 2 - 2 \times 2 \times 1) = 8$.

Note that since the window is shifted by one cell, each shift can result in the inclusion of at most one new value and the exclusion of at most one old value. Hence, given a window $w_p = [c_1^{'} + p, .., c_1^{'} + p + d_c]$, where $0 < p \leq (c_m^{'} - c_1^{'} - d_c)$ is an integer, the set of new values possible for any user is either empty or $\{(c_1^{'} + p + d_c)\}$, the count for user j being denoted as,

$$t_j^p = \begin{cases} 1 & , (c_1^{'} + p + d_c) \in \mathcal{LK}_j.c \\ 0 & , otherwise \end{cases}.$$

The set of old values in the window w_p consists of the old and new values in the previous window, with the possibility of one exclusion. The value that may get excluded is $(c_1^{'} + p - 1)$, depending on whether it is a possible value for a user. The number of old values to use for user j is given as,

$$h_j^p = t_j^{p-1} + h_j^{p-1} - \begin{cases} 1 & , (c_1^{'} + p - 1) \in \mathcal{LK}_j.c \\ 0 & , otherwise \end{cases},$$

where h_j^0 is equal to zero. The number of vectors added to $V(d_c).C$ by this window is then

$$\prod_j (t_j^p + h_j^p) - \prod_j h_j^p.$$

The number of vectors added to $V_i(d_c).C$ is

$$\begin{cases} \prod_{j \neq i} (t_j^p + h_j^p) & , c_i = c_1^{'} + p + d_c \\ \prod_{j \neq i} (t_j^p + h_j^p) - \prod_{j \neq i} h_j^p & , c_i \in [c_1^{'} + p, .., c_1^{'} + p + d_c) \\ 0 & , otherwise \end{cases}.$$

TABLE 1
Traffic parameters used in the simulation.

road type	mean speed	std. dev. speed	traffic volume
expressway	90 km/h	20 km/h	2916.6 cars/h
arterial	60 km/h	15 km/h	916.6 cars/h
collector	50 km/h	10 km/h	250 cars/h

These counts are obtained for each possible distance value between the upper and lower bounds. If $W(d_c)$ is the number of windows that add at least one vector to $V(d_c).C$, then we also have

$$Pr[d_c] = \frac{W(d_c)}{\sum_d W(d)}.$$

4.4 Computing Uncertainty

Note that an adversary cannot compute correctness since the true cell of the user is unknown. However, the probability of a certain cell being the true one can be computed by the adversary. The probabilities can be obtained using the sliding window approach above, iterating over the possible cells (components) of the user and assuming a cell to be the true location. The uncertainty measure is then computed from these probabilities. Highest uncertainty is achieved when the probabilities are all equal.

5 Empirical Results

We have generated a trace data set using a simulator that operates multiple mobile objects based on real-world road network information available from the National Mapping Division of the US Geological Survey. We use an area of approximately $168\,km^2$ in the Chamblee region of Georgia, USA for this study. Three road types are identified based on the available data – expressway, arterial and collector. Real traffic volume data is used to determine the number of users on the different road types [1]. The total number of users on a road type vary proportionately to the total length and traffic volume of the road type, and reciprocally to the average speed. The mean speed, standard deviation and traffic volumes on the road types are shown in Table 1. Using the number of users on each road type, the simulator randomly places them on the network and moves them around. The users move with a speed drawn from a normal distribution, randomly making turns and changing speed at junctions. The simulator maintains the traffic volume statistics while moving the users.

The used traffic volume information results in 8,558 users with 34% on expressways, 8% on arterial roads and 58% on collector roads. The trace data consists of multiple records spanning one hour of simulated time. A record is made up of a time stamp, user identifier, and x (latitude) and y (longitude) coordinates of the user's location. The granularity of the data is maintained such that the distance between successive locations of the same user is approximately 100 meters. Each user has an associated k value drawn from the range $[2, 50]$ by using a Zipf distribution favoring higher values and with the exponent 0.6. The trace data is sorted by the time stamp of records. The first minute of records is used for initialization. Location coordinates in each record thereafter are subjected to perturbation. Over 4,000,000 records are processed during a pass of the trace data.

TABLE 2
Percentage of anonymization attempts where perturbed location is at close proximity to true location.

ϵ	$\leq 1000m$	$\leq 500m$	$\leq 100m$
0.01	1.05	0.37	0.01
0.1	36.61	13.70	1.00
0.3	84.16	48.33	4.64
0.5	93.79	64.53	7.81
1.0	97.41	76.10	11.89
2.0	98.11	79.91	14.42

We evaluate the accuracy of queries on two platforms – (i) synthetic objects and (ii) real-world POIs. Synthetic queried objects are distributed randomly over the entire map based on a density value (number of objects per km^2)[1]. A *Knn*-query is issued relative to every perturbed location. Displacement is based on the great-circle distance (shortest path along the surface of a sphere). The entire map is assumed to be on a grid of $2^{14} \times 2^{14}$ divisions (a division at every meter) while calculating the Hilbert indices [15]. Objects in the same division have the same Hilbert index. For real-world POIs, we use the SimpleGeo Places API (www.simplegeo.com) to retrieve *Knn*-objects corresponding to the true and perturbed locations. Query strings are chosen to reflect different POI densities – "cafe","hospital","atm" and "police". To evaluate the correctness and uncertainty, we assume a cell to be $100m \times 100m$. All simulation results are obtained on a 2.8GHz Quad-Core Intel Xeon machine with 8GB memory and running Mac OSX 10.6.7.

5.1 Nearness

Table 2 shows the percentage of perturbations that resulted in the perturbed point being generated within $1000m/500m/100m$ of the user's actual location. A value of $\epsilon = 0.01$ effectuates to saying that two users should effectively have the same probability of generating the perturbation ($e^{\epsilon} = 1.01$). This is difficult to achieve for most values of k. As the ϵ value approaches 0.5 ($e^{0.5} = 1.65$), more than 90% of the perturbations are within 1000 meters of the true location. 60% of the points are in a much closer proximity of 500 meters. The numbers increase favorably with increasing ϵ. However, higher values of ϵ reduce the practical significance of the approach. For instance, with $\epsilon = 2.0$, we are already willing to accept a factor of 7 difference in the probability

1. The results presented here for synthetic objects differ from that in [4] due to an error we discovered in the part of the program that distributes the synthetic objects over the map. The displacement metric is also defined differently in this work—here we take an average across mismatched objects, instead of all K objects.

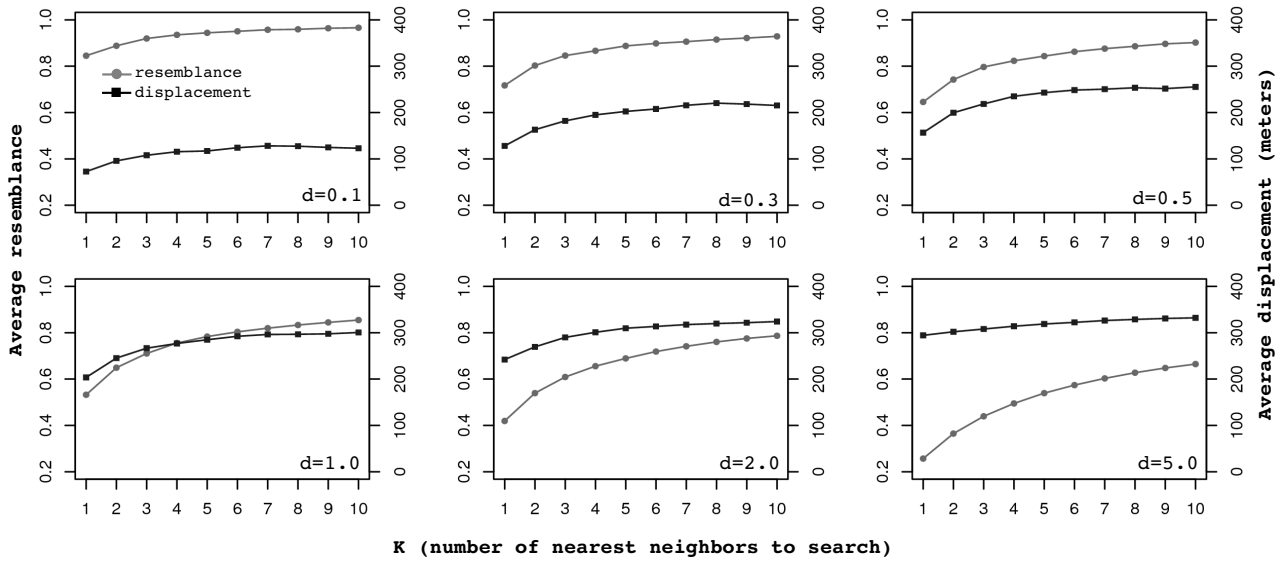

Fig. 3. Average resemblance and displacement values for Knn-queries on objects distributed with various density d. Perturbations are generated with $\epsilon = 0.5$.

estimates. Nonetheless, it is promising to see that significantly high nearness values are possible with smaller values of ϵ as well. The nearness metric is not applicable to a cloaking region based approach.

5.2 Resemblance and Displacement

Fig. 3 shows the resemblance and displacement values corresponding to different values of K (the number of nearest neighbors to retrieve) and density. The values are averaged over the the total number of requests processed (4,484,683). A density of 0.1 results in 25 objects across the entire region (sparsely distributed), while a value such as 5 results in 980 objects (densely distributed).

Subset similarity (resemblance) is over 80% for sparsely distributed objects. However, the metric shows a decreasing trend as objects become more densely situated. This is because the chances of finding points of interest in the immediate neighborhood increases as they become more closely packed, thereby reducing the subset overlap. For instance, asking for the nearest pizza place from two different locations in a city will most likely retrieve different answers, while the nearest hospital or airport is likely to be the same for both locations. Interestingly, increasing the number of nearest neighbors to search (K) improves the result set similarity. This improvement is more significant for high density objects. Retrieval of a higher number of objects can be viewed as enlarging the search radius, in which case, an object becomes more probable in the Knn set of more number of queries (locations). Continuing with the previous example, we can expect the two city locations to have a higher overlap in their lists of ten nearest pizza places. The exact extent of overlap will also depend on how close the two locations are to each other. The noise added to a location is crucial in this regard.

The displacement is within 400 meters in all cases. For sparse objects, mismatched objects are about 100 meters more distant from the true location. This increases to 300 meters for dense objects. This observation is rather counter-intuitive. Consider a $1nn$-query where the object w.r.t. the true location is not the same as the object retrieved w.r.t. the perturbed location (a mismatch). Owing to the less number of objects in space, we can say that the difference in distance to the first and the second (or other) nearest object will be higher in the low density case. Hence, displacement should also be higher in comparison to a high density object distribution. The cause for the inverse observation is grounded in the fact that mismatches happen rarely for sparse objects. Hence, on an average, the displacement value is lower than in the scenario with dense objects where mismatches are more likely.

Fig. 4 shows the average resemblance and displacement from 10,000 queries performed using the SimpleGeo Places API. The results on different query strings closely resemble those on the synthetic objects. While we observed that resemblance is poor for high density synthetic objects ($d > 1$), the real world results indicate that such high densities seldom appear. For example, dense POIs such as cafes correspond to a density value of around 0.5. Low density objects, such as a police station or an automated teller machine, can be retrieved with high accuracy. The penalty for choosing one of the incorrect results would be the traversal of an additional 100-200 meters.

Note that the resemblance measure for a cloaking region based approach is one for Knn-queries. For instance, a K nearest neighbor range query based on the HilbertCloak cloaking region will retrieve a superset of the objects that would be retrieved for a point query with respect to the true user location. The user can

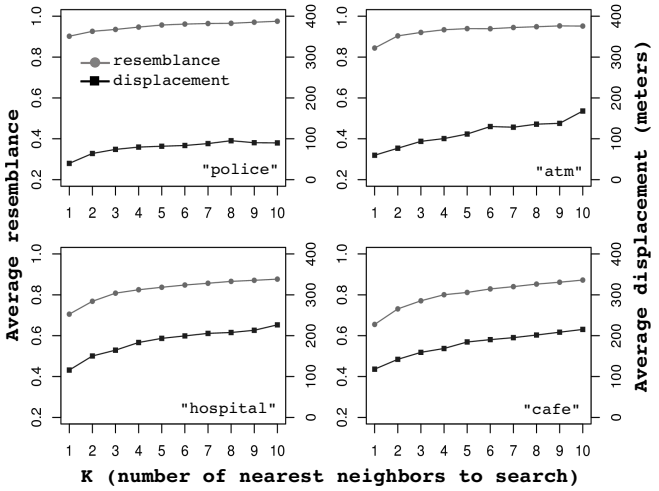

Fig. 4. Average resemblance and displacement values for Knn-queries on real world POIs. The query string is shown inside each plot. Averages are computed from the first 10,000 requests in the trace data. Perturbations are generated with $\epsilon = 0.5$.

perform a local filtering of this superset to extract the K nearest neighbors with respect to her location. Since the true result set will always be embedded in the retrieved superset, the extraction is exact and the resemblance value is one. On similar grounds, the displacement value is zero. Although accurate in terms of result retrieval, the privacy implications of a cloaking region approach can be concerning.

5.3 Correctness and Uncertainty

Fig. 5 depicts the normalized uncertainty $\left(\frac{Uncertainty}{\ln(2r+1)^2}\right)$ and correctness values corresponding to a subset of the trace data. We performed a set of experiments to cover three adversary types – a) approximate locator with access to cloaking regions (generated using Hilbert-Cloak), b) approximate locator with knowledge of the anonymity set size used in local differential perturbations, and c) a strong variant of (b) where the locator is perfect w.r.t. all but the query issuer. The approximate knowledge itself is considered at two different levels – i) dense urban localities where cell sites may be closely packed to enable accurate estimation (say $300m \times 300m$), and ii) suburban areas where approximations are larger (say $1100m \times 1100m$). These approximations can be far worse in sparsely populated areas. For the sake of clarity, we show the values from 1,000,000 randomly sampled requests, which includes the requests that generated the minimum or maximum values in the two measures. The top plots correspond to an adversary who is able to locate a user within an area of $300m \times 300m$ (radius $r = 1$). Bottom plots correspond to the more likely approximate knowledge of $1100m \times 1100m$ (radius $r = 5$).

As argued earlier, cloaking regions can result in full disclosure (correctness = 1.0) in certain situations (Fig. 5a). This is observed in the case of a strong (top plot), as well as a weak adversary (bottom plot). No robust mechanism should generate possibilities for an adversary to infer a location with high correctness and low uncertainty. The perturbed locations maintain an uncertainty level close to the uniform case (2.197 for $r = 1$ and 4.796 for $r = 5$). Correctness in certain cases improve from the prior knowledge based on the uniform distribution (Fig. 5b). However, this improvement is marginal. Further, we did not observe any strong correlation between the correctness values and the anonymity set size (k) of a user.

The perturbations maintain these characteristics (low correctness and high uncertainty) even when the adversary is a perfect locator ($r = 0$) with respect to all users except the query issuer (Fig. 5c). Strong auxiliary knowledge on other users in the anonymity set do not provide any significant advantage in inferring the true location of a user. Low correctness and high uncertainty indicates that the adversary's prior probability landscape did not undergo a significant change as a result of the attack. This is somewhat non-trivial for a strong adversary that knows the exact locations of all but one of the k users. One possible explanation can be given if the approximate location knowledge of the adversary is mostly contained inside the bounding box created by the k users. In this case, all cells corresponding to the user will be within the constraints of any eligible maximum distance value considered in sliding window algorithm.

5.4 Query Accuracy

Query results using perturbed locations are not guaranteed to contain the accurate answers. This directly points at the underlying trade-off between privacy and utility. As has been demonstrated in the database community, achieving one without sacrificing on the other may be a difficult, if not impossible, task. An evaluation of the impact of accuracy loss is needed, but is often hard to perform due to the subjective perception of utility.

For the local search application used in this study, few arguments can be made to justify that query accuracy will be maintained much better than that indicated by the resemblance metric. Contrary to the nearest neighbor based ranking assumed in the evaluation, applications such as local search often incorporate multiple other criteria in ranking their result set. For instance, local search results for "restaurants" may be ranked based on distance, price and user feedback. Given such a ranking, it is unlikely that result sets would undergo frequent changes over short distances. The nearness values indicate that a significant number of the perturbed locations are within a kilometer of the true location. Therefore, it is not unreasonable to assume that top results will undergo minor (or no) changes at the perturbed query point. Note that the underlying ranking function is considered business intelligence and is unlikely to be shared with the user. Hence, cloaking based approaches

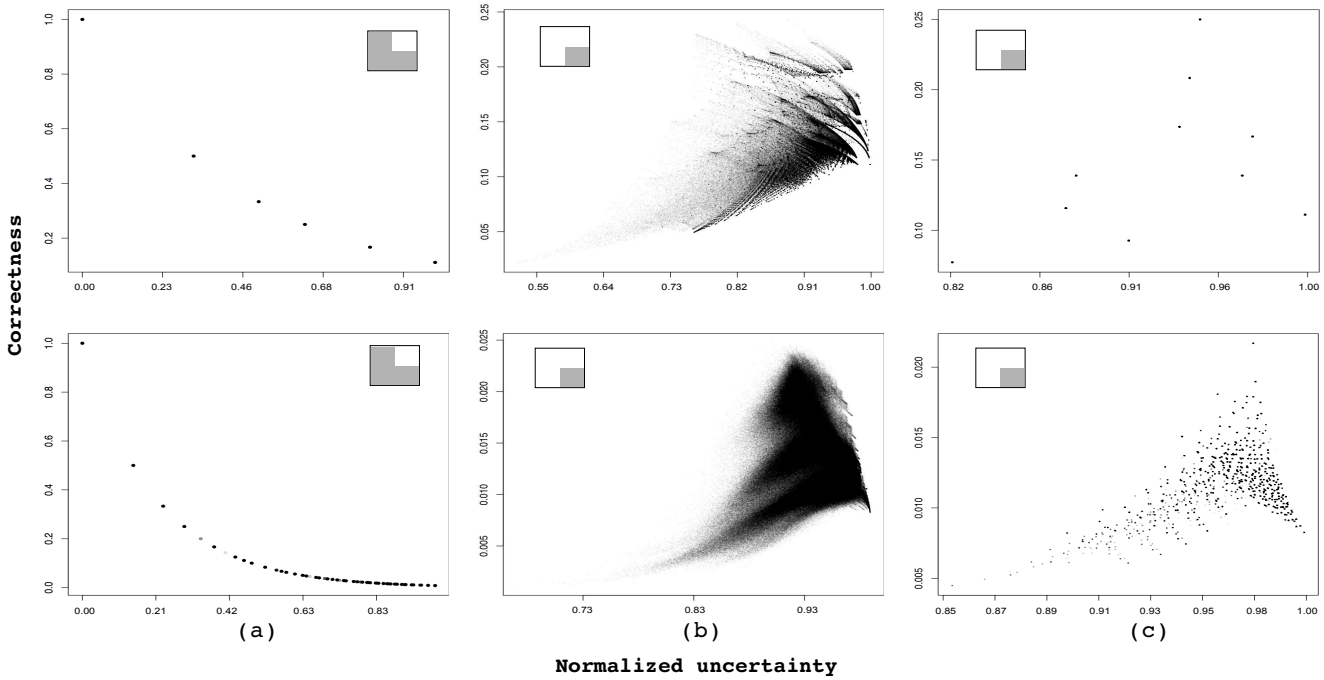

Fig. 5. Normalized uncertainty and correctness corresponding to 1,000,000 randomly sampled requests in the trace data. The areas where the points are distributed (w.r.t. Fig. 2) is shown inside each plot. Wherever applicable, knowledge radius is $r = 1$ $(300m \times 300m)$ in the top plots, and $r = 5$ $(1100m \times 1100m)$ in the bottom plots. (a) Values when cloaking region is known to approximate locator, (b) values when adversary is approximate locator corresponding to every user, (c) values when adversary is approximate locator corresponding to query issuer, and perfect locator corresponding to others.

will be unable to perform client-side extraction of the exact result set.

We opine that the final choice in this regard should come from the user. A remodeling of the LBS architecture and underlying querying mechanisms is required to enable a user-driven balancing of privacy and service accuracy. Under such a setting, an application's importance should be accounted for while making privacy decisions, and vice versa.

6 Related Work

Location obfuscation has been earlier achieved either through the use of dummy queries or cloaking regions. In the dummy query method, a user hides her actual query (with the true location) amongst a set of additional queries with incorrect locations [16], [17]. The additional processing overhead at the LBS, resulting from the dummy queries, must be addressed while using this method. Cheng et al. propose a data model to augment uncertainty to location data using circular regions around all objects [18]. They use imprecise queries that hide the location of the query issuer and yield probabilistic results. The results are modeled as the amount of overlap between the query range and the circular region around the queried objects. Yiu et al. propose an incremental nearest neighbor processing algorithm to retrieve query results [19]. The process starts with an anchor, a location different from that of the user, and it proceeds until an accurate query result can be reported. The work focuses on reducing the communication cost of the repeated querying mechanism.

Trusted third party based approaches rely on an anonymizer that creates spatial regions to hide the true location of users. The use of spatial and temporal cloaking to obfuscate user locations was first proposed by Gruteser and Grunwald [1]. Continuing on, Gedik and Liu develop a location privacy architecture where each user can specify maximum temporal and spatial tolerances for the cloaking regions [2]. Drawing inspiration from the concept of k-anonymity in database privacy [20], Gedik and Liu enforce a location k-anonymity requirement while creating the cloaking regions. This requirement ensures that the user will not be uniquely located inside the region in a given period of time. Multiple other suggestions are available on how the cloaking region should be formed. Bamba et al. enforce a location l-diversity requirement in addition, where the number of still-object counts must also be above a user-specified threshold [7]. Liu et al. propose that a minimum level of entropy should also be maintained in the queries originating from the cloaking region [21]. Dewri et al. have extended these concepts to the case of continuous services [22], [23]. Shin et al. introduce profile anonymization in cloaking regions, wherein at

least $k-1$ other users with the same profile (denoted by a vector) as the request issuer is present [24]. Riboni et al. make a similar argument, but in the context of service parameters. Inferences that can be drawn based on these parameters are avoided by smoothing the differences among the distribution of the parameters in requests from different cloaking regions [25]. Ghinita et al. propose a decentralized architecture to construct an anonymous spatial region, and eliminate the need for the centralized anonymizer [26]. In their approach, mobile nodes utilize a distributed protocol to self-organize into a fault-tolerant overlay network, from which a k-anonymous cloaking set of users can be determined. Kalnis et al. propose that all obfuscation methods should satisfy the reciprocity property [3]. This prevents inversion attacks where knowledge of the underlying anonymizing algorithm can be used to identify the actual object [10]. Parameter specification remains the biggest hindrance to real world application of these techniques.

A mix zone model is presented for location privacy by Beresford and Stajano [27]. The objective of mix zones is to prevent tracking of long-term user movements, while short-term revelation of location data is permissible. A trusted middleware usually mixes the identities of users in specific zones, thereby preventing continuous tracking. Extensions of this technique are proposed for the scenario where user movements are constrained to road networks [28].

Mokbel et al. explore query processing of different types on spatial regions – private queries over public data, public queries over private data, and private queries over private data [29]. Their effort is directed towards facilitating different query formats using cloaking regions. Lee et al. explore privacy concerns in path queries where source and destination inputs may reveal personal information about users [30]. They propose the notion of obfuscated path queries where multiple sources and destinations are specified to hide the true inputs. Historical location data is used by Xu and Cai in a variant of location k-anonymity, where the cloaking region is required to have at least k different footprints [31]. In a later work, the authors argue that the impact of a privacy parameter, such as k, on the level of privacy is often difficult to perceive. Hence, they treat privacy as a feeling-based property and propose using the popularity of a public region as the privacy level [32]. Soriano et al. show that the privacy assurances of this model do not hold when the adversary possesses footprint knowledge on the spatial regions over time [33]. Shokri et al. propose a framework to quantify location privacy based on the expected estimation error of an adversary [14]. This work provides a method to arrive at different types of inferences regarding a user's location based on a known mobility profile of the user. Using methods of likelihood estimations, the authors show that measures such as the anonymity set size or entropy, do not correctly quantify the privacy level of the user [9].

A new paradigm in location privacy is based on private information retrieval (PIR) techniques. Khoshgozaran et al. propose K nearest neighbor queries that can be reduced to a set of PIR block retrievals [34]. These retrievals can be performed using a tamper-resistant processor located at the server so that the content provider is oblivious of the retrieved blocks. Papadopoulos et al. further warrant the need to retrieve the same number of blocks across queries [35]. While the use of PIR techniques in providing location privacy is an interesting direction to explore, computational inefficiency or the dependence on additional hardware makes these approaches currently unsuitable for mainstream adoption.

7 Conclusions

Obfuscated locations can provide the means to access a location based service without risking privacy breaches. The strength of the obfuscation itself is dependent on the background knowledge of the adversary. Cloaking regions can be used to provide query privacy, but at the same time, can also enable an adversary with approximate location knowledge to improve her approximations. We propose a method based on location perturbation to address such adversaries. Perturbed locations are generated using a Laplace distributed noise function in a way such that any user, from a set of k users, is likely to be the query issuer within a parameterized bound. Empirical evaluation shows that the perturbed locations can still serve as promising query points. A high fraction of the actual result set can be retrieved, or otherwise, similarity in distances to the points of interest can be achieved. Further, using perturbed locations do not significantly improve the adversary's prior knowledge on a user's location.

Resolution of bad perturbations is an issue that remains to be explored. These are perturbations that are significantly far away from the true locations. While their occurrence has not been found to be concernedly high in the empirical study, it needs to be determined if they can be eliminated altogether. Reduced anonymity sets could be an option, specially when the size of the anonymity set has been found to have almost no correlation to the correctness measure. It should be even possible to discard the anonymity set requirement and instead perturb locations to enforce feeling-based privacy [32].

Based on the observed resemblance and displacement values, an interesting exploration would be to see the impact of location accuracy on the query results generated by an application. A lower bound on this accuracy requirement, say for a particular class of applications or object density, will reveal a default level of flexibility that a user has in perturbing her location. Perturbations can then be made freely without consulting how other users are located. With additional knowledge such as the variation in the resemblance due to different perturbed locations, a user can be in better control of balancing location privacy and service accuracy.

REFERENCES

[1] M. Gruteser and D. Grunwald, "Anonymous Usage of Location-Based Services Through Spatial and Temporal Cloaking," in *Proceedings of the 1st International Conference on Mobile Systems, Applications, and Services*, 2003, pp. 31–42.

[2] B. Gedik and L. Liu, "Protecting Location Privacy with Personalized k-Anonymity: Architecture and Algorithms," *IEEE Transactions on Mobile Computing*, vol. 7, no. 1, pp. 1–18, 2008.

[3] P. Kalnis, G. Ghinita, K. Mouratidis, and D. Papadias, "Preventing Location-Based Identity Inference in Anonymous Spatial Queries," *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, no. 12, pp. 1719–1733, 2007.

[4] R. Dewri, "Location Privacy and Attacker Knowledge: Who Are We Fighting Against?" in *Proceedings of the 7th nternational ICST Conference on Security and Privacy in Communication Networks*, 2011.

[5] C. Dwork, "Differential Privacy," *Automata, Languages and Programming*, vol. 4052, pp. 1–12, 2006.

[6] R. Shokri, J. Freudiger, and J.-P. Hubaux, "A Unified Framework for Location Privacy," in *Proceedings of the 9th International Symposium on Privacy Enhancing Technologies*, 2010, pp. 203–214.

[7] B. Bamba, L. Liu, P. Pesti, and T. Wang, "Supporting Anonymous Location Queries in Mobile Environments with Privacy Grid," in *Proceedings of the 17th International World Wide Web Conference*, 2008, pp. 237–246.

[8] M. Xue, P. Kalnis, and H. K. Pung, "Location Diversity: Enhanced Privacy Protection in Location Based Services," in *Proceedings of the 4th International Symposium on Location and Context Awareness*, 2009, pp. 70–87.

[9] R. Shokri, C. Troncoso, C. Diaz, J. Freudiger, and J.-P. Hubaux, "Unraveling an Old Cloak: k-Anonymity for Location Privacy," in *Proceedings of the 9th Annual ACM Workshop on Privacy in the Electronic Society*, 2010, pp. 115–118.

[10] G. Ghinita, K. Zhao, D. Papadias, and P. Kalnis, "A Reciprocal Framework for Spatial k-Anonymity," *Journal of Information Systems*, vol. 35, no. 3, pp. 299–314, 2010.

[11] A. Khoshgozaran and C. Shahabi, "Blind Evaluation of Nearest Neighbor Queries Using Space Transformation to Preserve Location Privacy," in *Proceedings of the 10th International Conference on Advances in Spatial and Temporal Databases*, 2007, pp. 239–257.

[12] P. Golle and K. Partridge, "On the Anonymity of Home/Work Location Pairs," in *Proceedings of the 7th International Conference on Pervasive Computing*, 2009, pp. 390–397.

[13] H. Zang and J. Bolot, "Anonymization of Location Data Does Not Work: A Large-Scale Measurement Study," in *Proceedings of the 17th Annual International Conference on Mobile Computing and Networking*, 2011, pp. 145–156.

[14] R. Shokri, G. Theodorakopoulos, J.-Y. L. Boudec, and J.-P. Hubaux, "Quantifying Location Privacy," in *Proceedings of the 32nd IEEE Symposium on Security and Privacy*, 2011, pp. 247–262.

[15] X. Liu and G. Schrack, "Encoding and Decoding the Hilbert Order," *Software-Practice and Experience*, vol. 26, no. 12, pp. 1335–1346, 1996.

[16] M. Duckham and L. Kulik, "A Formal Model of Obfuscation and Negotiation for Location Privacy," in *Proceedings of the 3rd International Conference on Pervasive Computing*, 2005, pp. 152–170.

[17] H. Kido, Y. Yanagisawa, and T. Satoh, "An Anonymous Communication Technique Using Dummies for Location-Based Services," in *Proceedings of the IEEE International Conference on Pervasive Services*, 2005, pp. 88–97.

[18] R. Cheng, Y. Zhang, E. Bertino, and S. Prabhakar, "Preserving User Location Privacy in Mobile Data Management Infrastructures," in *Proceedings of the 6th Workshop on Privacy Enhancing Technologies*, 2006, pp. 393–412.

[19] M. L. Yiu, C. S. Jensen, X. Huang, and H. Lu, "SpaceTwist: Managing the Trade-Offs Among Location Privacy, Query Performance, and Query Accuracy in Mobile Services," in *Proceedings of the 24th International Conference on Data Engineering*, 2008, pp. 366–375.

[20] P. Samarati, "Protecting Respondents' Identities in Microdata Release," *IEEE Transactions on Knowledge and Data Engineering*, vol. 13, no. 6, pp. 1010–1027, 2001.

[21] F. Liu, K. A. Hua, and Y. Cai, "Query l-Diversity in Location-Based Services," in *Proceedings of the 10th International Conference on Mobile Data Management: Systems, Services and Middleware*, 2009, pp. 436–442.

[22] R. Dewri, I. Ray, I. Ray, and D. Whitley, "On the Formation of Historically k-Anonymous Anonymity Sets in a Continuous LBS," in *6th International ICST Conference on Security and Privacy in Communication Networks*, 2010, pp. 71–88.

[23] R. Dewri, I. Ray, I. Ray, and D. Whitley, "Query m-Invariance: Preventing Query Disclosures in Continuous Location-Based Services," in *Proceedings of the 11th International Conference on Mobile Data Management*, 2010, pp. 95–104.

[24] H. Shin, J. Vaidya, and V. Atluri, "A Profile Anonymization Model for Location Based Services," *Journal of Computer Security*, vol. 19, no. 5, pp. 795–833, 2011.

[25] C. B. D. Riboni, L. Pareschi and S. Jajodia, "Preserving Anonymity of Recurrent Location-Based Queries," in *Proceedings of the 16th International Symposium on Temporal Representation and Reasoning*, 2009.

[26] G. Ghinita, P. Kalnis, and S. Skiadopoulos, "PRIVE: Anonymous Location-Based Queries in Distributed Mobile Systems," in *Proceedings of the 16th International Conference on World Wide Web*, 2007, pp. 371–380.

[27] A. R. Beresford and F. Stajano, "Mix Zones: User Privacy in Location-Aware Services," in *Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops*, 2004, pp. 127–131.

[28] B. Palanisamy and L. Liu, "MobiMix: Protecting Location Privacy with Mix-Zones Over Road Networks," in *Proceedings of the 27th International Conference on Data Engineering*, 2011, pp. 494–505.

[29] M. F. Mokbel, C. Chow, and W. G. Aref, "The New Casper: Query Processing for Location Services Without Compromising Privacy," in *Proceedings of the 32nd International Conference on Very Large Data Bases*, 2006, pp. 763–774.

[30] K. C. K. Lee, W.-C. Lee, H. V. Leong, and B. Zheng, "OPAQUE: Protecting Path Privacy in Directions Search," in *Proceedings of the 25th International Conference on Data Engineering*, 2009, pp. 1271–1274.

[31] T. Xu and Y. Cai, "Exploring Historical Location Data for Anonymity Preservation in Location-Based Services," in *IEEE INFOCOM 2008*, 2008, pp. 1220–1228.

[32] T. Xu and Y. Cai, "Feeling-Based Location Privacy Protection for Location-Based Services," in *Proceedings of the 16th ACM Conference on Computer and Communications Security*, 2009, pp. 348–357.

[33] M. Soriano, S. Qing, and J. Lopez, "Time Warp: How Time Affects Privacy in LBSs," in *Proceedings of the 12th International Conference on Information and Communications Security*, 2010, pp. 325–339.

[34] A. Khoshgozaran, C. Shahabi, and H. Shirani-Mehr, "Location Privacy: Going beyond k-Anonymity, Cloaking and Anonymizers," *Journal of Knowledge and Information Systems*, vol. 26, no. 3, pp. 435–465, 2011.

[35] S. Papadopoulos, S. Bakiras, and D. Papadias, "Nearest Neighbor Search with Strong Location Privacy," *VLDB Endowment*, vol. 3, no. 1-2, pp. 619–629, 2010.

Rinku Dewri is an Assistant Professor in the Computer Science Department at University of Denver. He obtained his Ph.D. in Computer Science from Colorado State University. His research interests are in the area of information security and privacy, risk management, and multi-criteria decision making. He is a member of the IEEE and the ACM.