2005

Querying Private Data in Moving-Object Environments

Reynold Cheng

Yu Zhang

Elisa Bertino
Purdue University, bertino@cs.purdue.edu

Sunil Prabhakar
Purdue University, sunil@cs.purdue.edu

Report Number:
05-015

QUERYING PRIVATE DATA IN MOVING-OBJECT ENVIRONMENTS

Reynold Cheng
Yu Zhang
Elisha Bertino
Sunil Prabhakar

CSD TR #05-015
July 2005

Submission no: 402

Querying Private Data in Moving-Object Environments

Reynold Cheng
Department of Computing
The Hong Kong Polytechnic University
Hung Hom, Kowloon, Hong Kong
Email: csckcheng@comp.polyu.edu.hk

Yu Zhang Elisa Bertino Sunil Prabhakar
Department of Computer Science
Purdue University
West Lafayette, IN 47907-1398, USA
Email: {zhangyu,bertino,sunil}@cs.purdue.edu

Abstract

Location-based services, such as finding the nearest gas station, require users to supply their location information. However, a user's location can be tracked without her consent or knowledge. Lowering the spatial and temporal resolution of location data sent to the server has been proposed as a solution. Although this technique is effective in protecting privacy, it may be overkill and the quality of desired services can be severely affected. In this paper, we investigate the relationship between uncertainty, privacy, and quality of services. We propose using imprecise queries to hide the location of the query issuer and evaluate uncertain information. We also suggest a framework where uncertainty can be controlled to provide high quality and privacy-preserving services. We study how the idea can be applied to a moving range query over moving objects. We further investigate how the linkability of the proposed solution can be protected against trajectory-tracing.

1 Introduction

Positioning technologies such as GPS, GSM, RF-ID and WiFi(802.11) have undergone rapid developments in recent years [19, 21, 7]. These new technologies allow locations of users to be determined accurately, and enable a new class of applications known as Location-Based Services (LBS). An important LBS is the E-911 application mandated by the U.S. (correspondingly E-112 in Europe), which requires cell phone companies to provide an accurate (within a few hundred feet) location of a cell phone user that calls for emergency help [7]. Another example is the use of RF-ID tags on items such as razors in large departmental stores for inventory management [21].

Although LBS applications hold the promise of safety, convenience, and new business opportunities, the ability to locate users and items accurately also raises a new concern – intrusion of *location privacy*. According to [2], location privacy is defined as "the ability to prevent other parties from learning one's current or past location". Using locationing technologies, a service provider can track the whereabouts of a user and discover her personal habits. These pieces of sensitive information can be sold to unknown third parties. It is often feared that government agencies can monitor the behavior of individuals, the places they have visited, etc. Preventing location privacy from being invaded is thus of utmost importance.

Recently several solutions for location privacy protection have been proposed. Some researchers suggest the use of "policies", in which the service provider is required to state explicitly how user's location information can be used [20, 10, 9]. In another proposal, a user "cloaks" her information before sending it to the LBS, by providing her location at a lower resolution in terms of time and space [7, 2]. In other words, rather than giving a precise location and time instant, a larger region covered in a time frame is reported. This solution, also known as *location cloaking*, provides the user with more flexibility in controlling her information. We will study it extensively in this paper.

By reducing the granularity of spatial and temporal information, location cloaking allows a user's privacy to be be better protected. Unfortunately, this scheme can also reduce the quality of service provided by the LBS. This is simply because the LBS does not have the most accurate information to provide the best service. Consider a remote cab service that allows a subscriber to call for a cab nearby. If the subscriber reports her precise location, the service provider can find her the closest cab, and can tell the cab driver how to reach the customer. However, if only a vague location is given, it may take more time for a cab to reach the customer. Indeed, for such a scheme, there is a tradeoff among: (1) How uncertain the location information sent by a user to the LBS is, (2) the location privacy of the user, and (3) the service quality. In this paper, we propose a framework designed for moving-object environments. The model takes into account these three factors, allowing us to have a better

understanding of their interaction. We also present a formal model for cloaked locations, and provide metrics for quantifying privacy of location cloaking.

We then investigate the role of the location-cloaking framework for non-anonymous application, where the owner of the location is reported to the service provider, in addition to the location data itself. We choose this type of applications because existing techniques usually focus on anonymity or pseudonymity of the users' identities, and it is not clear how they can be applied to non-anonymous solutions. Moreover, non-anonymous location-based applications post extra difficulties in privacy protection due to the fact that the owner of the location is also known to the service provider.

A non-anonymous query studied extensively in this paper is the *moving range query* (MRQ), where a user is notified any object of interest within a fixed distance from her current location. This query is well studied in spatial-temporal database literature (e.g.,[14, 15]). Here we study an "imprecise" version of moving range query, namely IMRQ. Essentially, an IMRQ processes cloaked locations instead of precise locations. Moreover, since the location of the query issuer is also inexact, the query itself also carries uncertain information. Due to the uncertainty of the query and data, the query result is "imprecise", and probabilistic guarantees are augmented to the answers. For example, an answer for IMRQ: $\{(S_1, 0.4), (S_2, 0.8)\}$ means that users S_1 and S_2 have probabilities of 0.4 and 0.8 respectively of satisfying the query. We develop query processing algorithms for computing probabilistic answers for IMRQ, based on spatial database techniques.

We also study the quality metric of IMRQ, in order to quantify the ambiguity due to the inexactness of cloaked location data. We define two different metrics, one based on uncertainty in the database, and the other based on the ambiguity of the query. These scoring metrics can be used to quantify the quality of a service, allowing the user to decide whether she should adjust the granularity of her cloaked location information in order to attain a better service. Extensive simulations are performed to study how these quality metrics fare in a moving-object environment.

Finally, we address the issues of inference attacks, where future locations can be inferred based on tracing movement in the past. We study modifications to our approach in order to prevent the linkability between a user's identity and locations from being increased, thereby reducing the impact of this kind of threats.

To summarize, our major contributions are:

- A framework that relates location cloaking, privacy and quality of service;
- A formal model of cloaking and privacy metrics;
- An evaluation algorithm for IMRQ that manipulates cloaked data;
- Quality metrics for IMRQ based on data and query imprecision;
- Experimental results for the proposed scheme; and
- Inference attacks and protection for the scheme.

The rest of this paper is organized as follows. We propose a framework to capture data uncertainty, privacy and quality of service in Section 2. In Section 3, we formally present the definitions of non-anonymous applications, location privacy, cloaking and service quality. Section 4 presents a querying algorithm, and Section 5 describes service quality metrics for moving-range queries. Experimental results are presented in Section 6. Section 7 investigates the problems of location inference and their corresponding solutions. Related works are presented in Section 8. We conclude the paper in Section 9.

2 A Framework for Balancing Privacy and Service Quality

Let us now describe a system model that connects privacy, cloaked information and service quality. It forms the basis for subsequent discussions.

Figure 1 illustrates this framework. Its main idea is to allow the user to specify her location, service request and privacy requirements to the **cloaking agent**, which then produces the cloaked location and an "imprecise" service request. On receiving these pieces of information, the service provider processes the request and sends back the service and feedback to the user.

Inside the cloaking agent, the **policy translator** produces a cloaked location (i.e., a larger region) based on the (precise) location of the user as well as her privacy requirements, which can be specified using some high-level languages such as EPAL [1] and P3P [6]. For instance, if the user's requirement is "generate a cloaked location that covers five buildings when I am in Area X", the policy translator produces the corresponding cloaked location when it detects the user is in Area X. The cloaked location produced is then directed to the **service translator**.

Based on the cloaked location and the service request, the service translator produces an "imprecise" service request. For example, the MRQ is a service request from the user, and the service translator transforms the MRQ to IMRQ, an imprecise service request that processes cloaked location data. Both the cloaked location and the imprecise service request are then shipped to the **imprecise service processor**, which stores the cloaked location in a spatial-temporal database and processes the service request. Since

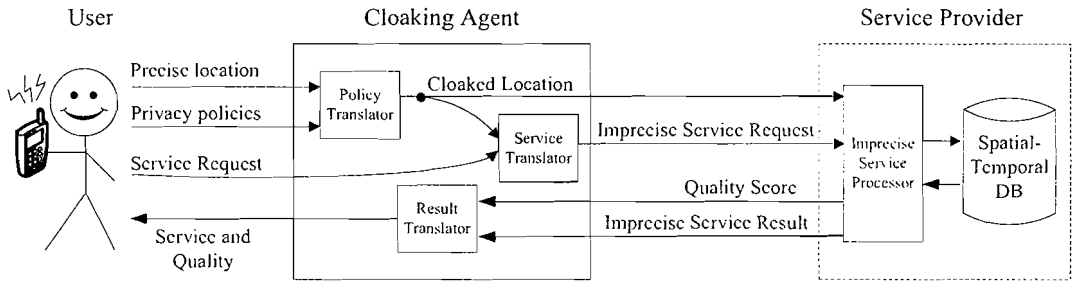

Figure 1. Managing Privacy and Service Quality with Cloaking Agent.

location values are imprecise, the service processor produces a "probabilistic service result" i.e., answers are augmented with probability to indicate the confidence of their presence [3]. For example, the result of IMRQ contains user names together with their probabilities. In addition, a score indicating the quality of the service is generated. These technical issues are detailed in Sections 4 and 5.

Both the probabilistic service result and the quality score can be transferred directly to the user, or optionally to the **result translator** inside the cloaking agent. The main purpose of the result translator is to hide the technical details of the probabilistic service result (e.g., probability, quality scores), and converts the answers to a higher-level form that even casual users can understand. For example, for an IMRQ, the translator can choose to return only the names for which there is a high confidence (e.g., $p_j > 0.8$) and not return any probability value. It can also describe to the user the quality as `LOW`, `MEDIUM` and `HIGH` for quality score ranges between $[0, 0.2], [0.2, 0.8], [0.8, 1]$ respectively, instead of requiring the user to interpret the numerical values. Based on the recommendation from the cloaking agent, the user can then decide if the degree of privacy should be reduced.

3 Privacy, Cloaking, and Service Quality

In this section, we outline the classification of LBS, based on which non-anonymous applications are defined. We then explore a formal model of location cloaking, based on which privacy is defined.

3.1 Classification of Location-based Services

An LBS application can be classified according to how the identity of the owner of the location information is disclosed along with the location information. In general, there are three classes of LBS applications [2]:

1. **Anonymous:** This application class works with location information only, and does not require a user's identity. For example, in querying a LBS about the price of a coffee when approaching a coffee shop, a user only needs to supply her location to the LBS.

2. **Pseudonymous:** This type of applications needs to know the identity of a user, but it can use the user's pseudonym, rather than her real identity. An example is: "When I walk past a computer kiosk, display my emails". The LBS can use the user's pseudonym, rather than her real name, to retrieve her emails.

3. **Non-Anonymous:** This application class does not work without knowing a user's true identity. A typical example is: "When I am inside the building, let my project groupmates know where I am".

These three classes of applications are arranged in the ascending order of the amount of information about the owner of the location information is disclosed. The more the information is disclosed, the higher is the risk to the intrusion of privacy. We are interested in studying the protection of privacy for **non-anonymous** applications, which involves more privacy-related information than the other two service classes.

Our framework, however, is not limited to non-anonymous applications – it can be applied to anonymous or pseudonymous applications as well.

3.2 Protecting Privacy by Cloaking

A non-anonymous application is defined formally as follows.

Definition 1 Non-anonymous Application: *a user supplies to the service provider a tuple* $(\texttt{UserID}, L(t), \texttt{request})$ *at time* t*, where* `UserID` *is the identity of the user and* $L(t)$ *is the location of the user at time* t *with coordinates* $(x(t), y(t))$*. On receiving this tuple, the service provider processes the* `request` *and returns the service results to the user.*

When a service provider receives the request, it can associate the identity of the user (`UserID`) with her current location $L(t)$. By correlating $L(t)$ with a map, it is easy to obtain the region the user is in. If the area is *sensitive* [8], e.g., a hospital or the house of a political leader, the user's privacy may be threatened, since this information can be sold to third parties without the user's consent. The user's identity is said to have a high degree of "linkability" with her location (using the definition of "linkability" in [17]).

The purpose of protecting privacy for non-anonymous applications is to reduce the degree of linkability. One way to do this is to require the service provider to state its policies of using the user's location information [20, 10, 9]. However, this places the burden of privacy protection to service providers, and it is often doubtful whether these policies are enforced adequately. Even if these policies are implemented correctly, location privacy can still be breached if attackers obtain this information through the communication channel. In this paper we use a complementary technique called *cloaking*, where the user takes a better control over linkability by adjusting the degree of accuracy of the spatial information sent to the service provider [7, 2]. Let us assume the system has n users with names $S_1, S_2, \ldots, S_n$. Also, the current location of each user S_i is $L_i(t)$. We can define cloaking as follows.

Definition 2 Cloaking: *A user S_i reports to the service provider a closed region called uncertainty region, denoted $U_i(t)$, such that $L_i(t)$ is inside $U_i(t)$.*

When the service provider receives the uncertainty region, it perceives that each point of the region has an equal chance of being the user's true location i.e., the probability density function (pdf) of the user's location within the uncertainty region is $\frac{1}{\text{Area}(U_i(t))}$. Hence the service provider does not know the user's precise location. Unless stated otherwise, we also assume that the uncertainty region information received by the server does not change until new location data is reported. Figure 2 shows the difference between an exact location and a cloaked location. It also illustrates that the user's location is uniformly distributed within the region from the service provider's perspective.

3.3 Measuring Privacy of Cloaking

By "injecting" different amount of spatial uncertainty to her location, cloaking provides a simple way for a user to control the release of her private information to untrusted parties. The degree of privacy can be measured in two ways: (i) size of uncertainty region and (ii) coverage of sensitive area.

1. Size of uncertainty region. By providing a larger uncertainty region, the spatial resolution of a location is reduced, making the user's location more difficult to be guessed. The size of the uncertainty region can thus be used to reflect the degree of privacy: the larger the region size, the more the privacy.

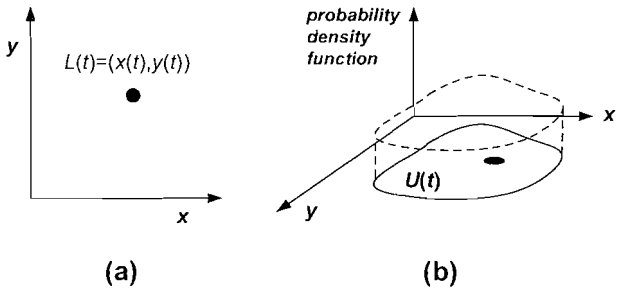

Figure 2. Exact and cloaked location.

2. Coverage of sensitive region. The second means of quantifying privacy depends on the location of the user. To see this, assume the size of the uncertainty region is fixed. Suppose the user is inside a hospital (which she does not want people to know about this), and her uncertainty region has a fraction of 90% overlap with the hospital. One can easily guess she is in the hospital. On the other hand, if the user is shopping in a mall, she may not be very concerned even if her location is known.

From this example, we can see that whether the user's located in a "sensitive region" (e.g., hospital, nightclub) affects the degree of privacy. Based on this observation, we define the "coverage" of sensitive region for user S_i as follows:

$$\text{Coverage} = \frac{\text{Area(sensitive regions of } S_i \cap U_i(t))}{\text{Area}(U_i(t))} \quad (1)$$

In general, the higher the coverage, the lower the privacy. In the previous example, the coverage is 90%, and thus the user can be easily guessed that she is in the hospital. Thus the uncertainty region should be enlarged in order to assure that the user's location cannot be easily associated with a sensitive region.

It is also worth mention that the definition of sensitive region is user-specific. For example, while for a physician a hospital may not be a sensitive region, the same cannot be said about a patient.

3.4 Cloaking and Service Quality

Although cloaking lessens the threat to location privacy, it can affect the *quality* of service provided. In particular, since the service provider does not receive accurate location information, it may be impossible for it to provide a good service. For example, suppose a user wants to know who is her nearest neighbor, and her cloaked location is supplied.

Then there can be more than one answer that satisfies her query, and the user may be unable to get a precise answer. Next we study the technical details of querying cloaked locations and measuring query quality.

4 Evaluation of Imprecise Queries

In this section, we study the technical details of the evaluating cloaked locations in a database system. We first discuss how a traditional query can be "transformed" to a query that handles cloaked information. We then illustrate how the query can be evaluated in a spatial database. We also examine the quality of moving range queries. The *moving range query* is used as a running example.

4.1 Precise and Imprecise Queries

Intuitively, a moving range query is a range query whose "range" depends on the position of the user. For example, a user may specify that she wants to be notified of any of her friend who is within ten meters from her. The reader is reminded that although here we assume a range query has a circular shape our methods can be applied to range queries with any geometric shape.

Let F_i be the set of users in which S_i is interested, and let r_i be the radius of the circle with $L_i(t)$ as the center. We can define a moving range query as follows.

Definition 3 *Given a user S_i with parameters F_i, $L_i(t)$ and r_i, a* **Moving Range Query (MRQ)** *returns $\{S_j | j = 1, \ldots, n\}$, such that $S_j \in F_i$, and S_j has a distance less than r_i units from S_i at time t.*

Figure 3(a) illustrates a MRQ. If we assume $F_1 = \{S_2, S_3, S_4\}$, then S_4 is returned as the only answer. Note that to answer MRQ, the system needs to know both the location and identity of each user so that the query can be answered. It is thus a non-anonymous query. Further, when a user submits a MRQ, the user needs to submit her name in addition to her current position, so that only the names of the people of interest to her are returned. As discussed in the last section, privacy can be threatened since both the identity and the location information are supplied to the service provider.

Location cloaking can alleviate the threat to privacy. Instead of supplying exact locations, users only supply their cloaked locations. We call the version of MRQ that employs cloaked location information **Imprecise Moving Range Query**. The word "imprecise" arises from the fact that the query is made ambiguous by imposing uncertain information on the location of the user submitting the query. It is formally defined below:

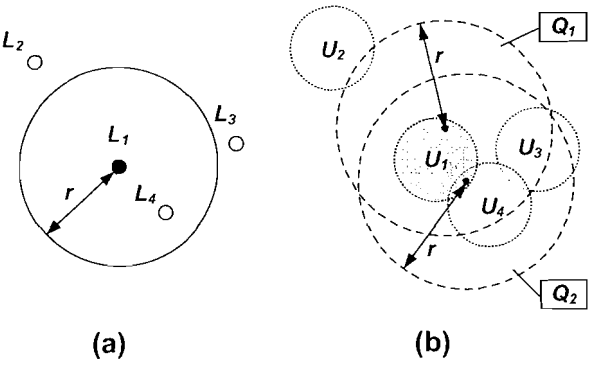

Figure 3. Moving Range Query using (a) exact locations, and (b) cloaked locations.

Definition 4 *Given a user S_i with parameters F_i, $U_i(t)$ and r_i, an* **Imprecise Moving Range Query (IMRQ)** *returns a set of tuples $\{(S_j, p_j) | j = 1, \ldots, n\}$, where $S_j \in F_i$, and $p_j > 0$ is the non-zero probability that user S_j has a distance less than r_i units from S_i at time t.*

Figure 3(b) shows a scenario where an IMRQ is computed over cloaked locations, with range queries issued at two different locations in $U_1(t)$. For Q_1, the answer is $\{(S_2, 0.2), (S_3, 0.6), (S_4, 0.7)\}$, while for the Q_2, the answer is $\{(S_3, 0.9), (S_4, 1)\}$. After considering the probabilities of the objects satisfying the range queries issued at all possible points in $U_1(t)$, the answer $\{(S_2, 0.1), (S_3, 0.7), (S_4, 0.9)\}$ is returned. The probabilities in the answer allow the user to place appropriate confidence in the answer, which is the consequence of evaluating cloaked (or imprecise) location values. Depending upon the requirements of the application, one may choose to report only the object with the k highest probability value, or only those objects whose probability values exceed a minimum threshold. Our proposed work will be able to work with any of these models. Now let us examine how IMRQ can be evaluated.

4.2 Evaluation of IMRQ

Given a MRQ and a cloaked location $U_i(t)$, computing its corresponding IMRQ involves two main steps:

1. Transformation Phase, which converts the MRQ to the IMRQ, and
2. Evaluation Phase, which computes probabilistic answers for the IMRQ.

Transformation Phase. In MRQ, the query range of the user S_i is a circle C_i with radius r_i and center $L_i(t)$. If the

user transmits her cloaked location, the query range is no longer C_i, since the service provider has no idea of where $L_i(t)$ exactly is. The service provider does know that $L_i(t)$ is within $U_i(t)$, so it transforms the query into sub-queries over all possible locations of S_i. In other words, at each point $(u, v) \in U_i(t)$, a query is issued to find out which users are within the region $C_i'(u, v)$, where $C_i'(u, v)$ is the circle with radius r centered at (u, v). The result of IMRQ is essentially the union of the results of the range queries issued at each point in $U_i(t)$. The transformation potentially covers more objects than MRQ. In Figure 3, for example, the converted ranges in (b) overlap with $\{S_2, S_3, S_4\}$ while the original query in (a) only covers S_4.

Evaluation Phase. Since the location of each object is uncertain, each user only has some chance of satisfying the IMRQ. In particular, if $S_j \in F_i$, then the probability $p_j(u, v)$ of user S_j satisfying S_i's request at point $(u, v) \in U_i(t)$ is given by

$$p_j(u, v) = \frac{\text{Area}(U_j(t_j) \cap C_i'(u, v))}{\text{Area}(U_j(t_j))} \quad (2)$$

where $t_j \leq t$ is the time instant of the latest value of U_j, and $U_j(t_j) \cap C_i'(u, v)$ is the common region between $U_j(t_j)$ and $C_i'(u, v)$. For simplicity, we assume $U_j(t_j) = U_j(t)$.[1] Essentially, $p_j(u, v)$ is the fraction of $U_j(t)$ that overlaps $C_i'(u, v)$.

The total probability of S_j satisfying the IMRQ issued by S_i is given by the integration of the product of the pdf of user S_i's location at (u, v) (i.e., $\frac{1}{\text{Area}(U_i(t))}$) and $p_j(u, v)$ over all $(u, v) \in U_i(t)$. Therefore,

$$p_j = \int_{U_i(t)} \frac{1}{\text{Area}(U_i(t))} p_j(u, v) du dv \quad (3)$$

$$= \frac{\int_{U_i(t)} \text{Area}(U_j(t_j) \cap C_i'(u, v)) du dv}{\text{Area}(U_i(t)) \text{Area}(U_j(t_j))} \quad (4)$$

by substituting $p_j(u, v)$ with Equation 2. The probability value so computed serves as an indication of the confidence placed on the answer. For example, in Figure 3(b), p_2 is only 0.1, showing that S_2 is unlikely to be answer, while S_3 and S_4 have a much higher chance (0.5 and 0.9 respectively) of being the answers.

4.3 Query Implementation

We now address the implementation issues of IMRQ presented in the last section.

We assume the service provider maintains a spatial-temporal database system for storing the location information of each user. Let T be a relation with two attributes *<user-name, region>*, which stores the identity and the geometry of the current uncertainty region of all users. Figure 4 describes an evaluation algorithm for IMRQ.

Input

T /* relation containing *<ID, uncertainty region>* of all users */
$S_i, U_i(t)$ /* identity and uncertainty region of user S_i */
F_i, r_i /* parameters of IMRQ for user S_i */
M /* resolution of IMRQ */

Output

(S_j, p_j) /* names and probabilities of users that satisfies IMRQ */

Transformation Phase

1. Divide $U_i(t)$ into M equal subregions.
2. Let q_m ($m = 1, \ldots, M$) be the midpoint of the m-th subregion.
3. Let I be a relation with attributes *<user-name, region>*.
4. **for** $m \leftarrow 1, \ldots, M$ **do**
   a. Let the m-th row of I be $< S_i, C_i'$(midpoint of m-th subregion) $>$

Evaluation Phase

5. Let V be a relation with attributes $<$ *user-name*, *prob* $>$.
6. Evaluate the following query:

```
INSERT INTO V VALUES
(SELECT T.ID, Area(Intersection(I.region, T.region))/Area(T.region)
FROM I, T
WHERE Overlaps(I.region, T.region)
AND I.ID <> T.ID
AND (T.ID INTERSECT Fi) <> NULL;
```

7. Evaluate the following query:

```
SELECT ID, SUM(prob)
FROM V
GROUPBY ID;
```

Figure 4. Evaluating an IMRQ.

In this algorithm, the first four steps correspond to the **Transformation Phase**. Steps 1 and 2 partition $U_i(t)$ into M subregions, where M is called "resolution" and is a parameter that controls the precision of the query answer. The range query region formed by each midpoint of the subregion is inserted to relation I (Steps 3 and 4).

In the **Evaluation Phase**, a spatial-join using the *Overlaps* predicate is performed between the range query region of I and the uncertainty region of T (i.e., the tuple pairs that have non-zero overlap are joined [18]). These joined tuples correspond to users that satisfy any of the queries formed by the midpoints of the subregions. Out of these join pairs, only the identities of users who are the members of F_i are inserted, together with their probabilities (Equation 2), to relation V (Steps 5 and 6). Notice that the *Intersection*

[1] The possible locations of S_j at time t may be derived from the location at t_j if the maximum speed of S_j is known. We investigate this issue in Section 7.

function evaluates the geometry of the common region of two given regions, while the *Area* function returns the area of a given region, which can be computed using well known algorithms from the spatial database literature [18]. Finally, Step 7 sums up all the probability values that belong to the same user in the relation V, corresponding to Equation 4. It returns the identity and probability of each user that satisfying the IMRQ.

This algorithm can be implemented by PL/SQL and any spatial database system that supports the *Overlap* join, *Intersection* and *Area*. Also, for presentation purpose, we perform two queries in Steps 6 and 7, but they may be combined into a single query for efficiency.

Complexity. Steps 1 to 4 take $O(M)$ times. The worst case of Step 6 needs $O(Mn)$ times and Step 7 needs $O(Mn)$ times. Thus the complexity of the algorithm is $O(M + Mn + Mn) = O(Mn)$. In practice, many efficient spatial join techniques based on z-ordering trees and R-trees [18] can significantly improve the cost of evaluation.

5 Quality of Imprecise Queries

Due to the inherent imprecision in location data and the query itself, an imprecise query returns probabilistic answers. In this section we try to answer the question: how ambiguous is an answer? We investigate the notion of quality of imprecise queries, which can serve as a hint for the query issuer on whether she should adjust the degree of her location uncertainty. There are two types of quality metrics: one due to the inexactness of data, and the other one due to the ambiguity of the a query.

5.1 Quality Due to Data Imprecision

The first factor that produces answer uncertainty is the ambiguity of cloaked location data. This ambiguity is reflected by the probability of the query answer. Here we modify the metric for probabilistic query range queries described in [3, 13].

For example, for an IMRQ, the result is the clearest if we are sure that S_j is either completely inside or outside the query range; p_j equal to 100% and 0% respectively. Uncertainty arises when we are less than 100% sure whether the location of S_j is inside the query range. This corresponds to the case when the uncertainty region of S_j, i.e., $U_j(t)$, only lies partially inside S_i's query range. The most ambiguous case happens when p_j is 0.5 i.e., S_j has a half chance of being inside the range. Hence a reasonable metric for measuring the quality of an answer due to p_j is:

$$\frac{|p_j - 0.5|}{0.5} \tag{5}$$

The value of Equation 5 varies between 0 to 1, with a larger value representing a better quality. We can define the *data score* of an IMRQ as the average of the values evaluated in Equation 5 for all objects that satisfy the IMRQ:

$$\text{Data score for } S_i = \frac{1}{|R_i|} \sum_{j \in R_i \wedge j \neq i} \frac{|p_j - 0.5|}{0.5} \tag{6}$$

where R_i is the set of tuples (T_j, p_j) returned by an IMRQ for S_i.

Metrics for quantifying the quality of answers exist for other queries like nearest-neighbor and SUM, and readers are referred to [3] for more details. Also notice that the quality defined here depends on the location data of users being queried. Next, we present quality metrics due to the uncertainty of the query issuer herself.

5.2 Quality Due to Query Imprecision

Recall from the Evaluation Phase that the answer to IMRQ is in fact the union of the answers to the sub-queries (with range $C_i'(u, v)$), executed over the uncertainty region of the query issuer S_i. Out of these range queries, only one is correct. The union operation can potentially produce incorrect answers (called false positives in [16]), due to the imprecision of the location of the query issuer. Here we present a metric for computing quality of an answer due to the uncertainty of the IMRQ.

Let us assume that each sub-query returns a set of answers $Q_i'(u, v)$. Also, suppose there are m distinctive results, $R_{i,1}, \ldots, R_{i,m}$, for all the sub-queries. Let R_i be the set of identities returned by IMRQ, and thus $R_i = \bigcup_{k=1}^{m} R_{i,k}$. Let $p(R_{i,k})$ be the probability that $R_{i,k}$ is the true result. Then $p(R_{i,k})$ is also the probability that user S_i gets the answer $R_{i,k}$:

$$p(R_{i,k}) = \int_{(u,v) \in U_i(t) \wedge R_{i,k} = Q_i'(u,v)} \frac{1}{Area(U_i(t))} dudv \tag{7}$$

that is, the integration of uniform pdf over all points in $U_i(t)$ that evaluate the same result $R_{i,k}$. Note that $\sum_{k=1}^{m} p(R_{i,k}) = 1$.

We also define the *precision* of R with respect to $R_{i,k}$ as

$$V(R_{i,k}) = \frac{|R_{i,k}|}{|R_i|} \tag{8}$$

where $V(R_{i,k})$ indicates the amount of "impurities" injected to $R_{i,k}$ assuming $R_{i,k}$ is the correct answer. Note that $V(R_{i,k})$ varies from 0 to 1, with a higher value indicating a higher precision.

The *query score* of IMRQ can then be measured by

$$\text{Query score for } S_i = \sum_{k=1}^{m} p(R_{i,k}) V(R_{i,k}) \tag{9}$$

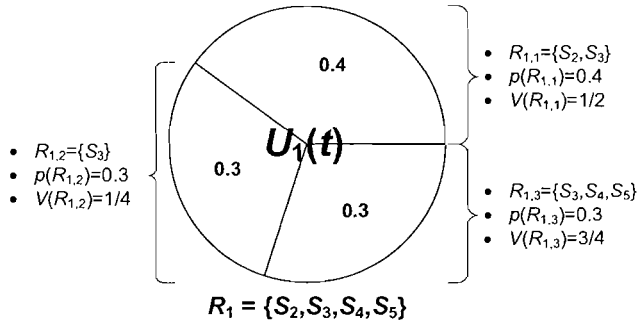

Figure 5. Illustrating the query score of IMRQ.

which varies between 0 (lowest quality) and 1 (highest quality). To understand this metric, let us look at Figure 5, which shows three distinct answers for query issuer S_1, and also the probability that S_1 yields each of the answer (i.e., the fraction of $U_1(t)$ that yields the answer). Since S_1 is located at only one point in $U_1(t)$, only one of the three answers is correct. Suppose $R_{1,3}$ is correct. Then its precision $V(R_{1,3})$ is 3/4, since S_2 is a false positive. The value $p(R_{1,3})$ is the probability that S_1 gets the answer $R_{1,3}$, which is 0.3. The query score of this IMRQ is thus the weighted sum of the $V(R_{1,k})$'s, that is, $0.4 \cdot \frac{1}{2} + 0.3 \cdot \frac{1}{4} + 0.3 \cdot \frac{3}{4} = 0.5$.

Implementation of Query Score. Similar to the transformation phase of IMRQ, the query score is computed by first getting M sampling points from $U_i(t)$. The query results of the range query for each of the M points are then grouped according to their query answers. Equation 7 is then simply equal to the fraction of a total of M points that share the same set of objects in their query answers. Due to the limitation of space, we omit the algorithm details.

5.3 Managing Answer Quality

The answer quality metrics allow a user to trade-off privacy for a potentially better answer quality. In particular, the query score depends on the size of the uncertainty region – a larger uncertainty region potentially yields more distinct answers and lower query scores. Therefore, a low query score indicates that the user may reduce the size of her uncertainty region and resubmit the query.

However, reducing uncertainty region size may not improve the data score, since it depends on the uncertainty of the cloaked location information of other users that cannot be controlled by the query issuer herself. To see whether the data score is improved as a result of shrinking uncertainty region, the server can use the same cloaked location provided by the user and re-evaluate the query with a smaller uncertainty region. The server then suggests to the user to reduce her uncertainty region only if there is an improvement of the data score. Notice that as the query results are obtained by sampling over the uncertainty region, rerunning the query with a smaller uncertainty region means reusing the results of a subset of sampling points over the uncertainty region. Hence the server may be able to compute the new query incrementally.

6 Experimental Results

We have performed an extensive simulation study on the behavior of location cloaking. Here we present the simulation model, followed by experimental results.

6.1 Simulation Model

Param	Default	Meaning
City Simulator parameters		
λ_u	5,000	Location update rate (sec^{-1})
T_{start}	0.15	Start threshold
T_{fill}	0.09	Fill threshold
T_{empty}	0.5	Empty threshold
N_{obj}	100	# of moving objects
N_{relax}	2000	Max samples skipped before recording
Location cloaking parameters		
r	150	Radius of query
$U_i(t).r$	20	Radius of uncertainty region
M	49	Sampling size

Table 1. Parameters and baseline values.

Our experiments are based upon data generated by the City Simulator 2.0 [12] developed independently at IBM. The City Simulator simulates the realistic motion of N_{obj} people moving in a city. The input to the simulator is a map of a city. We used the sample map provided with the simulator that models a city of size 840×1260 square units, with 71 buildings, 48 roads, six road intersections and one park. Each building is three-dimensional and contains a number of floors. The simulator models the movement of objects within the buildings and on the roads and park. To generate reasonable movement and occupation of buildings, the simulator keeps track of two conditions based on parameters T_{fill} and T_{empty}. The simulator ensures that the fraction of people at the ground level lies between T_{fill} and T_{empty}.

Each object reports its location to the server at an average rate of λ_u. Before recording the simulation results, the simulator enters a warm-up phase, where at most N_{relax} samples for each object are generated, or at least T_{start} of the population are at the ground level of buildings. Next, the simulator records the location updates of each object in a trace file, which contains the timestamp of the update and

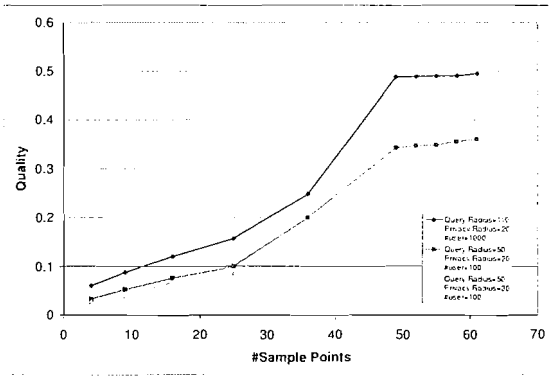

Figure 6. Quality vs. M

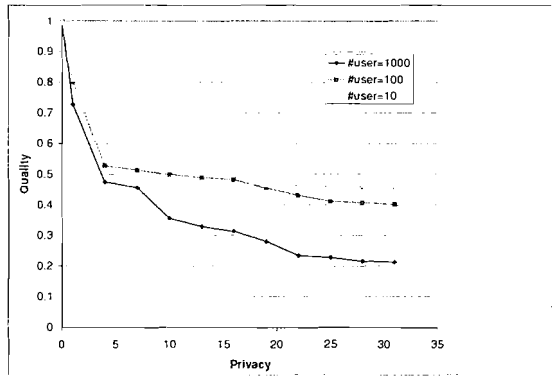

Figure 8. Quality vs. Privacy

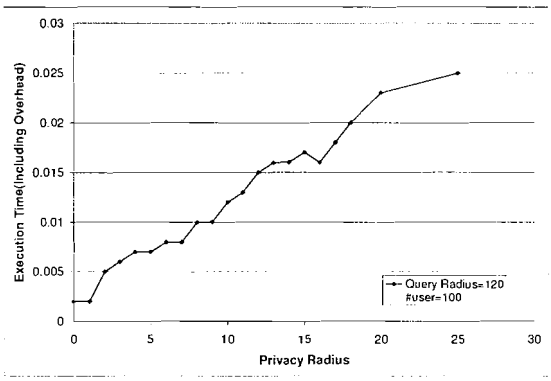

Figure 7. Execution time vs. Privacy

the spatial coordinates of the object at that time. The trace file serves as the data source for our experiments.

An IMRQ is generated by randomly choosing a user as the query issuer. The IMRQ has a range of radius r. Each user has an uncertainty region of radius $U_i(t).r$. A sampling size of M is used to implement the IMRQ evaluation algorithm.

The City Simulator is implemented in Java and runs under Windows XP. The simulation and cloaking agent program is written in C++, and the testbed is run on a UNIX server. Each data point is the average value over 200 location update cycles. We use the radius of uncertainty region as a measure of the location privacy of user – a larger radius implies a higher degree of privacy. Since we are interested in the interaction between privacy and service quality, our experiments use the IMRQ's query score as the primary metric of quality, the value of which can be adjusted by changing the resolution of the cloaked location. Table I illustrates the parameters of the simulation model.

6.2 Results

Quality and Performance. We first decide experimentally the number of sampling points for the uncertainty region, M, that gives us the highest quality with the lowest evaluation cost. Figure 6 shows the results for some combinations of privacy value, query size and number of users. When more sampling points are used, the quality increases (due to the increase in value of $p(R_{i,k})$ in Equations 7,9). The rate of increase drops when M is larger than 49. We therefore choose M to be 49 in other experiments.

In Figure 7 we see that an increase in privacy value lengthens the execution time of IMRQ. With a higher privacy (or uncertainty region of the query issuer), the ranges of sub-queries cover a larger area. Thus more objects are involved in computation, resulting in a higher execution time. We remark that an IMRQ needs little time to complete in our experiments; for example, it takes only 25ms for an IMRQ with a privacy radius of 25 units.

Quality and Privacy. We investigate the effect of location privacy on query score of the IMRQ. Figure 8 shows the result for different number of users. The quality is 1 (highest) when there is no privacy at all. As privacy (i.e., uncertainty region area) increases, the query score drops. This is because the larger uncertainty region increases the number of distinct query answers, thereby lowering the query score.

An interesting observation is that the query score does not drop linearly. This is due to the fact that the data distribution is not uniform. When an object enters a building, it can spend some time traveling around different floors of the building before going out. As a result, many moving objects are clustered in a fixed area (buildings) rather than being scattered on roads. As explained before, an increase in uncertainty region of the query issuer creates more distinct answer sets. When her uncertainty region starts to overlap a densely-populated region (i.e., a building), a slight expansion of her uncertainty region can generate many different distinct answer sets, due to the inclusion of many location data during this expansion. Thus we can see a sharper drop at some regions of the curve. On the contrary, when the un-

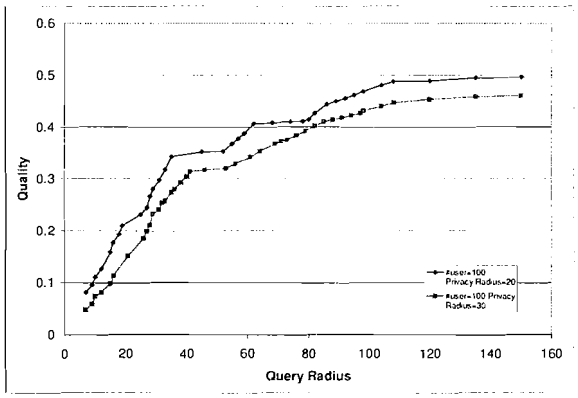

Figure 9. Quality vs. Query Radius

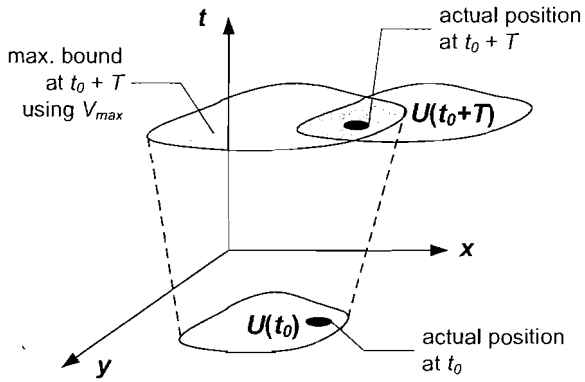

Figure 10. Increasing linkability with the last location record.

certainty region starts to cover the road, the drop is much slower because the population density on the roads is lower.

We also observe the difference in quality when the number of users varies. In general, for the same privacy value, a larger population produces a lower score, since more distinct answer sets are produced. The reason why the quality for $N_{obj} = 100$ is slightly better than $N_{obj} = 10$ when privacy is less than 20 is again due to non-uniform distribution of location data. At $N_{obj} = 10$, the query issuer chosen is located in a denser area than the case for $N_{obj} = 100$. Thus an increase in privacy value has a stronger effect on the quality when $N_{obj} = 10$ than when $N_{obj} = 100$.

We can conclude that the query score is sensitive to the density of the region covered by the cloaked location. If the region is highly dense, a slight increase in uncertainty region can reduce the quality significantly. This observation can be useful to the cloaking agent. For example, it may advise the user not to further reduce the spatial resolution of her location if she is in a crowded area.

The quality continues to decrease (slowly) when the uncertainty radius further increases. The dropping rate is much slower because the uncertainty region covers most of the objects, and so there is not much difference in the answer sets. The quality drop is mostly due to the reduction of the pdf at each point in the uncertainty region, and in turn the value of Equation 7. Due to space limitation we do not show the detailed results here.

Quality and Query Size. Next, we study the effect of query size on answer quality. Figure 9 illustrates the results: the answer quality increases with query size. With a fixed privacy value (uncertainty radius), a continuous increase in the query size will not create many distinct answer sets. When the query range has a very large radius (160) compared with the uncertainty radius (20), the query ranges created will render many similar answers, since the difference in the queries at different points in the uncertainty region is relatively small. At a larger radius (30), the relative difference between the uncertainty size and privacy is smaller than when the radius is 20, and thus the quality is lower too.

7 Protecting Linkability Against Trajectory Tracing

Before we can claim that cloaking safeguards location privacy, there is one final puzzle left to be solved – preventing the use of trajectories for inferring locations. Suppose the service provider saves all the cloaked locations it received. Also assume the maximum speed of the movement of the user is known, which can be obtained through the movement history, the vehicle owned by the user, etc. We now show that it is possible for the service provider (or attacker) to increase the linkability of a user's identity with her locations, even when it has been cloaked by the user.

Specifically, let the maximum speed of a certain user be V_{max}. Assume the user sent her last cloaked location at time t_0, i.e., $U(t_0)$, and then again after T time units, i.e., $U(t_0 + T)$. Using V_{max}, it is possible to derive the bound enclosing the user's location at time t_0+T (called *maximum bound*), as shown in Figure 10. Even if the user says she is located somewhere in $U(t_0 + T)$, her possible location is actually limited within the overlapping region between $U(t_0 + T)$ and the maximum bound, which is smaller than $U(t_0 + T)$. The linkability between the user and the locations is thus higher than she expected. Notice that this is an accumulative effect, since the service provider can derive a smaller bound based on the overlapping region. We propose two techniques, called **patching** and **delaying**, in order to solve this important problem.

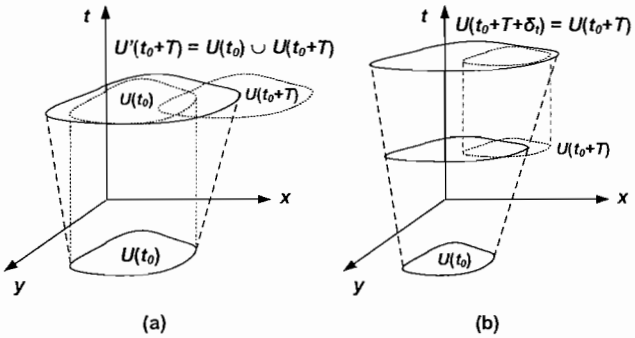

Figure 11. Linkability protection techniques by (a) patching, and (b) delaying.

7.1 Patching and Delaying

The first idea of preventing linkability from being increased is to combine the cloaked locations released in the past with the current cloaked location before it is sent. We call this technique *patching*. Figure 11(a) illustrates this concept. At time $t_0 + T$, in place of $U(t_0 + T)$, the region $U'(t_0 + T) = U(t_0) \cup U(t_0 + T)$ is sent. The increase in linkability due to trajectory tracing, or "loss" of uncertainty in $U(t_0 + T)$, is thus "compensated" by the inclusion of $U(t_0)$, which is assured to be within the maximum bound. Essentially, the spatial accuracy of the location is further relaxed. Notice that this may cause a degradation in query score due to the increase in uncertainty.

Another technique is based on relaxing the timing requirement, which we termed "*delaying*". The idea is to suspend the request until the cloaked location fits into the maximum bound. As shown in Figure 11(b), $U(t_0 + T)$ is not sent until after δ_t more time units, when $U(t_0 + T)$ is guaranteed to be within the maximum bound. The advantage of this scheme over patching is that the extent of the cloaked location remains unchanged and so the query score is not affected. However, the response time of the query can be increased due to the delay introduced, which can be an important Quality-of-Service parameter in time-critical applications.

8 Related Works

The idea of cloaking location information has been recently proposed by Gruteser et al.[7] for anonymous applications. In their model, each tuple (x, y, t) (i.e., location (x, y) at time t) is transformed to $([x_1, x_2], [y_1, y_2], [t_1, t_2])$ where $([x_1, x_2]. [y_1. y_2])$ is the rectangular area within which (x, y) is found, between the time interval $[t_1, t_2]$. To measure the degree of privacy introduced by cloaking, they propose a metric called k-anonymity, which measures between time interval $[t_1, t_2]$ the number of users, k, at the same spatial vicinity $([x_1, x_2], [y_1, y_2])$.

Another work that uses the k-anonymity metric is found in [2], where pseudonymous applications are studied. The authors use a middleware to rename pseudonyms, so that a user's identity cannot be traced. Moreover, this renaming is done while there are at least k users in the same zone at the same time period.

The k-anonymity metric has several problems. First, the scheme may not be used if there are fewer than k users in the system. Secondly, even if there are more than k users, they may span in a large area over an extended time period, in which case the cloaked location can be very large and cause a severe degradation of service quality. Thirdly, algorithms using this metric assume a trusted middleware which collects information from all users. It may present a performance bottleneck and face the risk of being compromised. It is also not clear how k-anonymity can be applied to non-anonymous applications, since it measures the anonymity of a user, while in non-anonymous application the identity of the user is already known. We suggest the level of privacy of cloaked location be measured by the the uncertainty region size and the entropy of uncertainty pdf, independent of the number of users inside the uncertainty region.

As far as we know, few papers study location privacy in non-anonymous applications. A recent paper by Gruteser et al. [8] proposes the idea of classifying a map into sensitive and non-sensitive areas. Further, every k sensitive areas are clustered into a partition. When a user enters a partition, her location updates are not released until she left the partition, provided that she had not entered any sensitive area while she was inside the partition. In this scheme, if the k sensitive areas are close to each other, it is still easy to guess that the user has entered one of the sensitive areas. In addition, there is no guarantee that there are enough sensitive areas to be clustered. Moreover, service quality is not considered, which can be seriously affected due to delay and omission of location information.

To our best knowledge, there is no previous work on relating the effect of location cloaking with service quality. We proposed in the position paper [5] a framework to balance the uncertainty injected to a location and quality of service. Here we study this idea in more detail, and present a solution for supporting IMRQ, a typical example of non-anonymous queries.

Another idea for querying private data is to use encrypted databases. Recently, Hore et al. [11] discussed a privacy-preserving index for querying range queries over encrypted data. To the best of our knowledge, these techniques only work for specific query operators. Also, the feasibility of those schemes depend on the strength of encryption. Our method does not need encryption and can be easily extended

to work with other queries.

In [4], the idea of using an uncertainty model to capture the imprecision of moving objects (due to the measurement and sampling error) is proposed. That model is a generalized version of the one presented here, where the uncertainty can change with time and the pdf within the uncertainty region can be non-uniform. That paper also presented algorithms for probabilistic nearest-neighbor queries over different object movement models. In [3], we studied other types of probabilistic queries, such as range queries and aggregate queries, and also defined notions of answer quality for them. The main difference between probabilistic queries and imprecise queries is that the information about the query issuer in probabilistic queries is exact, which may not be the case for imprecise queries. For instance, in the imprecise moving range query model, the query issuer's location is uncertain rather than exact. This calls for new evaluation algorithms and quality notions for imprecise queries.

9 Conclusions

Location privacy is an important and emerging topic. To allow a user more flexibility in controlling her privacy, the idea of injecting uncertainty to sanitize location information has been proposed recently. However, those schemes did not consider the quality and accuracy of services provided, and it was not clear how the cloaked information can be queried. We suggested a framework to connect privacy, information cloaking and service quality. We proposed imprecise queries, which hide the identity of the query issuer and enable evaluation of cloaked information. We studied an evaluation algorithm and quality metrics of moving range queries, and showed how they can be implemented conveniently using spatial-database technologies. We performed an extensive simulation to investigate behavior of the proposed scheme. We also presented techniques to protect linkability of cloaked information against trajectory tracing.

There are interesting avenues for future work. We would like to build a software system for the cloaking agent. We want to examine how our proposed metric for location cloaking can be applicable to anonymous and pseudonymous applications. We will also investigate other kinds of imprecise queries such as nearest-neighbor and average queries.

References

[1] P. Ashley, S. Hada, G. Karjoth, C. Powers, and M. Schunter. Enterprise privacy authorization language 1.2 (EPAL 1.2). In *W3C Member Submission*, Nov. 2003.

[2] A. R. Beresford and F. Stajano. Location Privacy in Pervasive Computing. *IEEE Pervasive Computing*, 2(1):46–55, 2003.

[3] R. Cheng, D. Kalashnikov, and S. Prabhakar. Evaluating probabilistic queries over imprecise data. In *Proc. ACM SIGMOD*, 2003.

[4] R. Cheng, D. V. Kalashnikov, and S. Prabhakar. Querying imprecise data in moving object environments. *IEEE TKDE*, 16(9), Sept. 2004.

[5] R. Cheng and S. Prabhakar. Using uncertainty to provide privacy-preserving and high-quality location-based services. In *Workshop on Location Systems Privacy and Control, MobileHCI 04*, Sept. 2004.

[6] L. Cranor, M. Langheinrich, M. Marchiori, M. Pressler-Marshall, and J. Reagle. The platform for privacy preferences 1.0 (P3P 1.0) specifi cation. In *W3C Member Submission*, Apr. 2002.

[7] M. Gruteser and D. Grunwald. Anonymous Usage of Location-Based Services Through Spatial and Temporal Cloaking. In *Proc. 1st Intl. Conf. on Mobile Systems, Applications, and Services*, 2003.

[8] M. Gruteser and X. Liu. Protecting privacy in continuous location-tracking applications. *IEEE Security and Privacy*, 2(2), 2004.

[9] U. Hengartner and P. Steenkiste. Access control to information in pervasive computing environments. In *Proc. 9th USENIX Workshop on HotOS*, 2003.

[10] U. Hengartner and P. Steenkiste. Protecting Access to People Location Information. In *Proc. 1st Intl. Conf. on Security in Pervasive Computing*. 2003.

[11] B. Hore, S. Mehrotra, and G. Tsudik. A privacy-preserving index for range queries. In *Proc. VLDB*, 2004.

[12] J. Kaufman, J. Myllymaki, and J. Jackson. IBM City Simulator Spatial Data Generator 2.0.

[13] I. Lazaridis and S. Mehrotra. Approximate selection queries over imprecise data. In *ICDE*, 2004.

[14] I. Lazaridis, K. Porkaew, and S. Mehrotra. Dynamic queries over mobile objects. In *Intl. Conf. EDBT*, 2002.

[15] M. Mokbel, X. Xiong, and W. Aref. SINA: Scalable incremental processing of continuous queries in spatio-temporal databases. In *Proc. ACM SIGMOD*, 2004.

[16] J. Ni and C. V. Ravishankar. Probabilistic spatial database operations. In *Proc. SSTD*, 2003.

[17] A. Pfi tzmann and M. Hansen. Anonymity, unobservability, psuedonymity, and identity management - a proposal for terminology. Sept. 2004.

[18] P. Rigaux, M. Scholl, and A. Voisard. *Spatial Databases: with application to GIS*. Morgan Kaufmann Publishers, 2002.

[19] T. Robinson. Location is everything. *Internet week online*, September 12, 2000.

[20] E. Snekkenes. Concepts for personal location privacy policies. In *Proceedings of the 3rd ACM conference on Electronic Commerce*, pages 48–57. ACM Press, 2001.

[21] J. Warrior, E. McHenry, and K. McGee. They know where you are. *Spectrum*, 40(7):20– 25, July 2003.