Efficient Evaluation of Imprecise Location-Dependent Queries

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

Abstract

In location-based services, it is common for a user to issue a query based on his/her current position. One such example is "find the available cabs within two miles of my current location". Very often, the query issuers' locations are imprecise due to measurement error, sampling error, or message delay. They may also want to protect their privacy by providing a less precise location. In this paper, we study the efficiency of queries that return probabilistic guarantees for location data with uncertainty. We classify this query into two types, based on whether the data (1) has no uncertainty (e.g., shops and restaurants), or (2) has a controlled degree of uncertainty (e.g., moving vehicles). Based on this classification, we develop three methods to improve the computational and I/O performance. The first method expands the query range based on the query issuer's uncertainty. The second idea exchanges the roles of query and data. The third technique exploits the fact that users may only be interested in answers with probabilities higher than some threshold. Experimental simulation over a realistic dataset reveals that our approaches improve the query performance significantly.

1 Introduction

In recent years, positioning technologies like the Global Positioning Systems (GPS), GSM, RF-ID and the Wireless LAN have undergone rapid development [21]. These 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 system mandated by the U.S. (correspondingly E-112 in Europe), which requires cell phone companies to provide an accurate location (i.e., within a few hundred feet) of a cell phone user who calls for emergency help [21]. Other LBS applications include downloading driving directions to a gas station, receiving an alarm when a military adversary has crossed the border, retrieving the current locations of family members, and displaying the user's location on the map. All these applications require extensive use of location data [13].

An important issue concerning the LBS is the *uncertainty* of location data. In particular, location data obtained from physical devices are inherently imprecise due to measurement error, sampling error and network latency [17, 7, 4]. Some recent works (e.g., [1, 6, 9]) have suggested that location privacy can be protected by injecting a controlled degree of spatial uncertainty into location data, so that the actual location is hidden. In many situations, therefore, it is often impossible for the query processing server to get an accurate location value. It is thus reasonable to use a location uncertainty model to describe the imprecision of the data values, and evaluate the location uncertainty in order to provide probabilistic guarantees over the validity of the query answers. A common model for characterizing location uncertainty of an object is a closed region together with a probability distribution of the object in the region [17, 7]).Some previous work, such as [17, 7, 4], used this model to compute probabilities of each location object for satisfying a query, including the range and nearest-neighbor queries. The probability values provide confidence guarantees about the query answer, and allow quality of service metrics to be defined [3, 6].

In this paper, we study the effect of uncertainty on *location-dependent queries*, which takes into account the location of the user who issues the query (called "query issuer") in order to decide the query result [13, 8, 12]. For example, the user may want to know the available cabs within two miles of his/her current location. In addition to the uncertainty of the data being queried, the imprecision of the query issuer's location further affects the validity of the query answer. Our goal is to quantify the query answer validity by efficiently computing the *qualification probability* of an object for satisfying this type of query i.e., the probability that the object can satisfy this query. To our best knowledge, this has not been studied before.

Specifically, we study the imprecise version of the *location-dependent range query*. Based on the location information of the query issuer, together with a range region centered at the query issuer's location, the query returns the identities of all objects that fall within the region. We propose efficient algorithms to compute qualification probabilities. Moreover, we develop a set of novel filtering techniques to determine quickly the region which contains objects that may satisfy the query. In particular, we classify a query according to whether the location data being accessed is (1) *precise* (e.g., locations of gas stations, schools etc.), or (2) *uncertain* (e.g., locations of moving objects). Based on this classification, we develop three fundamental concepts to enhance the evaluation of the qualification probabilities:

- **Query expansion:** Incorporate the uncertainty information of the query issuer's location into the query range by using computational geometry techniques, so that query evaluation can take advantage of traditional query processing methods.
- **Query-Data duality:** By interchanging the role of the query issuer and that of the location object, simplify the qualification probability formula and thus save the query evaluation cost.
- **Use of the probability threshold constraint:** By exploiting the assumption that users are only concerned about answers with high qualification probabilities, special data constructs are pre-computed for uncertain objects in order to facilitate pruning.

The aforementioned methods can be executed efficiently. They can also deal with *any* type of probability distribution about the object's location. We perform detailed experimental evaluations to examine our approaches.

The rest of this paper is organized as follows. In Section 2, we present the related work. Section 3 gives a formal definition of the problems. In Section 4 we discuss how to improve the performance of imprecise location-dependent queries. Section 5 then presents algorithms that exploit the probability threshold constraint. In Section 6 we report our experimental results. Section 7 concludes the paper.

2 Related Works

Location-Dependent Queries have been a subject of research interest in these few years. In [13], a survey of research problems related to location-dependent information services has been presented. The problem of managing location queries in a distributed manner has been studied in [8], where the MobiEyes system is developed to answer queries efficiently. In [12], the authors define location-dependent query operators so that more complex queries can be constructed. They also study how mobile agents can be used to support distributed query processing.

Probabilistic Queries are queries that evaluate data uncertainty and provide probabilistic guarantees. For location-based services, the evaluation of probabilistic range queries have been studied in [17, 7]. Efficient indexing methods for these queries in one-dimensional and multi-dimensional space are studied in [5] and [19]. In [4], efficient computation and indexing algorithms have been proposed for evaluating probabilistic nearest neighbor queries. The evaluation and quality of different probabilistic queries in a sensor database is studied in [3]. Notice that the location of the query issuer considered by a probabilistic query is usually precise. For imprecise location-dependent queries, the location of the query issuer can also be uncertain.

For **Imprecise Location-Dependent Queries**, Song et al. [18] study the evaluation of a a continuous nearest-neighbor query for retrieving the nearest neighbors for all points on a line segment. Their algorithm is further improved in [20]. In [11], the range nearest-neighbor query is proposed, which retrieves the nearest neighbor for every point in a multi-dimensional rectangular range. In these works, although the query issuer's location is imprecise, the data being queried has no uncertainty. Furthermore, they do not consider the problem of computing qualification probabilities. Recently, [15] considers both query and data uncertainty for nearest-neighbor queries, but it does not compute qualification probabilities.

To the best of our knowledge, none of the previous work address the issue of providing probability guarantees for imprecise location-dependent queries. In [6], we studied the trade-off of location privacy, service quality, and the uncertainty of location-dependent range queries. A service quality metric based on the objects' qualification probabilities is proposed. In this paper we address the efficiency issues of evaluating this type of query.

3 Problem Definition

In this section we describe the location uncertainty model, and definitions of queries studied in this paper. We also investigate preliminary solutions for these queries.

3.1 Probabilistic Uncertainty Model

To capture location uncertainty, a data scheme known as *location uncertainty model* was proposed in [17] and [7]. This model assumes that each location data item can be represented by a region of possible values and their distributions. Formally, given an object O_i (which we called *uncertain object*), the *probabilistic uncertainty* of the two-dimensional location of O_i consists of two components:

Definition 1 *The* **uncertainty region** *of O_i, denoted by U_i, is a closed region where O_i is located.*

Definition 2 *The* **uncertainty probability density function (pdf)** *of object O_i, denoted by $f_i(x, y)$, is a pdf of O_i's location, that has a value of 0 outside U_i.*

Since $f_i(x, y)$ is a pdf, it has the property that $\int_{U_i} f_i(x, y)dxdy = 1$. The formula for the uncertain pdf is application-specific. Wolfson et al. propose that the object location follows the Gaussian distribution inside the uncertainty region [17]. An important case of uncertainty pdf is a uniform distribution [7], that is, $f_i(x, y) = \frac{1}{U_i}$; essentially, this implies a "worst-case" scenario where we have no knowledge of which point in the uncertainty region possesses a higher probability. Our solutions are applicable to any form of uncertainty pdf.

Without loss of generality, we denote O_0 as the identity of the query issuer, and $O_1, \ldots, O_n$ as the identities of the objects being queried. We also assume the uncertainty region is an axis-parallel rectangle. For objects with no uncertainty (called *point objects*), we denote them $S_1, \ldots, S_m$. In particular, object S_i's location is exactly a point (x_i, y_i) (e.g., a non-moving user or a building).

3.2 Imprecise Location-Dependent Range Queries

In this paper, we focus on a type of snapshot, location-dependent query – the location-dependent range query. Given an axis-parallel rectangle $R(x, y)$ with center (x, y), half-width w and half-height h, and the location of O_0 as its center, two types of queries can be defined:

Definition 3 *An Imprecise Location-Dependent Range Query over Point Objects (IPQ) returns a set of tuples $\{(S_i, p_i)|i \in [1, m]\}$ where $p_i > 0$, called qualification probability, is the non-zero probability that S_i's location, (x_i, y_i), is inside $R(x, y)$, with $(x, y) \in U_0$.*

Definition 4 *An Imprecise Location-Dependent Range Query over Uncertain Objects (IUQ) returns a set of tuples $\{(O_i, p_i)|i \in [1, n]\}$ where $p_i > 0$, called qualification probability, is the non-zero probability that O_i is located within $R(x, y)$, with $(x, y) \in U_0$.*

Figure 1 illustrates the IUQ and IPQ. For convenience, we may use R to represent $R(x, y)$.

Sometimes users are more concerned about answers with sufficiently high probability values. In fact, it is useful to define a "probability threshold constraint", which restricts queries to return answers with probability values higher than a certain pre-defined value. We call this parameter the *probability threshold* (Q_p in short), which is a real value between 0 and 1. The following queries can then be defined:

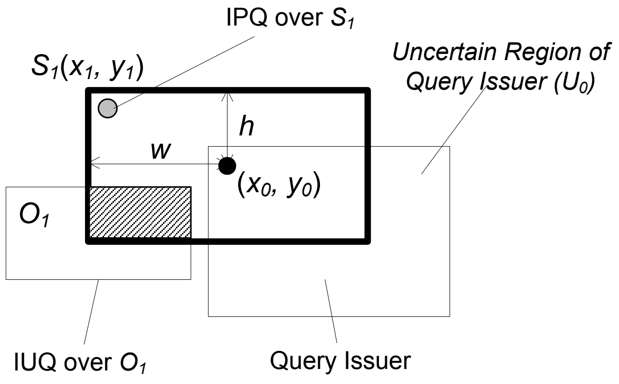

Figure 1. Evaluating IPQ and IUQ.

Definition 5 *A Constrained Imprecise Range Query over Point Objects (C-IPQ) returns a set of tuples $\{S_i|i \in [1, m]\}$ such that $p_i \geq Q_p$, where p_i is the qualification probability of S_i for satisfying the corresponding IPQ.*

Definition 6 *A Constrained Imprecise Range Query over Uncertain Objects (C-IUQ) returns a set of tuples $\{P_i|i \in [1, n]\}$ such that $p_i \geq Q_p$, where p_i is the qualification probability of O_i for satisfying the corresponding IUQ.*

Later we will show how to use the probability threshold to improve query performance. Table 1 describes the notations used in this paper.

Symbol	Meaning
O_i	An uncertain object ($i = 1, \ldots, n$)
U_i	Uncertainty region of O_i
$f_i(x, y)$	Uncertainty pdf of O_i
S_i	A point object ($i = 1, \ldots, m$)
(x_i, y_i)	Position of S_i
O_0	Query issuer (an uncertain object)
$R(x, y), w, h$	Range query with half-width w and half-height h, centered at (x, y)
p_i	Qualification probability
Q_p	Probability threshold

Table 1. Notations and their meanings.

3.3 Basic Evaluation Methods

Let us now present a basic solution for evaluating IPQ and IUQ. They form the foundation for further discussions.

For **IPQ**, the qualification probability of S_i can be obtained by conceptually examining every point $(x_0, y_0) \in U_0$, and then checking whether the location of S_i is within $R(x_0, y_0)$. Figure 1, for example, illustrates that S_1 satisfies $R(x_0, y_0)$. The final result can be obtained by integrating the uncertainty pdf of all the points (x, y) in U_0 at which S_i

satisfies $R(x, y)$. Formally, we define a boolean function, $b_i(x, y)$ (for $i = 1, \ldots, n$), as follows:

$$b_i(x,y) = \begin{cases} 1 & \text{if } S_i \text{ is inside } R(x,y) \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

The qualification probability of S_i for satisfying the IPQ is then given by

$$p_i = \int_{U_0} b_i(x,y) f_0(x,y) dxdy \quad (2)$$

For **IUQ**, we examine the probability that an uncertain object O_i satisfies the query at each point in U_0. This is given by integrating the uncertainty pdf of O_i in the overlapping area of U_i and $R(x, y)$, i.e.,

$$p_i(x,y) = \int_{U_i \cap R(x,y)} f_i(x,y) dxdy \quad (3)$$

Figure 1 shows that the probability of O_1 satisfying $R(x_0, y_0)$, given that O_0 is at point (x_0, y_0), is the integral of $f_1(x, y)$ over the shaded region. Considering the uncertainty pdf of O_0, the following gives the general formula for computing p_i.

$$p_i = \int_{U_0} p_i(x,y) f_0(x,y) dxdy \quad (4)$$

In practice, Equations 2 and 4 are costly to implement. Both equations may necessitate the use of numerical integration. For example, in order to obtain p_i in Equation 2, U_0 is first represented by a set of sampling points; for each sampling point, Equation 2 is evaluated. A large number of sampling points will be needed to produce an accurate answer. This is required even if the uncertainty pdf is as simple as a uniform distribution. Let us investigate how this situation can be improved.

4 Efficient Evaluation of Imprecise Queries

We just see that straightforward computation of probabilities for IPQ and IUQ can lead to expensive integral operations. This section illustrates how such operations can be avoided, by (1)expanding the range query, and (2) exploiting the duality between the locations of the query issuer and data being queried. We also examine indexing techniques for these queries.

4.1 Query Expansion

The first technique performs an inexpensive filtering over objects that have no chance of being included in the query answer. The main idea is to expand the query range R with the query issuer's position information. Any object that does not touch this expanded query range (called the *Minkowski Sum* [2]) can be pruned.

Lemma 1 *The qualification probability of a point object (an uncertain object) is non-zero if and only if its location (uncertainty region) lies within (overlaps) the Minkowski Sum of R and U_0.*

To explain the above lemma, let us consider the IUQ (similar arguments can be applied to IPQ). First, notice that the Minkowski Sum is defined as follows:

$$A \oplus B = \{x + y | x \in A, y \in B\}$$

where A and B are two given polygons, and x and y are points [2]. Conceptually, the Minkowski Sum is the union of all translations of A by a point y located in B. We can view the Minkowski Sum of the query range R and the uncertainty region U_0, that is, $R \oplus U_0$, as the union of all range queries, by considering all the possible positions of O_0 who resides somewhere in U_0. If the uncertainty region of any object being queried does not overlap $R \oplus U_0$, we can be assured that this object does not have any chance of satisfying any range query issued at any position in U_0. Thus we can use $R \oplus U_0$ as a query range to obtain objects that have non-zero qualification probability of satisfying the IUQ (i.e., their uncertainty regions overlap with the query range).

Figure 2 illustrates the Minkowski Sum of R and U_0, which can simply be obtained by extending U_0 by w on the left and right, and by h on the top and bottom.[1] Hence, the Minkowski Sum can be derived in a linear time. As shown in the same figure, the expanded query range allows objects with zero qualification probability (i.e., objects O_1) to be pruned immediately.

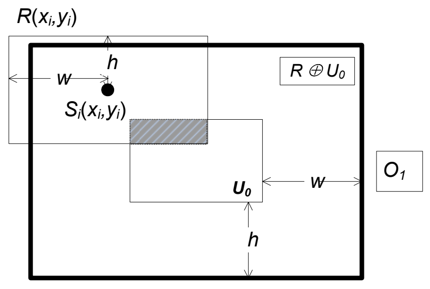

Figure 2. Illustrating the evaluation of IPQ. The thick line is the expanded query using the Minkowski Sum.

4.2 Query-Data Duality

The second method exploits the fact the role of the query issuer and the data being queried can be changed. Specif-

[1]If U_0 and R are m-sided and n-sided polygons, the Minkowski Sum is a convex polygon with at most $m + e$ edges, which requires $O(m + e)$ time to compute [2].

ically, the following observation describes this "query-data duality" property:

Lemma 2 Query-Data Duality *Given two point objects S_i and S_q (with locations (x_i, y_i) and (x_q, y_q) respectively), S_i satisfies $R(x_q, y_q)$ if and only if S_q satisfies $R(x_i, y_i)$.*

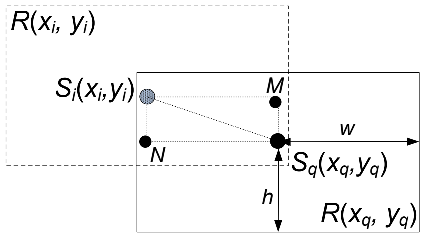

Figure 3. The Duality of Query and Data.

Proof : Construct a rectangle with vertices M, S_i, N, and S_q, as shown in Figure 3. If S_i satisfies $R(x_q, y_q)$, then $|S_iM| \le w$ and $|S_iN| \le h$. This implies $|S_qN| \le w$ and $|S_qM| \le h$. Hence S_q must satisfy the query $R(x_i, y_i)$. Conversely, if S_q satisfies $R(x_i, y_i)$, we can construct the same rectangle and prove that $|S_iM| \le w$ and $|S_iN| \le h$. Hence S_i must also satisfy $R(x_q, y_q)$. □

In other words, if a point object S_i satisfies the range query issued by S_q, then S_q must also satisfy the range query issued by S_i. This leads us to the following result.

Lemma 3 *The qualification probability p_i of a point object S_i for satisfying an IPQ can be computed by*

$$\int_{R(x_i,y_i)\cap U_0} f_0(x, y)dxdy \tag{5}$$

Proof : Consider the overlapping region of U_0 and the range query $R(x_i, y_i)$ issued by S_i, as shown in the shaded area in Figure 2. Obviously, any point $(x_e, y_e) \in U_0 \cap R(x_i, y_i)$ satisfies the query $R(x_i, y_i)$. Using Lemma 2, S_i also satisfies the range query $R(x_e, y_e)$. Moreover, S_i does not satisfy any range queries centered at points outside the overlapping region. Hence only the queries issued at points (x_e, y_e) can have S_i in their answer, and the qualification probability of S_i is simply the integration of the uncertainty pdf of O_0 in the overlapping region, i.e., $\int_{R(x_i,y_i)\cap U_0} f_0(x, y)dxdy$. □

Compared with the original formula for IPQ (i.e., Equation 2), we can see that Equation 5 is simpler to evaluate. This is because now we do not need to form a query at each point in U_0 and test whether S_i satisfies the query at that point, as in Equation 2. More importantly, if the uncertainty pdf of the query issuer is a simple function, the integration operation of Equation 2 can be eliminated. As an important example, if $f_0(x, y)$ is a uniform distribution, p_i is simply the fraction of U_0 that overlaps $R(x_i, y_i)$, i.e.,

$$p_i = \frac{\text{Area}(R(x_i, y_i) \cap U_0)}{\text{Area}(U_0)} \tag{6}$$

When the position of S_i is at different positions relative to U_0, the exact formula for calculating the overlapping region $U_0 \cap R(x_i, y_i)$ (and hence Equation 6) can also vary. In particular, the 2D space can be partitioned into nine regions, and depending on which regions that S_i is located, a different algebraic expression is needed. Interested readers are referred to our technical report. [2]

Lemma 3 can also be used to compute the qualification probability of an uncertain object O_i for satisfying the IUQ. We can conceptually treat every point $(x, y) \in U_i$ as a point object, and compute the qualification probability of each individual point (x, y) for satisfying the IPQ (termed $Q(x, y)$), with Lemma 3. The qualification probability of O_i is then simply equal to the integral of all these $Q(x, y)$ values, weighted by the uncertainty pdf of O_i at (x, y), i.e.,

$$p_i = \int_{U_i} f_i(x, y) \cdot Q(x, y)dxdy \tag{7}$$

Hence, Equation 7 provides an alternative to Equation 4 for computing IUQ. Although it is not immediately clear which method is better, we note that the performance of Equation 7 can be further improved when combined with our results about query expansion.

Lemma 4 *The qualification probability p_i of an uncertain object O_i for satisfying an IUQ can be computed by*

$$p_i = \int_{U_i\cap(R\oplus U_0)} f_i(x, y) \cdot Q(x, y)dxdy \tag{8}$$

The only difference between this equation and Equation 7 is that U_i is replaced by $U_i \cap (R \oplus U_0)$ – which potentially produces a smaller integration region and better computational performance. How is this possible? Observe that for any point $(x_t, y_t) \in U_i - (R \oplus U_0)$, $Q(x_t, y_t)$ must be zero, according to Lemma 1. Hence it is fine to focus on the portion of U_i that overlaps the expanded query region.

4.3 An Efficient I/O Solution

To improve the I/O performance of query processing, spatial data indexes such as the R-tree [10] and the grid file [16] are often used. In order to utilize these indexes for processing imprecise queries, we first construct an expanded query range online (i.e., find the Minkowski Sum of R and U_0). This expanded query range is then used to query the spatial index. All the point objects or uncertain

[2] Available at http://www.comp.polyu.edu.hk/∼csckcheng/tech/cc2006.pdf.

objects that lie completely outside the expanded range can be pruned. The qualification probabilities of the remaining objects can then be computed by using the lemmas described in Section 4.2.

5 Constrained Imprecise Queries

So far we have addressed imprecise queries that produce answers with non-zero probabilities. In this section we study how query performance can be improved by only allowing objects with qualification probabilities higher than a pre-defined threshold value to be returned. These queries, called C-IPQ and C-IUQ, are the constrained version of IPQ and IUQ, as defined in Section 3. Our main idea is to use pre-computed boxes for uncertain objects, based on their uncertainty pdf, in order to achieve better pruning effects. In particular, we employ the concept of the p-bound [5, 19], as described next.

5.1 Pruning Point Objects for C-IPQ

A p-bound of an uncertain object O_i is a function of p, where $p \in [0, 0.5]$. It is composed of four line segments, namely $l_i(p), r_i(p), t_i(p), b_i(p)$ in 2D space, as illustrated by the hatched region in Figure 4. The requirement of $l_i(p)$ is that the probability of the location of O_i on the left of $l_i(p)$ has to be exactly equal to p (as shown by the shaded area). Similarly, the probability of O_i on the right of the line $r_i(p)$ is exactly equal to p. The remaining line segments ($t_i(p)$ and $b_i(p)$) are defined analogously. Notice that based on this definition, the boundary of U_i can be represented by $l_i(0), r_i(0), t_i(0)$ and, $b_i(0)$. We will show that if p-bounds have been pre-computed and stored for any value of p, better pruning power can be achieved.

In practice, it is not possible to pre-compute a p-bound for each value of p. Instead, a "U-catalog" is created for each object, which is a table of a small fixed number of tuples $\{v, v\text{-bound}\}$, where $v \in [0, 1]$ [19]. The tuples in the U-catalog can then used for query pruning. For ease of discussions, we assume that p-bound is created for each object, for any value of p. However, we will revisit the issue of U-catalog whenever appropriate.

Next, we define the *p-expanded-query as follows*:

Definition 7 *A p-expanded-query is a rectangular region such that any point object lying outside it has a qualification probability of less than p for satisfying the IPQ.*

Figure 5 illustrates a p-expanded-query, where by definition S_j has a probability of less than p of satisfying the IPQ. Notice that the Minkowski Sum of R and U_0 is equivalent to a 0-expanded-query, outside which no object has a qualification probability of more than zero. Also for any

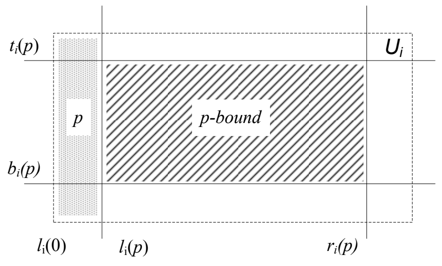

Figure 4. Illustrating the p-bound of O_i.

$p_0 > 0$, the p_0-expanded-query is always enclosed by the 0-expanded-query. In general, $p_j \geq p_k$ if and only if the p_j-expanded-query is enclosed by the p_k-expanded-query.

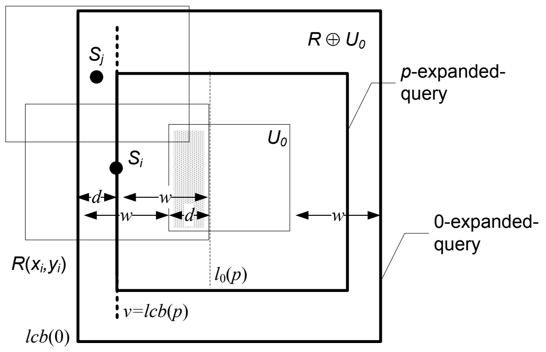

Figure 5. Pruning S_i for C-IPQ.

Let us use $lcb(p)$ to denote the left side of a p-expanded-query. The following lemma states an important property of $lcb(p)$.

Lemma 5 *$lcb(p)$ is d units from the right of $lcb(0)$, where d is the distance between the two lines $l_0(0)$ and $l_0(p)$ of U_0.*

Proof : Consider a point S_i, which is w units on the left of $l_0(p)$ (Figure 5). If an IPQ is issued by O_0, the qualification probability p_i of S_i must be less than or equal to p. This is because the integration of $f_0(x, y)$ over the (shaded) common region of $R(x_i, y_i)$ and U_0 cannot be larger than p, and according to Lemma 3 this is exactly equal to p_i. Next, consider the vertical line v intersecting S_i. We claim that line v is $lcb(p)$. This is because for any object S_j lying on v or on the left of v, either (1) $R_i(x, y)$ does not touch U_0 at all, or (2) $R_j(x, y) \cap U_0$ is a portion of the shaded region. Using Lemma 3,

$$p_j = \int_{R_j(x,y)\cap U_0} f_0(x, y)dxdy \quad (9)$$

$$\leq \int_{\text{shaded region}} f_0(x, y)dxdy \quad (10)$$

$$= p \quad (11)$$

Therefore, any point object on the left of the line v must have a qualification probability less than or equal to v. Hence $v = lcb(p)$. Moreover, as shown in Figure 5, $lcb(p)$ is d units from $lcb(0)$. Hence the lemma holds. □

By using Lemma 5, we can construct a p-expanded-query easily. As shown in Figure 5, $lcb(p)$ is simply the vertical line with a distance of w units from $l_0(p)$. The other sides of the p-expanded-query can be obtained analogously.

The p-expanded-query can be used to prune point objects for C-IPQ. Specifically, we first construct a Q_p-expanded query (where Q_p is the probability threshold of C-IPQ). Then, a point object S_i can be pruned if it lies outside the Q_p-expanded query, since it is guaranteed to have a qualification probability less than Q_p. We note that this pruning is often better than using the Minkowski Sum as described in Section 4.1, since the p-expanded-query usually has a smaller range.

If the U-catalog has to be used to find the p-expanded-query, we can use the maximum value of M in the U-catalog such that $M \leq Q_p$. The M-expanded-query, which encloses Q_p-expanded-query, can then be used for pruning: any object pruned by the M-expanded-query must also be pruned by the Q_p-expanded-query.

5.2 Pruning Uncertain Objects for C-IUQ

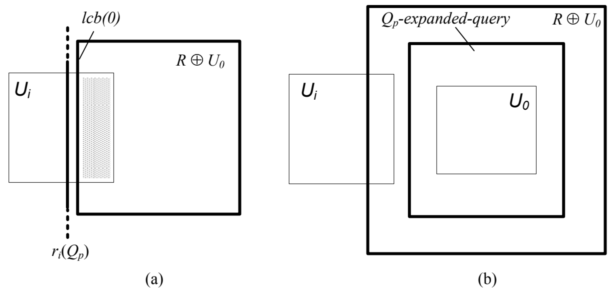

Figure 6. Pruning O_i for C-IUQ (a) using the $r_i(Q_p)$ bound of O_i; (b) using the Q_p-expanded-query.

We now investigate how the concept of p-bound and p-expanded-query can be used to facilitate pruning of an uncertain object O_i for C-IUQ. Three pruning strategies are possible.

- **Strategy 1: Use the p-bound of O_i.** Observe that we can prune O_i if the common region of U_i and $R \oplus U_0$ is on the right of $r_i(Q_p)$. As shown in Figure 6(a), we can be sure that $\int_{U_i \cap (R \oplus U_0)} f_i(x,y)dxdy \leq Q_p$. From Lemma 4, we have

$$\begin{aligned} p_i &= \int_{U_i \cap (R \oplus U_0)} f_i(x,y) \cdot Q(x,y)dxdy \quad (12) \\ &\leq \int_{U_i \cap (R \oplus U_0)} f_i(x,y)dxdy \quad (13) \\ &\leq Q_p \quad (14) \end{aligned}$$

Therefore, O_i can be removed from the result. If $r_i(Q_p)$ cannot be found, then we find the maximum value M in the U-catalog of O_i such that $M \leq Q_p$, and then use $r_i(M)$ instead of $r_i(Q_p)$. Notice that $r_i(M)$ is on the right side of $r_i(Q_p)$, so it may be possible that the shaded region crosses $r_i(M)$ but not $r_i(Q_p)$. The same idea can be applied to other dimensions. For example, if $U_i \cap (R \oplus U_0)$ is at the lower part of $b_i(Q_p)$, U_i can be pruned.

- **Strategy 2: Use the p-expanded-query.** O_i can be pruned if U_i is completely outside the Q_p-expanded-query (Figure 6(b)). Recall from Definition 7 that any point object outside the p-expanded-query cannot have a qualification probability higher than p. Therefore, by Lemma 4,

$$\begin{aligned} p_i &= \int_{U_i \cap (R \oplus U_0)} f_i(x,y) \cdot Q(x,y)dxdy \quad (15) \\ &\leq Q_p \int_{U_i \cap (R \oplus U_0)} f_i(x,y)dxdy \quad (16) \\ &\leq Q_p \quad (17) \end{aligned}$$

If U-catalog is to be used, the M-expanded-query should be chosen, where M is the maximum value in the catalog such that $M \leq Q_p$. Notice that this query range may be larger than that of Q_p-expanded-query, resulting in less efficient pruning.

Notice that both methods create more pruning opportunities than the query expansion technique described in Section 4.1, since O_i can be pruned even if they overlap with the Minkowski Sum of R and U_0.

- **Strategy 3: Use both the p-bound and the p-expanded query.** The third pruning strategy can be used when both Strategy 1 and Strategy 2 do not work. An example scenario is shown in Figure 7, where $R \oplus U_0$ crosses the $r_i(Q_p)$ line, and U_i crosses the $lcb_(Q_p)$ line. Hence, both pruning methods cannot be applied. We now show that it is still possible for O_i to be pruned. Let d_{min} be the minimum value in the U-catalog of O_i such that $d_{min} \geq Q_p$ and $R \oplus U_0$ is on the right of $r_i(d_{min})$. Also let q_{min} be the minimum value in the U-catalog of O_0 such that $q_{min} \geq Q_p$ and U_i is on the outside of the q_{min}-expanded query. By using Lemma 4, we have

$$\begin{aligned} p_i &= \int_{U_i \cap (R \oplus U_0)} f_i(x,y) \cdot Q(x,y)dxdy \quad (18) \\ &\leq q_{min} \int_{U_i \cap (R \oplus U_0)} f_i(x,y)dxdy \quad (19) \end{aligned}$$

$$\leq \quad q_{min} \cdot d_{min} \tag{20}$$

Therefore, if the product of q_{min} and d_{min} is smaller than Q_p, O_i can be pruned.

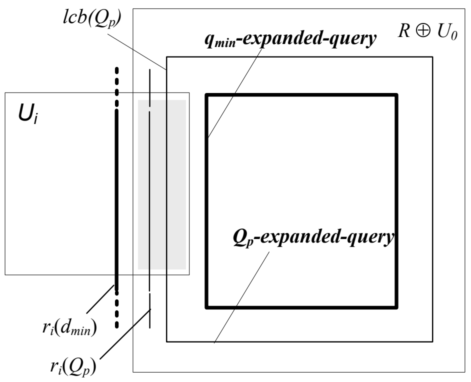

Figure 7. Pruning O_i for C-IUQ using both bounding box information of O_i and the expanded query.

The three strategies just described involve some simple condition testing, and the size of an object's U-catalog is usually small (for example, in our experiments, we store six probability values and their p-bounds). Hence these methods can be used to prune uncertain objects efficiently.

5.3 Efficient I/O Solutions

To improve the I/O performance of constrained imprecise queries, the steps presented for Section 4.3 can be readily used. The only difference is that the expanded query range (i.e., the Minkowski Sum of R and U_0) is replaced by the Q_p-expanded-query. The performance of query pruning is potentially better, since the Q_p-expanded-query is usually smaller than $R \oplus U_0$.

The performance of C-IUQ can be further improved by using a data structure designed for storing uncertain data. Called *Probability Threshold Index* (PTI) in [5], the U-catalog information of uncertain objects under the same parent node of the R-tree is summarized . In each intermediate node, the minimum bounding rectangle (MBR(m)) for each probability value of m in the U-catalog is stored. This MBR(m) should be tight and enclose all the m-bounds for all children in the node. For example, suppose a node X that consists of two objects, O_1 and O_2, and in their U-catalogs the 0.3-bounds are stored. The left sides of their 0.3-bounds are $l_1(0.3)$ and $l_2(0.3)$ respectively. If $l_2(0.3)$ is on the left of $l_1(0.3)$, then $l_2(0.3)$ is assigned to be the 0.3-bound for the node X.

With the aid of the PTI, we can apply the pruning techniques described in Section 5.2 in the index level. As described in Section 5.2, the U-catalog of an object can be used for pruning. We can use the same techniques with the U-catalog that resides in the intermediate node. This is because every p-bound stored in this U-catalog must enclose the p-bounds for each children. Moreover, our pruning techniques rely on the fact that the uncertainty region of the objects lie outside the p-bound. In Figure 6, for example, if $R \oplus U_0$ is on the right side of the m-bound of the U-catalog in the intermediate node, $R \oplus U_0$ must also be on the right side of $r_i(Q_p)$, assuming U_i is stored under that intermediate node. In other words, if the p-bound in the intermediate node level satisfy the pruning condition, so does its children.

6 Experimental Results

We have performed experimental evaluation on the effectiveness of our approaches. We first present our simulation model, followed by the detailed results.

6.1 Experiment Setup

We use two realistic data sets, namely *California* and *Long Beach*.[3] The California data set contains 62K points. The Long Beach data set contains 53K rectangles. The objects in both data sets occupy a 2D space of $10,000 \times 10,000$ units. We use the California data set as a point object database, and the Long Beach data as an uncertain object database. We also assume that the uncertainty pdf of any uncertain object (including the query issuer) is a uniform distribution. Each uncertain object is associated with a U-catalog, which consists of ten p-bounds for the probability values $0, 0.1, \ldots, 1$. The size of an R-tree node is 4K. Unless stated otherwise, R-tree is used for data indexing.

The datasets above are evaluated by the imprecise queries. As an example scenario, a policeman may wish to look for suspect vehicles (in the database) within some distance from his (imprecise) location. For each query tested, both the uncertainty region of the query issuer (U_0) and the range query (R) have square shapes. For convenience, we denote the size of U_0 and R as u and w. Here, the term "size" refers to the half of the length of a square. By default, $u = 250$ and $w = 500$. The center point of U_0 is uniformly distributed in the data space. We also assume the probability threshold of an imprecise query is 0 (i.e., $Q_p = 0$).

The performance metric used here is the total amount of time for executing a query (called "response time", T in short). Each data point is an average over 500 runs. All our experiments are run on a sunfire4800 server with four US-III+900MHz CPUs and 4096MB of memory. We use the R-tree provided by the Spatial Index Library version 0.44.2b [14].We also use this library to implement the PTI.

[3]Available at `http://www.census.gov/geo/www/tiger/`.

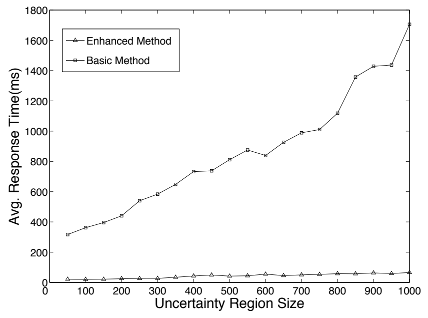

Figure 8. Basic vs. Enhanced (IUQ)

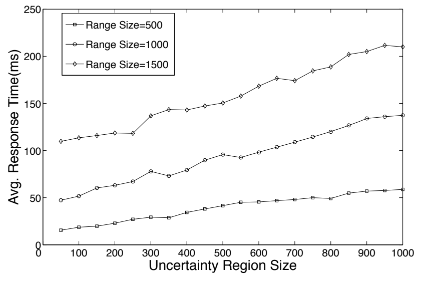

Figure 9. T vs. c (IPQ)

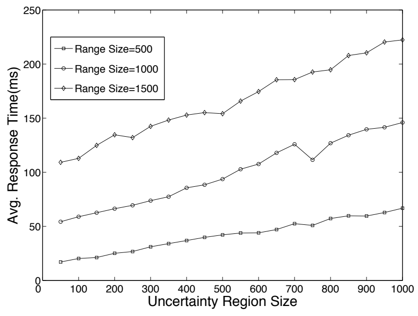

Figure 10. T vs. c (IUQ)

Our simulation is written in $j2sdk1.4.2\_11$. Table 2 summarizes the parameters used in the experiments.

Param	Default	Meaning
u	250	Size of U_0
w	500	Size of range query
$f_i(x,y)$	$1/\lvert U_i\rvert$	O_i's uncertainty pdf (uniform)
Q_p	0	Probability threshold
T	$--$	Query response time

Table 2. Parameters and baseline values.

6.2 Results

Comparison with the basic solution. We first compare the performance of the basic solution described in Section 3.3, with our other techniques. Our experiments reveal that the basic solution is much more costly. Figure 8, for example, illustrates that the basic solution for calculating qualification probability (Equation 4) performs much worse than our solution (Equation 8). Similar results can be concluded for other queries. Next, we focus on the methods developed in Sections 4 and 5.

IPQ and IUQ. Let us examine the performance of IPQ using the methods developed in Section 4. Figure 9 presents the relationship between T and u, under different values of w. We observe that T varies from 20 to 220 ms, under a wide range of values of u and w. Also, T increases with both parameter values. Recall that the expanded query range is essentially the Minkowski Sum of R and U_0. It is enlarged when either w or u increases. As a result, more candidates are captured by the expanded query and more probability computations are required. Figure 10 shows a similar behavior for IUQ.

C-IPQ and C-IUQ. Next, we compare the performance of C-IPQ, using (1) the Minkowski-Sum, and (2) the p-expanded-query, which considers the probability threshold (Q_p) of the C-IPQ. Figure 11 shows the performance under different values of Q_p. In general, the performance of p-expanded-query improves with an increasing value of Q_p. As discussed in Section 5.1, the p-expanded-query shrinks with a higher value of Q_p. Hence the number of candidates that satisfy the p-expanded-query decreases, rendering a much better performance than using the Minkowski Sum alone (e.g., three times of improvement at $Q_p = 0.6$).

In Figure 12 we investigate the performance of C-IUQ. The R-tree is used with the Minkowski Sum, while the PTI is used with the p-expanded-query. Again, we see the benefits of using p-expanded-query and PTI over methods that do not utilize probability threshold constraints, because they allow much more data pruning for all values of Q_p. For example, at $Q_p = 0.6$, the performance gain is around 60%. Notice that this gain is smaller compared to C-IPQ. The reason is that it is usually harder to prune uncertain objects, whose uncertainty regions may occupy large amount of space, than point objects.

Non-Uniform Distribution. Finally, we examine the performance of queries when the uncertainty pdf is not uniform. We choose the Gaussian distribution, which is a common distribution used in modeling location uncertainty [17, 4]. For each object, the mean of the Gaussian distribution is the center of its uncertainty region, while the variance is one-sixth of the size of its uncertainty region.

When the uncertainty pdf is non-uniform, it is more expensive to evaluate than uniform distribution. In particular, Equations 5 and 8 may not have closed-form solutions, and numerical techniques are often required. We have used the Monte-Carlo technique for evaluation, where the positions of the query issuer and uncertain objects are sampled for a number of times, and the average result is obtained. In order to achieve an accurate result, we have performed a sensitivity analysis: we need at least 200 samples for evaluating a C-IPQ, and 250 samples for C-IUQ. Figure 13 shows the result for C-IPQ. We again see that the use of p-expanded-query achieves a better performance by exploiting the probability threshold constraint.

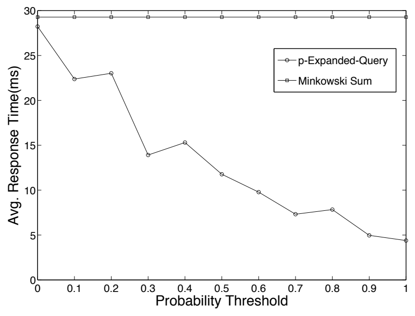

Figure 11. T vs. Q_p(C-IPQ)

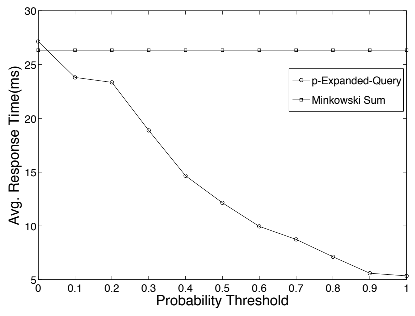

Figure 12. T vs. Q_p(C-IUQ)

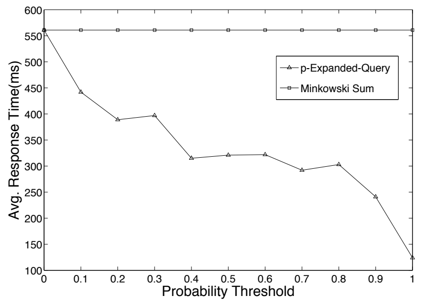

Figure 13. T vs. Q_p (C-IPQ)

7 Conclusions

Computing qualification probabilities for imprecise location-dependent queries by using their definitions directly can be expensive. In this paper, we proposed several techniques for improving the query performance. The use of the Minkowski Sum as the expanded query range enables pruning techniques developed for traditional range query processing (such as pruning with a R-tree) to be used. The Query-Data Duality Theorem simplifies the cost of computing qualification probabilities for both IPQ and IUQ. When the Minkowski Sum and the Query-Data Duality Theorem are combined appropriately with the probability threshold constraint, more pruning opportunities can be created. Particularly, if the probability information about the uncertain objects is pre-computed and stored in the PTI, pruning based on the objects' uncertainty information can be done in the index level. In our future work, we will study how other location-dependent queries (such as the nearest-neighbor queries) can be supported. We will also consider queries and uncertain regions with non-rectangular shapes.

Acknowledgement

The work described in this paper was partially supported by a grant from the Central Policy Unit of the Goverment of the Hong Kong SAR and The Research Grants Council of the Hong Kong SAR, China (Project No. PolyU 5138/06E). We would like to thank the anonymous reviewers for their insightful comments and suggestions.

References

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

[2] M. Berg, M. Kreveld, M. Overmars, and O. Schwarzkopf. *Computational Geometry – Algorithms and Applications, 2nd ed.* Springer Verlag, 2000.

[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, Y. Xia, S. Prabhakar, R. Shah, and J. S. Vitter. Efficient indexing methods for probabilistic threshold queries over uncertain data. In *Proc. VLDB*, 2004.

[6] R. Cheng, Y. Zhang, E. Bertino, and S. Prabhakar. Preserving user location privacy in mobile data management infrastructures. In *Proc. of the 6th Workshop on Privacy Enhancing Technologies*, June 2006.

[7] D.Pfoser and C. Jensen. Capturing the uncertainty of moving-objects representations. In *Proc. SSDBM*, 1999.

[8] B. Gedik and L. Liu. Mobieyes: Distributed processing of continuously moving queries on moving objects in a mobile system. In *EDBT*, 2004.

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

[10] A. Guttman. R-trees: A dynamic index structure for spatial searching. *Proc. of the ACM SIGMOD Int'l. Conf.*, 1984.

[11] H. Hu and D. Lee. Range nearest-neighbor query. *IEEE TKDE*, 18(1), 2006.

[12] S. Ilarri, E. Mena, and A. Illarramendi. Location-dependent queries in mobile contexts: Distributed processing using mobile agents. *ACM TMC*, 5(8), August 2006.

[13] D. Lee, J. Xu, and B. Zheng. Data management in location-dependent information services. *IEEE Pervasive Computing*, 1(3), 2002.

[14] M.Hadjieleftheriou. Spatial index library version 0.44.2b.

[15] M. F. Mokbel, C.-Y. Chow, and W. G. Aref. The new casper: Query processing for location services without compromising privacy. In *VLDB*, 2006.

[16] J. Nievergelt, H. Hinterberger, and K. Sevcik. The grid file: an adaptable, symmetric multikey file structure. *ACM Trans. Database Syst.*, 9(1), 1984.

[17] P. A. Sistla, O. Wolfson, S. Chamberlain, and S. Dao. Querying the uncertain position of moving objects. In *Temporal Databases: Research and Practice*. 1998.

[18] Z. Song and N. Roussopoulos. K-nearest neighbor search for moving query point. In *Proc. SSTD*, 2001.

[19] Y. Tao, R. Cheng, X. Xiao, W. K. Ngai, B. Kao, and S. Prabhakar. Indexing multi-dimensional uncertain data with arbitrary probability density functions. In *Proc. VLDB*, 2005.

[20] Y. Tao, D. Papadias, and Q. Shen. Continuous nearest neighbor search. In *Proc. VLDB*, 2002.

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