

Efficient Computation of Range Aggregates against Uncertain Location Based Queries

Ying Zhang¹ Xuemin Lin^{1,2} Yufei Tao³ Wenjie Zhang¹ Haixun Wang⁴

¹ University of New South Wales, {yingz, lxue, zhangw}@cse.unsw.edu.au ² NICTA

³ Chinese University of Hong Kong, taoyf@cse.cuhk.edu.hk ⁴ Microsoft Research Asia, haixunw@microsoft.com

Abstract—In many applications, including location based services, queries may not be precise. In this paper, we study the problem of efficiently computing range aggregates in a multidimensional space when the query location is uncertain. Specifically, for a query point Q whose location is uncertain and a set S of points in a multi-dimensional space, we want to calculate the aggregate (e.g., *count*, *average* and *sum*) over the subset S' of S such that for each $p \in S'$, Q has at least probability θ within the distance γ to p . We propose novel, efficient techniques to solve the problem following the *filtering-and-verification* paradigm. In particular, two novel filtering techniques are proposed to effectively and efficiently remove data points from verification. Our comprehensive experiments based on both real and synthetic data demonstrate the efficiency and scalability of our techniques.

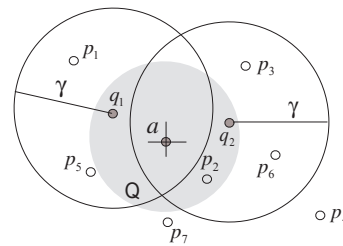
Index Terms—Uncertainty, Index, Range aggregate query

1 INTRODUCTION

Query imprecision or uncertainty may be often caused by the nature of many applications, including location based services. The existing techniques for processing location based spatial queries regarding certain query points and data points are not applicable or inefficient when uncertain queries are involved. In this paper, we investigate the problem of efficiently computing distance based range aggregates over certain data points and uncertain query points as described in the abstract. In general, an *uncertain query* Q is a multi-dimensional point that might appear at any location x following a probabilistic density function $pdf(x)$ within a region $Q.region$. There is a number of applications where a query point may be uncertain. Below are two sample applications.

Motivating Application 1. A blast warhead carried by a missile may destroy things by blast pressure waves in its *lethal area* where the lethal area is typically a circular area centered at the point of explosion (blast point) with radius γ [24] and γ depends on the explosive used. While firing such a missile, even the most advanced laser-guided missile cannot exactly hit the aiming point with 100% guarantee. The actual falling point (blast point) of a missile blast warhead regarding a target point usually follows some probability density functions (*PDFs*); different *PDFs* have been studied in [24] where *bivariate normal* distribution is the simplest and the most common one [24]. In military applications, firing such a missile may not only destroy military targets but may also damage civilian objects. Therefore, it is important to avoid the civilian casualties by estimating the likelihood of damaging civilian objects once the aiming point of a blast missile is determined. As depicted in Fig. 1, points $\{p_i\}$ for $1 \leq i \leq 7$ represent some civilian objects (e.g., residential buildings, public facilities). If q_1 in Fig. 1 is the actual falling point of the missile, then objects p_1 and

p_5 will be destroyed. Similarly, objects p_2, p_3 and p_6 will be destroyed if the actual falling point is q_2 . In this application, the risk of civilian casualties may be measured by the total number n of civilian objects which are within γ distance away from a possible blast point with at least θ probability. Note that the probabilistic threshold is set by the commander based on the levels of trade-off that she wants to make between the risk of civilian damages and the effectiveness of military attacks; for instance, it is unlikely to cause civilian casualties if $n = 0$ with a small θ . Moreover, different weight values may be assigned to these target points and hence the aggregate can be conducted based on the *sum* of the values.



Q : shadowed region to indicate the possible locations of the query
 q_1, q_2 : to indicate two possible locations of Q
 γ : query distance

Fig. 1. Missile Example

Motivating Application 2. Similarly, we can also estimate the effectiveness of a police vehicle patrol route using range aggregate against uncertain location based query Q . For example, Q in Fig. 1 now corresponds to the possible locations of a police patrol vehicle in a patrol route. A spot (e.g., restaurant, hotel, residential property), represented by a point in $\{p_1, p_2, \dots, p_7\}$ in Fig. 1, is likely under reliable *police patrol coverage* [11] if it has at least θ probability within γ distance to a moving patrol vehicle, where γ and θ are set by domain experts. The number of spots under reliable *police patrol coverage* is often deployed to evaluate the effectiveness

of the police patrol route.

Motivated by the above applications, in the paper we study the problem of aggregate computation against the data points which have at least probability θ to be within distance γ regarding an uncertain location based query.

Challenges. A naive way to solve this problem is that for each data point $p \in S$, we calculate the probability, namely *falling probability*, of Q within γ distance to p , select p against a given probability threshold, and then conduct the aggregate. This involves the calculation of an integral regarding each p and $Q.pdf$ for each $p \in S$; unless $Q.pdf$ has a very simple distribution (e.g., uniform distributions), such a calculation may often be very expensive and the naive method may be computationally prohibitive when a large number of data points is involved. In the paper we target the problem of efficiently computing range aggregates against an uncertain Q for arbitrary $Q.pdf$ and $Q.region$. Note that when $Q.pdf$ is a uniform distribution within a circular region $Q.region$, a circular “window” can be immediately obtained according to γ and $Q.region$ so that the computation of range aggregates can be conducted via the window aggregates [27] over S .

Contributions. Our techniques are developed based on the standard *filtering-and-verification* paradigm. We first discuss how to apply the existing probabilistically constrained regions (PCR) technique [26] to our problem. Then, we propose two novel distance based filtering techniques, statistical filtering (STF) and anchor point filtering (APF), respectively, to address the inherent limits of the PCR technique. The basic idea of the STF technique is to bound the falling probability of the points by applying some well known statistical inequalities where only a small amount of statistic information about the uncertain location based query Q is required. The STF technique is simple and space efficient (only $d + 2$ float numbers required where d denotes the dimensionality), and experiments show that it is effective. For the scenarios where a considerable “large” space is available, we propose a view based filter which consists of a set of *anchor points*. An *anchor point* may reside at any location and its falling probability regarding Q is pre-computed for several γ values. Then many data points might be effectively filtered based on their distances to the *anchor points*. For a given space budget, we investigate how to construct the anchor points and their accessing orders.

To the best of our knowledge, we are the first to identify the problem of computing range aggregates against uncertain location based query. In this paper, we investigate the problem regarding both *continuous* and *discrete* Q . Our principle contributions can be summarized as follows.

- We propose two novel filtering techniques, *STF* and *APF*, respectively. The *STF* technique has a decent filtering power and only requires the storage of very limited pre-computed information. *APF* provides the flexibility to significantly enhance the filtering power by demanding more pre-computed informa-

tion to be stored. Both of them can be applied to *continuous* case and *discrete* case.

- Extensive experiments are conducted to demonstrate the efficiency of our techniques.
- While we focus on the problem of range *counting* for uncertain location based queries in the paper, our techniques can be immediately extended to other range aggregates.

The remainder of the paper is organized as follows. Section 2 formally defines the problem and presents preliminaries. In Section 3, following the *filtering-and-verification* framework, we propose three filtering techniques. Section 4 evaluates the proposed techniques with extensive experiments. Then some possible extensions of our techniques are discussed in Section 5. This is followed by related work in Section 6. Section 7 concludes the paper.

2 BACKGROUND INFORMATION

We first formally define the problem in Section 2.1, then Section 2.2 presents the PCR technique [26] which is employed in the filtering technique proposed in Section 3.3.

Notation	Definition
Q	uncertain location based query
S	a set of points
q	instance of an uncertain query Q
d	dimensionality
P_q	the probability of the q to appear
θ and γ	probabilistic threshold and query distance
$P_{fall}(Q, p, \gamma)$	the falling probability of p regarding Q and γ
$Q_{\theta, \gamma}(S)$	$\{p p \in S \wedge P_{fall}(Q, p, \gamma) \geq \theta\}$
$p, x, y, b(S)$	point (a set of data points)
e	R tree entry
$C_{p,r}$	a circle(sphere) centred at p with radius r
$\delta(x, y)$	the <i>distance</i> between x and y
$\delta_{max(min)}(r_1, r_2)$	the maximal(minimal) distance between two rectangular regions
g_Q	<i>mean</i> of Q
η_Q	weighted average distance of Q
σ_Q	<i>variance</i> of Q
ϵ	arbitrarily small positive constant value
a	<i>anchor point</i>
n_{ap}	the number of <i>anchor points</i>
$LP_{fall}(p, \gamma)$	lower bound of the $P_{fall}(p, \gamma)$
$UP_{fall}(p, \gamma)$	upper bound of the $P_{fall}(p, \gamma)$
n_d	the number of different distances pre-computed for each <i>anchor point</i>
D_a	a set of distance values used by <i>anchor point</i> a

TABLE 1

The summary of notations.

2.1 Problem Definition

In the paper, S is a set of points in a d -dimensional numerical space. The distance between two points x and y is denoted by $\delta(x, y)$. Note that techniques developed in the paper can be applied to any *distance* metrics [5]. In the examples and experiments, the *Euclidean* distance is used. For two rectangular regions r_1 and r_2 , we have $\delta_{max}(r_1, r_2) = \max_{\forall x \in r_1, y \in r_2} \delta(x, y)$ and

$$\delta_{min}(r_1, r_2) = \begin{cases} 0 & \text{if } r_1 \cap r_2 \neq \emptyset \\ \min_{\forall x \in r_1, y \in r_2} \delta(x, y) & \text{otherwise} \end{cases} \quad (1)$$

An uncertain (location based) query Q may be described by a *continuous* or a *discrete* distribution as follows.

Definition 1 (Continuous Distribution). An uncertain query Q is described by a probabilistic density function $Q.pdf$. Let $Q.region$ represent the region where Q might appear, then $\int_{x \in Q.region} Q.pdf(x)dx = 1$;

Definition 2 (Discrete Distribution). An uncertain query Q consists of a set of instances (points) $\{q_1, q_2, \dots, q_n\}$ in a d -dimensional numerical space where q_i appears with probability P_{q_i} , and $\sum_{q \in Q} P_q = 1$;

Note that, in Section 5 we also cover the applications where Q can have a non-zero probability to be absent; that is, $\int_{x \in Q.region} Q.pdf(x)dx = c$ or $\sum_{q \in Q} P_q = c$ for a $c < 1$.

For a point p , we use $P_{fall}(Q, p, \gamma)$ to represent the probability of Q within γ distance to p , called *falling* probability of p regarding Q and γ . It is formally defined below.

For *continuous* cases,

$$P_{fall}(Q, p, \gamma) = \int_{x \in Q.region \wedge \delta(x,p) \leq \gamma} Q.pdf(x)dx \quad (2)$$

For *discrete* cases,

$$P_{fall}(Q, p, \gamma) = \sum_{q \in Q \wedge \delta(q,p) \leq \gamma} P_q \quad (3)$$

In the paper hereafter, $P_{fall}(Q, p, \gamma)$ is abbreviated to $P_{fall}(p, \gamma)$, and $Q.region$ and $Q.pdf$ are abbreviated to Q and pdf respectively, whenever there is no ambiguity. It is immediate that $P_{fall}(p, \gamma)$ is a monotonically increasing function with respect to distance γ .

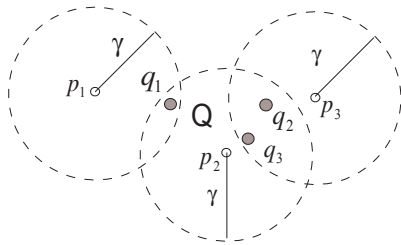


Fig. 2. Example of $P_{fall}(Q, p, \gamma)$

Problem Statement.

In many applications, users are only interested in the points with falling probabilities exceeding a given probabilistic threshold regarding Q and γ . In this paper we investigate the problem of probabilistic threshold based uncertain location range aggregate query on points data; it is formally defined below.

Definition 3. [Uncertain Range Aggregate Query] Given a set S of points, an uncertain query Q , a query distance γ and a probabilistic threshold θ , we want to compute an aggregate function (e.g., count, avg, and sum) against points $p \in Q_{\theta, \gamma}(S)$, where $Q_{\theta, \gamma}(S)$ denotes a subset of points $\{p\} \subseteq S$ such that $P_{fall}(p, \gamma) \geq \theta$.

In this paper, our techniques will be presented based on the aggregate *count*. Nevertheless, they can be immediately extended to cover other aggregates, such as *min*,

max, *sum*, *avg*, etc., over some non-locational attributes (e.g., weight value of the object in missile example).

Example 1. In Fig. 2, $S = \{p_1, p_2, p_3\}$ and $Q = \{q_1, q_2, q_3\}$ where $P_{q_1} = 0.4$, $P_{q_2} = 0.3$ and $P_{q_3} = 0.3$. According to Definition 3, for the given γ , we have $P_{fall}(p_1, \gamma) = 0.4$, $P_{fall}(p_2, \gamma) = 1$, and $P_{fall}(p_3, \gamma) = 0.6$. Therefore, $Q_{\theta, \gamma}(S) = \{p_2, p_3\}$ if θ is set to 0.5, and hence $|Q_{\theta, \gamma}(S)| = 2$.

2.2 Probabilistically Constrained Regions (PCR)

In [26], Tao *et al.* study the problem of range query on uncertain objects, in which the query is a rectangular window and the location of each object is uncertain. Although the problem studied in [26] is different with the one in this paper, in Section 3.3 we show how to modify the techniques developed in [26] to support uncertain location based query.

In the following part, we briefly introduce the Probabilistically Constrained Region (PCR) technique developed in [26]. Same as the uncertain location based query, an uncertain object U is modeled by a probability density function $U.pdf(x)$ and an uncertain region $U.region$. The probability that the uncertain object U falls in the rectangular window query r_q , denoted by $P_{fall}(U, r_q)$, is defined as $\int_{x \in U.region \cap r_q} U.pdf(x)dx$. In [26], the probabilistically constrained region of the uncertain object U regarding probability θ ($0 \leq \theta \leq 0.5$), denoted by $U.pcr(\theta)$, is employed in the filtering technique. Particularly, $U.pcr(\theta)$ is a rectangular region constructed as follows.

For each dimension i , the projection of $U.pcr(\theta)$ is denoted by $[U.pcr_{i-}(\theta), U.pcr_{i+}(\theta)]$ where $\int_{x \in U.region \& x_i \leq U.pcr_{i-}(\theta)} U.pdf(x)dx = \theta$ and $\int_{x \in U.region \& x_i \geq U.pcr_{i+}(\theta)} U.pdf(x)dx = \theta$. Note that x_i represents the coordinate value of the point x on i -th dimension. Then $U.pcr(\theta)$ corresponds to a rectangular region $[U.pcr_{-}(\theta), U.pcr_{+}(\theta)]$ where $U.pcr_{-}(\theta)$ ($U.pcr_{+}(\theta)$) is the lower (upper) corner and the coordinate value of $U.pcr_{-}(\theta)$ ($U.pcr_{+}(\theta)$) on i -th dimension is $U.pcr_{i-}(\theta)$ ($U.pcr_{i+}(\theta)$). Fig. 3(a) illustrates the $U.pcr(0.2)$ of the uncertain object U in 2 dimensional space. Therefore, the probability mass of U on the left (right) side of l_{1-} (l_{1+}) is 0.2 and the probability mass of U below (above) the l_{2-} (l_{2+}) is 0.2 as well. Following is a motivating example of how to derive the lower and upper bounds of the falling probability based on PCR.

Example 2. According to the definition of PCR, in Fig. 3(b) the probabilistic mass of U in the shaded area is 0.2, i.e., $\int_{x \in U.region \& x_i \geq U.pcr_{i+}(\theta)} U.pdf(x)dx = 0.2$. Then, it is immediate that $P_{fall}(U, r_{q_1}) < 0.2$ because r_{q_1} does not intersect $U.pcr(0.2)$. Similarly, we have $P_{fall}(U, r_{q_2}) \geq 0.2$ because the shaded area is enclosed by r_{q_2} .

The following theorem [26] formally introduces how to *prune* or *validate* an uncertain object U based on $U.pcr(\theta)$ or $U.pcr(1 - \theta)$. Note that we say an uncertain object is *pruned* (*validated*) if we can claim $P_{fall}(U, r_q) < \theta$ ($P_{fall}(U, r_q) \geq \theta$) based on the PCR.

Theorem 1. Given an uncertain object U , a range query r_q (r_q is a rectangular window) and a probabilistic threshold θ .

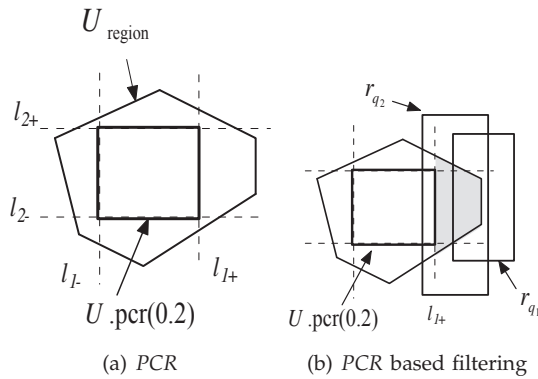


Fig. 3. A 2d probabilistically constrained region (PCR (0.2))

- 1) For $\theta > 0.5$, U can be pruned if r_q does not fully contain $U.pcr(1 - \theta)$;
- 2) For $\theta \leq 0.5$, the pruning condition is that r_q does not intersect $U.pcr(\theta)$;
- 3) For $\theta > 0.5$, the validating criterion is that r_q completely contains the part of $U_{m_{bb}}$ on the right (left) of plane $U.pcr_{i-}(1-\theta)$ ($U.pcr_{i+}(1-\theta)$) for some $i \in [1, d]$, where $U_{m_{bb}}$ is the minimal bounding box of uncertain region $U.region$;
- 4) For $\theta \leq 0.5$ the validating criterion is that r_q completely contains the part of $U_{m_{bb}}$ on the left (right) of plane $U.pcr_{i-}(\theta)$ ($U.pcr_{i+}(\theta)$) for some $i \in [1, d]$;

3 Filtering-and-Verification ALGORITHM

According to the definition of falling probability (i.e., $P_{fall}(p, \gamma)$) in Equation 2, the computation involves integral calculation, which may be expensive in terms of CPU cost. Based on Definition 3, we only need to know whether or not the falling probability of a particular point regarding Q and γ exceeds the probabilistic threshold for the uncertain aggregate range query. This motivates us to follow the filtering-and-verification paradigm for the uncertain aggregate query computation. Particularly, in the filtering phase, effective and efficient filtering techniques will be applied to prune or validate the points. We say a point p is pruned (validated) regarding the uncertain query Q , distance γ and probabilistic threshold θ if we can claim that $P_{fall}(p, \gamma) < \theta$ ($P_{fall}(p, \gamma) \geq \theta$) based on the filtering techniques without explicitly computing the $P_{fall}(p, \gamma)$. The points that cannot be pruned or validated will be verified in the verification phase in which their falling probabilities are calculated. Therefore, it is desirable to develop effective and efficient filtering techniques to prune or validate points such that the number of points being verified can be significantly reduced.

In this section, we first present a general framework for the filtering-and-verification Algorithm based on filtering techniques in Section 3.1. Then a set of filtering techniques are proposed. Particularly, Section 3.2 proposes the statistical filtering technique. Then we investigate how to apply the PCR based filtering technique in Section 3.3. Section 3.4 presents the anchor point based filtering technique.

For presentation simplicity, we consider the continuous case of the uncertain query in this section. Section 3.5

shows that techniques proposed in this section can be immediately applied to the discrete case.

3.1 A framework for filtering-and-verification Algorithm

In this subsection, following the filtering-and-verification paradigm we present a general framework to support uncertain range aggregate query based on the filtering technique. To facilitate the aggregate query computation, we assume a set S of points is organized by an aggregate R -Tree [22], denoted by R_S . Note that an entry e of R_S might be a data entry or an intermediate entry where a data entry corresponds to a point in S and an intermediate entry groups a set of data entries or child intermediate entries. Assume a filter, denoted by F , is available to prune or validate a data entry (i.e., a point) or an intermediate entry (i.e., a set of points).

Algorithm 1 illustrates the framework of the filtering-and-verification Algorithm. Note that details of the filtering techniques will be introduced in the following subsections. The algorithm consists of two phases. In the filtering phase (Line 3-16), for each entry e of R_S to be processed, we do not need to further process e if it is pruned or validated by the filter F . We say an entry e is pruned (validated) if the filter can claim $P_{fall}(p, \gamma) < \theta$ ($P_{fall}(p, \gamma) \geq \theta$) for any point p within $e_{m_{bb}}$. The counter cn is increased by $|e|$ (Line 6) if e is validated where $|e|$ denotes the aggregate value of e (i.e., the number of data points in e). Otherwise, the point p associated with e is a candidate point if e corresponds to a data entry (Line 10), and all child entries of e are put into the queue for further processing if e is an intermediate entry (Line 12). The filtering phase terminates when the queue is empty. In the verification phase (Line 17-21), candidate points are verified by the integral calculations according to Equation 2.

Cost Analysis. The total time cost of Algorithm 1 is as follows.

$$Cost = N_f \times C_f + N_{io} \times C_{io} + N_{ca} \times C_{vf} \quad (4)$$

Particularly, N_f represents the number of entries being tested by the filter on Line 5 and C_f is the time cost for each test. N_{io} denotes the number of nodes (pages) accessed (Line 13) and C_{io} corresponds to the delay of each node (page) access of R_S . N_{ca} represents the size of candidate set C and C_{vf} is the computation cost for each verification (Line 15) in which numerical integral computation is required. With a reasonable filtering time cost (i.e., C_{vf}), the dominant cost of Algorithm 1 is determined by N_{io} and N_{ca} because C_{io} and C_{vf} might be expensive. Therefore, in the paper we aim to develop effective and efficient filtering techniques to reduce N_{ca} and N_{io} .

Filtering. Suppose there is no filter F in Algorithm 1, all points in S will be verified. Regarding the example in Fig. 4, 5 points p_1, p_2, p_3, p_4 and p_5 will be verified. A straightforward filtering technique is based on the minimal and maximal distances between the minimal

Algorithm 1 Filtering-and-Verification($R_S, Q, F, \gamma, \theta$)

Input: R_S : an aggregate R tree on data set S ,
 Q : uncertain query, F : Filter, γ : query distance,
 θ : probabilistic threshold.

Output: $|Q_{\theta, \gamma}(S)|$

Description:

```

1: Queue :=  $\emptyset$ ; cn := 0; C :=  $\emptyset$ ;
2: Insert root of  $R_S$  into Queue;
3: while Queue  $\neq \emptyset$  do
4:   e  $\leftarrow$  dequeue from the Queue;
5:   if e is validated by the filter F then
6:     cn := cn + |e|;
7:   else
8:     if e is not pruned by the filter F then
9:       if e is data entry then
10:        C := C  $\cup$  p where p is the data point e
           represented;
11:      else
12:        put all child entries of e into Queue;
13:      end if
14:    end if
15:  end if
16: end while
17: for each point p  $\in$  C do
18:   if  $P_{fall}(Q, p, \gamma) \geq \theta$  then
19:     cn := cn + 1;
20:   end if
21: end for
22: Return cn
    
```

bounding boxes(MBBs) of an entry and the uncertain query. Clearly, for any θ we can safely *prune* an entry if $\delta_{min}(Q_{mbb}, e_{mbb}) > \gamma$ or *validate* it if $\delta_{max}(Q_{mbb}, e_{mbb}) \leq \gamma$. We refer this as *maximal/minimal distance based filtering technique*, namely *MMD*. *MMD* technique is time efficient as it takes only $O(d)$ time to compute the minimal and maximal distances between Q_{mbb} and e_{mbb} . Recall that Q_{mbb} is the minimal bounding box of Q .region.

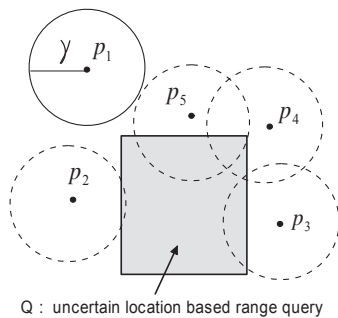


Fig. 4. Running Example

Example 3. As shown in Fig. 4, suppose the *MMD* filtering technique is applied in Algorithm 1, then p_1 is pruned and the other 4 points p_2, p_3, p_4 and p_5 will be verified.

Although the *MMD* technique is very time efficient, its filtering capacity is limited because it does not make use of the distribution information of the uncertain query Q and the probabilistic threshold θ . This motivates us to develop more effective filtering techniques based on some pre-computations on the uncertain query Q such that the number of entries (i.e., points) being pruned or validated

in Algorithm 1 is significantly increased. In the following subsections, we present three filtering techniques, named *STF*, *PCR* and *APF* respectively, which can significantly enhance the filtering capability of the filter.

3.2 Statistical Filter

In this subsection, we propose a statistical filtering technique, namely *STF*. After introducing the motivation of the technique, we present some important statistic information of the uncertain query and then show how to derive the lower and upper bounds of the *falling probability* of a point regarding an uncertain query Q , distance γ and probabilistic threshold θ .

Motivation. As shown in Fig. 5, given an uncertain query Q_1 and γ we cannot *prune* point p based on the *MMD* technique, regardless of the value of θ , although intuitively the falling probability of p regarding Q_1 is likely to be small. Similarly, we cannot *validate* p for Q_2 . This motivates us to develop a new filtering technique which is as simple as *MMD*, but can exploit θ to enhance the filtering capability. In the following part, we show that lower and upper bounds of $P_{fall}(p, \gamma)$ can be derived based on some statistics of the uncertain query. Then a point may be immediately *pruned* (*validated*) based on the upper(lower) bound of $P_{fall}(p, \gamma)$, denoted by $UP_{fall}(p, \gamma)$ ($LP_{fall}(p, \gamma)$).

Example 4. In Fig. 5 suppose $\theta = 0.5$ and we have $UP_{fall}(Q_1, p, \gamma) = 0.4$ ($LP_{fall}(Q_2, p, \gamma) = 0.6$) based on the statistical bounds, then p can be safely pruned (*validated*) without explicitly computing its falling probability regarding Q_1 (Q_2). Regarding the running example in Fig.4, suppose $\theta = 0.2$ and we have $UP_{fall}(p_2, \gamma) = 0.15$ while $UP_{fall}(p_i, \gamma) \geq 0.2$ for $3 \leq i \leq 5$, then p_2 is pruned. Therefore, three points (p_3, p_4 and p_5) are verified in Algorithm 1 when *MMD* and statistical filtering techniques are applied.

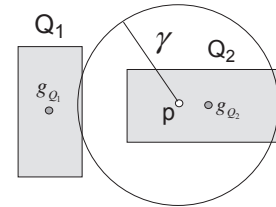


Fig. 5. Motivation Example

Statistics of the uncertain query

To apply the statistical filtering technique, the following statistics of the uncertain query Q are pre-computed.

Definition 4 (mean (g_Q)). $g_Q = \int_{x \in Q} x \times Q.pdf(x)dx$.

Definition 5 (weighted average distance (η_Q)). η_Q equals $\int_{x \in Q} \delta(x, g_Q) \times Q.pdf(x)dx$

Definition 6 (variance (σ_Q)). σ_Q equals $\int_{x \in Q} \delta(x, g_Q)^2 \times Q.pdf(x)dx$

Derive lower and upper bounds of $P_{fall}(p, \gamma)$.

For a point $p \in S$, the following theorem shows how to derive the lower and upper bounds of $P_{fall}(p, \gamma)$ based on above statistics of Q . Then, without explicitly computing $P_{fall}(p, \gamma)$, we may *prune* or *validate* the point p based on $UP_{fall}(p, \gamma)$ and $LP_{fall}(p, \gamma)$ derived based on the statistics of Q .

Theorem 2. Given an uncertain query Q and a distance γ , and suppose the mean g_Q , weighted average distance η_Q and variance σ_Q of Q are available. Then for a point p , we have

- 1) If $\gamma > \mu_1$, $P_{fall}(p, \gamma) \geq 1 - \frac{1}{1 + \frac{(\gamma - \mu_1)^2}{\sigma_1^2}}$, where $\mu_1 = \delta(g_Q, p) + \eta_Q$ and $\sigma_1^2 = \sigma_Q - \eta_Q^2 + 4\eta_Q \times \delta(g_Q, p)$.
- 2) If $\gamma < \delta(g_Q, p) - \eta_Q - \epsilon$, $P_{fall}(p, \gamma) \leq \frac{1}{1 + \frac{(\gamma' - \mu_2)^2}{\sigma_2^2}}$, where $\mu_2^2 = \Delta + \eta_Q$, $\sigma_2^2 = \sigma_Q - \eta_Q^2 + 4\eta_Q \times \Delta$, $\Delta = \gamma + \gamma' + \epsilon - \delta(p, g_Q)$ and $\gamma' > 0$. The ϵ represents an arbitrarily small positive constant value.

Before the proof of Theorem 2, we first introduce the *Cantelli's Inequality* [19] described by Lemma 1 which is one-sided version of the *Chebyshev Inequality*.

Lemma 1. Let X be an univariate random variable with the expected value μ and the finite variance σ^2 . Then for any $C > 0$, $Pr(X - \mu \geq C \times \sigma) \leq \frac{1}{1 + C^2}$.

Following is the proof of Theorem 2.

Proof: Intuition of the Proof. For a given point $p \in S$, its distance to Q can be regarded as an univariate random variable Y , and we have $P_{fall}(p, \gamma) = Pr(Y \leq \gamma)$. Given γ , we can derive the lower and upper bounds of $Pr(Y \leq \gamma)$ ($P_{fall}(p, \gamma)$) based on the statistical inequality in Lemma 1 if the expectation ($E(Y)$) and variance ($Var(Y)$) of the random variable Y are available. Although $E(Y)$ and $Var(Y)$ take different values regarding different points, we show that the upper bounds of $E(Y)$ and $Var(Y)$ can be derived based on *mean(g_Q)*, *weighted average distance (η_Q)* and *variance(σ_Q)* of the query Q . Then, the correctness of the theorem follows.

Details of the Proof. The uncertain query Q is a random variable which equals $x \in Q.region$ with probability $Q.pdf(x)$. For a given point p , let Y denote the distance distribution between p and Q ; that is, Y is an univariate random variable and $Y.pdf(l) = \int_{x \in Q.region \text{ and } \delta(x,p)=l} Q.pdf(x)dx$ for any $l \geq 0$. Consequently, we have $P_{fall}(p, \gamma) = Pr(Y \leq \gamma)$ according to Equation 2. Let $\mu = E(Y)$, $\sigma^2 = Var(Y)$ and $C = \frac{\gamma - \mu}{\sigma}$, then based on lemma 1, if $\gamma > \mu$ we have

$$Pr(Y \geq \gamma) = Pr(Y - \mu \geq C \times \sigma) \leq \frac{1}{1 + \left(\frac{\gamma - \mu}{\sigma}\right)^2}$$

Then it is immediate that

$$Pr(Y \leq \gamma) \geq 1 - Pr(Y \geq \gamma) \geq 1 - \frac{1}{1 + \frac{(\gamma - \mu)^2}{\sigma^2}} \quad (5)$$

According to Inequation 5 we can derive the lower bound of $P_{fall}(p, \gamma)$. Next, we show how to derive upper bound of $P_{fall}(p, \gamma)$. As illustrated in Fig. 6, let p' denote a dummy point on the line $\overline{pg_Q}$ with $\delta(p', p) = \gamma + \gamma' + \epsilon$ where ϵ is an arbitrarily small positive constant value. Similar to the definition of Y , let Y' be the distance distribution between p' and Q ; that is, Y' is an univariate random variable where $Y'.pdf(l) = \int_{x \in Q.region \text{ and } \delta(x,p')=l} Q.pdf(x)dx$ for any $l \geq 0$. Then, as shown in Fig. 6, for any point $x \in Q$ with $\delta(x, p') \leq \gamma'$ (shaded area), we have $\delta(x, p) > \gamma$. This implies that $Pr(Y \leq \gamma) \leq Pr(Y' \geq \gamma')$. Let $\mu' = E(Y')$ and

$\sigma'^2 = Var(Y')$, according to Lemma 1 when $\gamma' > \mu'$ we have

$$Pr(Y \leq \gamma) \leq Pr(Y' \geq \gamma') \leq \frac{1}{1 + \frac{(\gamma' - \mu')^2}{\sigma'^2}} \quad (6)$$

Because values of μ , σ^2 , μ' and σ'^2 may change regarding different point $p \in S$, we cannot pre-compute them. Nevertheless, in the following part we show that their upper bounds can be derived based on the statistic information of the Q , which can be pre-computed based on the probabilistic distribution of Q .

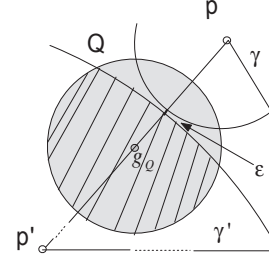


Fig. 6. Proof of Upper bound

Based on the **triangle inequality**, for any $x \in Q$ we have $\delta(x, p) \leq \delta(x, g_Q) + \delta(p, g_Q)$ and $\delta(x, p) \geq |\delta(x, g_Q) - \delta(p, g_Q)|$ for any $x \in Q$. Then we have

$$\begin{aligned} \mu &= \int_{y \in Y} y \times Y.pdf(y)dy = \int_{x \in Q} \delta(x, p) \times pdf(x)dx \\ &\leq \int_{x \in Q} (\delta(p, g_Q) + \delta(x, g_Q)) \times pdf(x)dx \\ &\leq \delta(g_Q, p) + \eta_Q = \mu_1 \end{aligned}$$

and

$$\begin{aligned} \sigma^2 &= E(Y^2) - E^2(Y) \\ &\leq \int_{x \in Q} (\delta(g_Q, p) + \delta(x, g_Q))^2 pdf(x)dx \\ &\quad - (\delta(g_Q, p) + \eta_Q)^2 \\ &= 2 \int_{x \in Q} \delta(g_Q, p) \times \delta(x, g_Q) \times pdf(x)dx \\ &\quad + \int_{x \in Q} \delta(x, g_Q)^2 \times pdf(x)dx \\ &\quad + 2 \times \delta(g_Q, p) \times \eta_Q - \eta_Q^2 \\ &= \sigma_Q - \eta_Q^2 + 4\eta_Q \times \delta(g_Q, p) = \sigma_1^2 \end{aligned}$$

Together with Inequality 5, we have $Pr(Y \leq \gamma) \geq 1 - \frac{1}{1 + \frac{(\gamma - \mu)^2}{\sigma^2}} \geq 1 - \frac{1}{1 + \frac{(\gamma - \mu_1)^2}{\sigma_1^2}}$ if $\mu_1 < \gamma$. With similar rationale, let $\Delta = \delta(g_Q, p')$ and $\mu_2 = \Delta + \eta_Q = \mu_2$ and $\sigma'^2 \leq \sigma_Q - \eta_Q^2 + 4\eta_Q \times \Delta = \sigma_2^2$. Based on Inequality 6, we have $Pr(Y \leq \gamma) \leq \frac{1}{1 + \frac{(\gamma' - \mu')^2}{\sigma'^2}} \leq \frac{1}{1 + \frac{(\gamma' - \mu_2)^2}{\sigma_2^2}}$ if $\gamma < \delta(g_Q, p) - \eta_Q - \epsilon$. Therefore, the correctness of the theorem follows. \square

The following extension is immediate based on the similar rationale of Theorem 2.

Extension 1. Suppose r is a rectangular region, we can use $\delta_{min}(r, g_Q)$ and $\delta_{max}(r, g_Q)$ to replace $\delta(p, g_Q)$ in Theorem 2 for lower and upper probabilistic bounds computation respectively.

Based on Extension 1, we can compute the upper and lower bounds of $P_{fall}(e_{mbb}, \gamma)$ where e_{mbb} is the minimal bounding box of the entry e , and hence *prune* or *validate* e in Algorithm 1. Since g_Q, η_Q and σ_Q are pre-computed, the dominant cost in filtering phase is the distance computation between e_{mbb} and g_Q which is $O(d)$.

3.3 PCR based Filter

Motivation. Although the statistical filtering technique can significantly reduce the candidate size in Algorithm 1, the filtering capacity is inherently limited because only a small amount of statistics are employed. This motivates us to develop more sophisticated filtering techniques to further improve the filtering capacity; that is, we aim to improve the filtering capacity with more pre-computations (i.e., more information kept for the filter). In this subsection, the PCR technique proposed in [26] will be modified for this purpose.

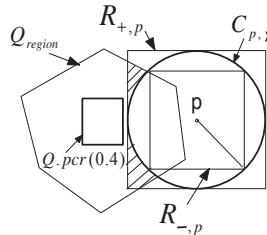


Fig. 7. Transform query

PCR based Filtering technique. The PCR technique proposed in [26] cannot be directly applied for filtering in Algorithm 1 because the range query studied in [26] is a rectangular window and objects are uncertain. Nevertheless we can adapt the PCR technique as follows. As shown in Fig. 7, let $C_{p,\gamma}$ represent the circle (sphere) centered at p with radius γ . Then we can regard the uncertain query Q and $C_{p,\gamma}$ as an uncertain object and the range query respectively. As suggested in [28], we can use $R_{+,p}$ (mbb of $C_{p,\gamma}$) and $R_{-,p}$ (inner box) as shown in Fig. 7 to *prune* and *validate* the point p based on the PCRs of Q respectively. For instance, if $\theta = 0.4$ the point p in Fig. 7 can be *pruned* according to case 2 of Theorem 1 because R_1 does not intersect $Q.pcr(0.4)$. Note that similar transformation can be applied for the intermediate entries as well.

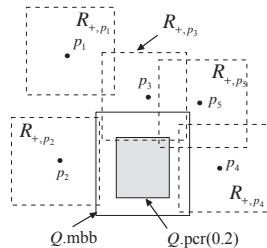


Fig. 8. Running example

Example 5. Regarding the running example in Fig. 8, suppose $Q.pcr(0.2)$ is pre-computed, then p_1, p_2 and p_4 are pruned because R_{+,p_1}, R_{+,p_2} and R_{+,p_4} do not overlap

$Q.pcr(0.2)$ according to Theorem 1 in Section 2.2. Consequently, only p_3 and p_5 go to the verification phase when $Q.pcr(0.2)$ is available.

Same as [26], [28], a finite number of PCRs are pre-computed for the uncertain query Q regarding different probability values. For a given θ at query time, if the $Q.pcr(\theta)$ is not pre-computed we can choose two pre-computed PCRs $Q.pcr(\theta_1)$ and $Q.pcr(\theta_2)$ where θ_1 (θ_2) is the largest (smallest) existing probability value smaller (larger) than θ . We can apply the modified PCR technique as the filter in Algorithm 1, and the filtering time regarding each entry tested is $O(m + \log(m))$ in the worst case, where m is the number of PCRs pre-computed by the filter.

The PCR technique can significantly enhance the filtering capacity when a particular number of PCR s are pre-computed. The key of the PCR filtering technique is to partition the uncertain query along each dimension. This may inherently limit the filtering capacity of the PCR based filtering technique. As shown in Fig. 7, we have to use two rectangular regions for *pruning* and *validation* purpose, and hence the $C_{p,\gamma}$ is enlarged (shrunk) during the computation. As illustrated in Fig. 7, all instances of Q in the striped area is counted for $P_{fall}(p, \gamma)$ regarding $R_{+,p}$, while all of them have distances larger than γ . Similar observation goes to $R_{-,p}$. This limitation is caused by the transformation, and cannot be remedied by increasing the number of PCRs. Our experiments also confirm that the PCR technique cannot take advantage of the large index space. This motivates us to develop new filtering technique to find a better trade-off between the filtering capacity and pre-computation cost (i.e., index size).

3.4 Anchor Points based Filter

The *anchor (pivot) point* technique is widely employed in various applications, which aims to reduce the query computation cost based on some pre-computed *anchor (pivot) points*. In this subsection, we investigate how to apply *anchor point* technique to effectively and efficiently reduce the candidate set size. Following is a motivating example for the *anchor point* based filtering technique.

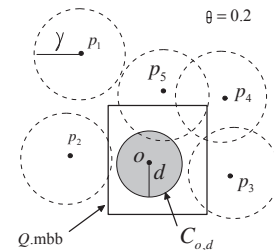


Fig. 9. Running example regarding the anchor point

Motivating Example. Regarding our running example, in Fig. 9 the shaded area, denoted by $C_{o,d}$, is the circle centered at o with radius d . Suppose the probabilistic mass of Q in $C_{o,d}$ is 0.8, then when $\theta = 0.2$ we can safely prune p_1, p_2, p_3 and p_4 because $C_{p_i,\gamma}$ does not intersect $C_{o,d}$ for $i = 1, 2, 3$ and 4.

In the paper, an anchor point a regarding the uncertain query Q is a point in multidimensional space whose falling probability against different γ values are pre-computed. We can *prune* or *validate* a point based on its distance to the *anchor point*. For better filtering capability, a set of *anchor points* will be employed.

In the following part, Section 3.4.1 presents the *anchor point* filtering technique. In Section 3.4.2, we investigate how to construct *anchor points* for a given space budget, followed by a time efficient filtering algorithm in Section 3.4.3.

3.4.1 Anchor Point filtering technique (APF)

For a given *anchor point* a regarding the uncertain query Q , suppose $P_{fall}(a, l)$ is pre-computed for arbitrary distance l . Lemma 2 provides lower and upper bounds of $P_{fall}(p, \gamma)$ for any point p based on the triangle inequality. This implies we can *prune* or *validate* a point based on its distance to an *anchor point*.

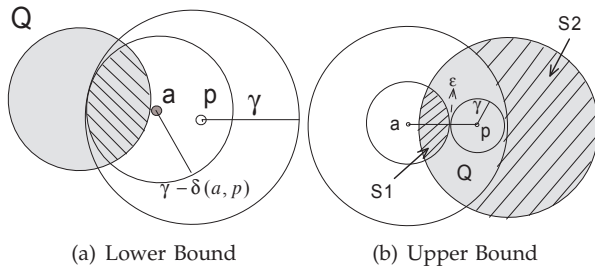


Fig. 10. Lower and Upper Bound

Lemma 2. Let a denote an anchor point regarding the uncertain query Q . For any point $p \in S$ and a distance γ , we have

- 1) If $\gamma > \delta(a, p)$, $P_{fall}(p, \gamma) \geq P_{fall}(a, \gamma - \delta(a, p))$.
- 2) $P_{fall}(p, \gamma) \leq P_{fall}(a, \delta(a, p) + \gamma) - P_{fall}(a, \delta(a, p) - \gamma - \epsilon)$ where ϵ is an arbitrarily small positive value.¹

Proof: Suppose $\gamma > \delta(a, p)$, then according to the triangle inequality for any $x \in Q$ with $\delta(x, a) \leq \gamma - \delta(a, p)$, we have $\delta(x, p) \leq \delta(a, p) + \delta(x, a) \leq \delta(a, p) + (\gamma - \delta(a, p)) = \gamma$. This implies that $P_{fall}(p, \gamma) \geq P_{fall}(a, \gamma - \delta(a, p))$ according to Equation 2. Fig. 10(a) illustrates an example of the proof in 2 dimensional space. In Fig. 10(a), we have $C_{a, \gamma - \delta(a, p)} \subseteq C_{p, \gamma}$ if $\gamma > \delta(a, p)$. Let S denote the striped area which is the intersection of $C_{a, \gamma - \delta(a, p)}$ and Q . Clearly, we have $P_{fall}(a, \gamma - \delta(a, p)) = \int_{x \in S} Q.pdf(x) dx$ and $\delta(x, p) \leq \gamma$ for any $x \in S$. Consequently, $P_{fall}(p, \gamma) \geq P_{fall}(a, \gamma - \delta(a, p))$ holds.

With similar rationale, for any $x \in Q$ we have $\delta(x, a) \leq \delta(a, p) + \gamma$ if $\delta(x, p) \leq \gamma$. This implies that $P_{fall}(p, \gamma) \leq P_{fall}(a, \delta(a, p) + \gamma)$. Moreover, for any $x \in Q$ with $\delta(x, a) \leq \delta(a, p) - \gamma - \epsilon$, we have $\delta(x, a) > \gamma$. Recall that ϵ represents an arbitrarily small constant value. This implies that x does not contribute to $P_{fall}(p, \gamma)$ if $\delta(x, a) \leq \delta(a, p) - \gamma - \epsilon$. Consequently, $P_{fall}(p, \gamma) \leq P_{fall}(a, \delta(a, p) + \gamma) - P_{fall}(a, \delta(a, p) - \gamma - \epsilon)$ holds. As shown in Fig. 10(b), we have $P_{fall}(p, \gamma) \leq P_{fall}(a, \delta(a, p) + \gamma)$ because $C_{p, \gamma} \subseteq C_{a, \delta(a, p) + \gamma}$. Since

1. We have $P_{fall}(a, \delta(a, p) - \gamma - \epsilon) = 0$ if $\delta(a, p) \leq \gamma$

$C_{a, \delta(a, p) - \gamma - \epsilon} \subseteq C_{a, \delta(a, p) + \gamma}$ and $C_{p, \gamma} \cap C_{p, \delta(a, p) - \gamma - \epsilon} = \emptyset$, this implies that $P_{fall}(p, \gamma) \leq P_{fall}(a, \delta(a, p) + \gamma) - P_{fall}(a, \delta(a, p) - \gamma - \epsilon)$. \square

Let $LP_{fall}(p, \gamma)$ and $UP_{fall}(p, \gamma)$ denote the lower and upper bounds derived from Lemma 2 regarding $P_{fall}(p, \gamma)$. Then we can immediately *validate* a point p if $LP_{fall}(p, \gamma) \geq \theta$, or *prune* p if $UP_{fall}(p, \gamma) < \theta$.

Clearly, it is infeasible to keep $P_{fall}(a, l)$ for arbitrary $l \geq 0$. Since $P_{fall}(a, l)$ is a monotonic function with respect to l , we keep a set $D_a = \{l_i\}$ with size n_d for each *anchor point* such that $P_{fall}(a, l_i) = \frac{i}{n_d}$ for $1 \leq i \leq n_d$. Then for any $l > 0$, we use $UP_{fall}(a, l)$ and $LP_{fall}(a, l)$ to represent the upper and lower bound of $P_{fall}(a, l)$ respectively. Particularly, $UP_{fall}(a, l) = P_{fall}(a, l_i)$ where l_i is the smallest $l_i \in D_a$ such that $l_i \geq l$. Similarly, $LP_{fall}(a, l) = P_{fall}(a, l_j)$ where l_j is the largest $l_j \in D_a$ such that $l_j \leq l$. Then we have the following theorem by rewriting Lemma 2 in a conservative way.

Theorem 3. Given an uncertain query Q and an anchor point a , for any rectangular region r and distance γ , we have:

- 1) If $\gamma > \delta_{max}(a, r)$, $P_{fall}(r, \gamma) \geq LP_{fall}(a, \gamma - \delta_{max}(a, r))$.
- 2) $P_{fall}(r, \gamma) \leq UP_{fall}(a, \delta_{max}(a, r) + \gamma) - LP_{fall}(a, \delta_{min}(a, r) - \gamma - \epsilon)$ where ϵ is an arbitrarily small positive value.

Let $LP_{fall}(r, \gamma)$ and $UP_{fall}(r, \gamma)$ represent the lower and upper bounds of the falling probability derived from Theorem 3. We can safely *prune* (*validate*) an entry e if $UP_{fall}(e_{mbb}, \gamma) < \theta$ ($LP_{fall}(e_{mbb}, \gamma) \geq \theta$). Recall that e_{mbb} represents the minimal bounding box of e . It takes $O(d)$ time to compute $\delta_{max}(a, e_{mbb})$ and $\delta_{min}(a, e_{mbb})$. Meanwhile, the computation of $LP_{fall}(a, l)$ and $UP_{fall}(a, l)$ for any $l > 0$ costs $O(\log n_d)$ time because pre-computed distance values in D_a are sorted. Therefore, the filtering time of each entry is $O(d + \log n_d)$ for each *anchor point*.

3.4.2 Heuristic with a finite number of anchor points

Let AP denote a set of *anchor points* for the uncertain query Q . We do not need to further process an entry e in Algorithm 1 if it is filtered by **any** anchor point $a \in AP$. Intuitively, the more *anchor points* employed by Q , the more powerful the filter will be. However, we cannot employ a large number of *anchor points* due to the space and filtering time limitations. Therefore, it is important to investigate how to choose a limited number of *anchor points* such that the filter can work effectively.

Anchor points construction. We first investigate how to evaluate the “goodness” of an *anchor point* regarding the computation of $LP_{fall}(p, \gamma)$. Suppose all *anchor points* have the same falling probability functions; that is $P_{fall}(a_i, l) = P_{fall}(a_j, l)$ for any two *anchor points* a_i and a_j . Then the closest *anchor point* regarding p will provide the largest $LP_{fall}(p, \gamma)$. Since there is no a priori knowledge about the distribution of the points, we assume they follow the uniform distribution. Therefore, *anchor points* should be uniformly distributed. If falling probabilistic functions of the *anchor points* are different,

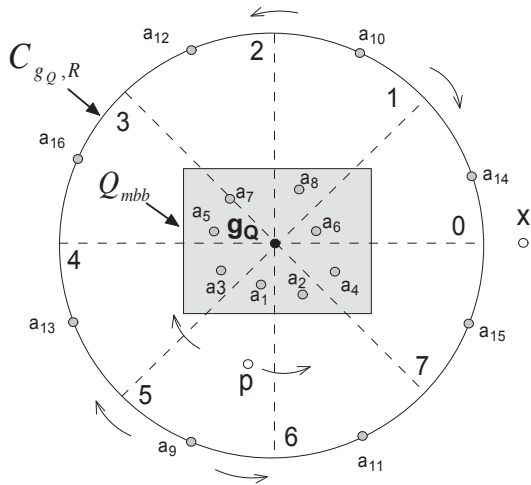


Fig. 11. *anchor points construction*

besides the distance a point p prefers the *anchor point* whose falling probability grows rapidly with respect to distance l . However, since it is costly to evaluate the falling probability functions for all possible *anchor points*, we only consider that *anchor points* should be evenly distributed.

As to $UP_{fall}(p, \gamma)$, observations above do not hold. According to Lemma 2, $UP_{fall}(p, \gamma)$ equals $P_{fall}(a, \delta(a, p) + \gamma) - P_{fall}(a, \delta(a, p) - \gamma - \epsilon)$. Clearly $P_{fall}(a, \delta(a, p) + \gamma)$ prefers a small $\delta(a, p)$, while $P_{fall}(a, \delta(a, p) - \gamma - \epsilon)$ is in favor of large $\delta(a, p)$. Therefore, for $UP_{fall}(p, \gamma)$ we evaluate the “goodness” of an anchor point from another point of view. As shown in Fig. 10(b), let R denote the shaded area which is the intersection of S and Q , where S represents $C_{a, \delta(a, p) + \gamma} - C_{a, \delta(a, p) - \gamma - \epsilon}$. Intuitively R with smaller area is more likely to lead to a tighter upper bound. Consequently the anchor point should be far from Q .

Considering the different preferences of LP_{fall} and UP_{fall} , we employ two kinds of *anchor points*, namely *inner anchor points* (IAPs) and *outer anchor points* (OAPs) respectively, where *inner anchor points* and *outer anchor points* are chosen in favor of *validating* and *pruning* respectively. Let k_i and k_o denote the number of IAPs and OAPs respectively, and the total number of anchor points is denoted by n_{ap} where $n_{ap} = k_i + k_o$. We assume $k_i = k_o$ in the paper. Note that the construction time of each anchor point is $O(n_d \times C_I)$ where C_I is the cost to identify each $l_i \in D_a$ such that $P_{fall}(a, l_i) = \frac{i}{n_d}$.

Fig. 11 illustrates how to construct *anchor points* on 2-dimensional data with $n_{ap} = 16$. Considering that points far from Q have less chance to be *validated*, we choose 8 IAPs such that their locations are uniformly distributed within Q_{mbb} . They are a_1, a_2, \dots, a_8 in Fig. 11. Then another 8 OAPs ($a_9, a_{10}, \dots, a_{16}$ in Fig. 11) are evenly located along $C_{g_Q, R}$ where R is set to $\frac{L}{2}$ in our experiment and L is the domain size. For each anchor point a , $P_{fall}(a, l)$ is computed for each $l \in D_a$. This construction approach can be easily extended to higher dimensions.

3.4.3 Time efficient filtering Algorithm

Based on Theorem 3, we can compute LP_{fall} and UP_{fall} for each entry e against the *anchor points* as discussed in Section 3.4.1, and then try to *prune* or *validate* e . Nevertheless, we show that the filtering time cost can be improved by the following techniques.

1. Quick-Filtering Algorithm. Since in Algorithm 1 we only need to check whether an entry e can be filtered, the explicit computation of LP_{fall} and UP_{fall} regarding e may be avoided. Let $UD(\theta)$ represent the smallest $l_i \in D_a$ such that $P_{fall}(a, l_i) \geq \theta$ and $LD(\theta)$ be the largest $l_i \in D_a$ with $P_{fall}(a, l_i) \leq \theta$. Clearly, for any $l \leq LD(\theta)$, we have $UP_{fall}(a, l) \leq \theta$ and $LP_{fall}(a, l) \geq \theta$ for any $l \geq UP(\theta)$.

Algorithm 2 Quick-Filtering(e, a, γ, θ)

Input: e : entry of R_S , a : anchor point
 γ : query distance, θ : probabilistic threshold

Output: *status* : { *validated, pruned, unknown* }

Description:

- 1: Compute $\delta_{max}(a, e_{mbb})$;
 - 2: **if** $\gamma - \delta_{max}(a, e_{mbb}) \geq UD(\theta)$ **then**
 - 3: **if** $\delta_{min}(a, e_{mbb}) + \gamma + \epsilon > UD(1 - \theta)$ **then**
 - 4: Return *pruned*;
 - 5: **else**
 - 6: Return *unknown*;
 - 7: **end if**
 - 8: **end if**
 - 9: **if** $\delta_{min}(a, e_{mbb}) - \gamma - \epsilon \leq LD(0)$ **then**
 - 10: **if** $\delta_{max}(a, e_{mbb}) < LD(\theta) - \gamma$ **then**
 - 11: Return *pruned*;
 - 12: **else**
 - 13: Return *unknown*;
 - 14: **end if**
 - 15: **end if**
 - 16: **if** $UP_{fall}(e_{mbb}, \gamma) < \theta$ **then**
 - 17: Return *pruned*;
 - 18: **else**
 - 19: Return *unknown*;
 - 20: **end if**
-

Algorithm 2 illustrates how to efficiently filter an entry based on an *anchor point*. In case 1 of Algorithm 2 (Line 2-8), $\gamma - \delta(a, e_{mbb}) \geq UD(\theta)$ implies $P_{fall}(e_{mbb}, \gamma) \geq \theta$ according to Theorem 3 and the definition of $UD(\theta)$. Therefore the entry e can be safely *validated*.

For case 2 (Line 9-15), $\delta_{max}(a, e_{mbb}) + \gamma \geq UD(1)$ implies $UP_{fall}(a, \delta_{max}(a, e_{mbb}) + \gamma) = 1.0$ ². We can *prune* e if $\delta_{min}(a, e_{mbb}) + \gamma + \epsilon > UD(1 - \theta)$ because this implies $UP_{fall}(e_{mbb}, \gamma) < \theta$. With similar rationale, the correctness of case 3 (Line 16-20) immediately follows³.

Once θ and γ are given for the query, we can pre-compute $UD(\theta)$, $LD(\theta)$, $LD(0)$ and $UD(1 - \theta)$ for each *anchor point*. Therefore, the time cost for Line 1-14 in Al-

2. Note that in case 2, we have $S2 = \emptyset$ in Fig. 10(b) where $S2 = \{x | x \in Q \text{ and } \delta(x, a) > \delta(a, p) + \gamma\}$. This implies that $P_{fall}(p, \gamma) \leq 1.0 - P_{fall}(a, \delta(a, p) - \gamma - \epsilon)$.

3. Note that in case 3, we have $S1 = \emptyset$ in Fig. 10(b) where $S1 = \{x | x \in Q \text{ and } \delta(x, a) < \delta(a, p) - \gamma - \epsilon\}$. This implies that $P_{fall}(p, \gamma) \leq P_{fall}(a, \delta(a, p) + \gamma) - 0$.

gorithm 2 is $O(d)$. We need to further apply Theorem 3 to compute $UP_{fall}(e_{mbb}, \gamma)$ in Line 15 if none of above three cases occurs. Consequently, the filtering time complexity of Algorithm 2 remains $O(d + \log n_d)$ for each *anchor point* in the worst case. Nevertheless, our experiment shows that new technique significantly improves the filtering time.

2. Accessing Order. In Algorithm 1, for a given entry e , we can apply Algorithm 2 for each *anchor point* $a \in AP$ until e is filtered by **any** *anchor point* or all *anchor points* fail to filter e . Intuitively, we should choose a good accessing order of the *anchor points* such that the entry e could be eliminated as early as possible. The ranking criteria for *validating* is simple, and *anchor points* close to e have high priority. As to the *pruning*, we prefer *anchor points* which are close to or far from e . This is because Line 6 of Algorithm 2 prefers the large $\delta_{min}(a, e_{mbb})$ while Line 11 is in favor of small $\delta_{max}(a, e_{mbb})$. Our experiment shows that a good accessing order can significantly reduce the filtering cost.

3. Sector Based Heuristic. Since we can immediately apply Algorithm 2 once the minimal and maximal distances between e and an *anchor point* are computed, it is less efficient to decide the accessing order after computing distances between all *anchor points* and e . Therefore we employ a simple heuristic to order *anchor points* without distance computations. We first consider the 2 dimensional case, and the heuristic can be easily extended to higher dimensions.

As shown in Fig. 11, suppose $C_{g_Q, R}$ is evenly partitioned into $n_s = 8$ sectors, labeled from 0 to 7 respectively. Recall that g_Q represents the *mean* of the uncertain query Q . Given a point p , we can find its corresponding sector by computing the angle α between vectors $\overrightarrow{g_Q p}$ and $\overrightarrow{g_Q x}$. Then p locates at $\frac{\alpha \times k_r}{2\pi}$ -th sector which is 5 in the example. The other sectors are sorted and the i -th sector accessed has subindex $s + (-1)^{i \frac{k_r}{2}} + n_s \bmod n_s$ for $2 \leq i \leq n_s$ where s indicates the subindex of the first sector visited, which is 5 in the example. Therefore, the accessing order of the sectors in the example is 5, 6, 4, 7, 3, 0, 2 and 1. Intuitively, *anchor points* in the sectors accessed earlier are likely to be closer to the point p . In the example, we visit *inner anchor points* with the following order: $a_1, a_2, a_3, a_4, a_5, a_6, a_7$ and a_8 .

As to *outer anchor points*, 8 sectors will be ordered exactly same as previous one. Nevertheless they will be accessed from head and tail at same time as shown in Fig. 11 because *anchor points* which are close to or far from p are preferred. Consequently, the accessing order of the *outer anchor points* is $a_9, a_{10}, a_{11}, a_{12}, a_{13}, a_{14}, a_{15}$ and a_{16} in the example. Our experiments confirm the effectiveness of this heuristic.

3.5 Discrete Case

Techniques proposed in this section can be immediately applied to *discrete* case. We abuse the notation of *pdf* and also use *pdf* to describe the uncertain query Q in *discrete* case; that is, $Q.pdf(x) = P_x$ if $x \in Q$ and

$Q.pdf(x) = 0$ if $x \notin Q$. $Q.region$ corresponds to the minimal bounding box of all instances of Q . Moreover, the statistic information of the uncertain query Q in *discrete* case is defined as follows: $g_Q = \sum_{q \in Q} q \times P_q$, $\eta_Q = \sum_{q \in Q} \delta(q, g_Q) \times P_q$ and $\sigma_Q = \sum_{q \in Q} \delta(q, g_Q)^2 \times P_q$. Then the techniques proposed in this section can be applied to *discrete* case.

4 EXPERIMENT

We present results of a comprehensive performance study to evaluate the efficiency and scalability of the techniques proposed in the paper. Following the framework of Algorithm 1 in Section 3, four different filtering techniques have been implemented and evaluated.

- MMD* The *maximal/minimal distance* filtering technique is employed as a benchmark to evaluate the efficiency of other filtering techniques.
- STF* The *statistical filtering* technique proposed in Section 3.2.
- PCR* The *PCR* technique discussed in Section 3.3. For the fairness of the comparison, we always choose a number of *PCRs* for the uncertain query such that space usage of *PCR* is same as that of *APF*.
- APF* The *anchor point filtering* technique proposed in Section 3.4.

The *Euclidean* distance is used in the experiments. All algorithms are implemented in C++ and compiled by GNU GCC. Experiments are conducted on PCs with Intel P4 2.8GZ CPU and 2G memory under Debian Linux.

Two real spatial datasets, *LB* and *US*, are employed as target data sets which contain 53k and 1M 2-dimensional points representing locations in the Long Beach country and the United State respectively⁴. All of the dimensions are normalized to domain $[0, 10000]$. To evaluate the performance of the algorithms, we also generate synthetic dataset *Uniform* with dimensionality 2 and 3 respectively, in which points are uniformly distributed. The domain size is $[0, 10000]$ for each dimension, and the number of the points varies from 1M to 5M. All of the datasets are indexed by aggregate *R* trees with page size 4096 bytes. *US* dataset is employed as the default dataset.

A workload consists of 200 uncertain queries in our experiment. The uncertain regions of the uncertain queries in our experiment are circles or spheres with radius q_r . q_r varies from 200 to 1000 with default value 600 which is 6% of the domain size. The centers of the queries are randomly generated within the domain. Recall that the domain of each dimension is normalized to $[0, 10000]$ for all datasets in our experiment. Two popular distributions, *Normal* and *Uniform*, are employed in our experiment to describe the PDF of the uncertain queries, where *Normal* serves as default distribution. Specifically, we use the *constrained normal* distribution such that the possible location of the uncertain query are restricted in the uncertain region of query Q . The standard deviation

4. Available at <http://www.census.gov/geo/www/tiger>

σ is set to $\frac{q_r}{2}$ by default. $10k$ sampled locations are chosen from the distributions⁵. For an uncertain query Q , instances of the *Normal* have the same appearance probabilities. Half of the instances of the *Uniform* have appearance probabilities $1/40000$, and another half have appearance probabilities $3/40000$. We assume instances of each query are maintained by an in memory aggregate R tree with fanout 8 where the aggregate probability of the child instances is kept for each entry. The query distance γ varies from 600 to 2000 with default value 1200. Note that queries in a workload share the same system parameters. In order to avoid favoring a particular θ value, we randomly choose the probabilistic threshold between 0 and 1 for each uncertain query.

According to the analysis of Section 3.1, the dominant cost of Algorithm 1 comes from the IO operation and verification. Therefore we measure the performance of the filtering techniques by means of IO cost and candidate size during the computation. The IO cost is the number of pages visited from R_P , and the candidate size is the number of points which need verification. In addition, we also evaluate the filtering time and the query response time.

In each experiment, we issue 200 uncertain queries and use the average candidate size, number of nodes accessed, filtering time and query response time to evaluate the performance of the techniques.

Table 2 below lists parameters which may potentially have an impact on our performance study. In our experiments, all parameters use default values unless otherwise specified.

Notation	Definition (Default Values)
q_r	the radius of the uncertain region (600)
σ	standard deviation for <i>Normal</i> distribution (300)
n_{ap}	the number of <i>anchor points</i> in <i>APF</i> (30)
n_d	the size of D_a for each <i>anchor point</i> (30)
γ	query distance (1200)
θ	probabilistic threshold ($\in [0, 1]$)
n	the number of data points in P (1m)

TABLE 2
System Parameters

4.1 Evaluate anchor point filtering technique

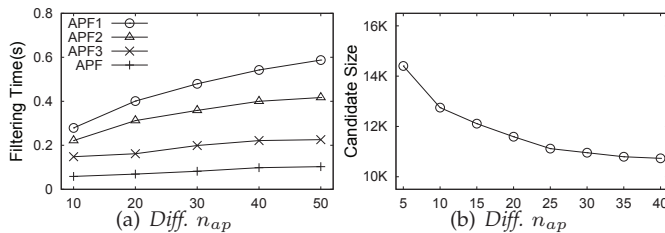


Fig. 12. Filter Performance Evaluation

In the first set of experiments, we evaluate the three techniques proposed for the *anchor point* filtering technique. All system parameters are set to default values.

5. As shown in [26], [28], the samples of the $Q.pdf$ are employed for the integral calculations. Therefore, we use the samples of the uncertain query to evaluate the performance of the algorithm for both *continuous* case and *discrete* case.

Let *APF1* represent the algorithm which computes the lower and upper probabilistic bounds for each entry based on Theorem 3. Algorithm 2 is applied in algorithm 3. *APF2*, *APF3* and *APF*. *APF2* always accesses *anchor points* in a random order, while *APF3* chooses a “good” order based on the distances between the entry and *anchor points*. The *sector* based heuristic is employed in *APF*. As expected, Fig. 12(a) demonstrates the effectiveness of the *APF*, which outperforms others and its filtering cost grows slowly against the number of *anchor points*. This implies the number of *anchor points* involved in the filtering process is less sensitive to n_{ap} for *APF*. Consequently, *APF* is used to represent the *anchor point* filtering technique in the rest of the experiments.

In the second set of experiments, we evaluate the impact of n_d (i.e., the number of different γ values pre-computed for each *anchor point*) against the performance of the *APF* technique based on *US* dataset. As expected, Fig. 12(b) shows that the candidate size decreases against the growth of n_d . We set $n_d = 30$ in the rest of the experiments unless otherwise specified.

Fig. 13 evaluates the effect of n_{ap} (i.e., the number of *anchor points*) against the filtering capacity of *APF* and *PCR*. Fig. 13(a) shows that *APF* outperforms *PCR* when there are not less than 5 *anchor points*. Moreover, the performance of the *APF* is significantly improved when the number of *anchor points* grows. Recall that we always choose a number of *PCRs* such that *PCR* uses the same amount of space as *APF*. However, there is only a small improvement for *PCR* when more space is available. Although the filtering cost of *APF* increases with the number of *anchor points* and it is always outperformed by *PCR* in terms of filtering cost as shown in Fig. 13(b), the cost is clearly paid off by the significant reduction of candidate size. This implies that *APF* can make a better trade-off between the filtering capacity and space usage. We set $n_{ap} = 30$ in the rest of the experiments unless otherwise specified.

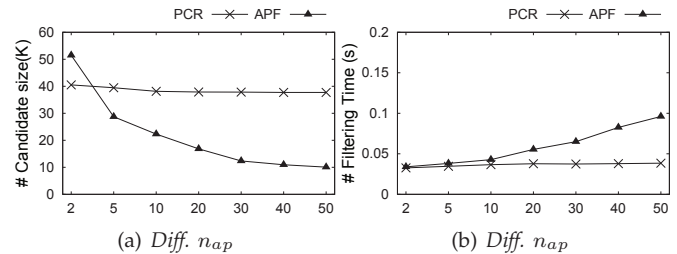


Fig. 13. Filter Performance vs Space usage

4.2 Performance Evaluation

In this subsection, we conduct comprehensive experiments to evaluate the effectiveness and efficiency of our filtering techniques proposed in the paper.

In the first set of experiments, we evaluate the impact of query distance γ on the performance of the filtering techniques in terms of candidate size, IO cost, query response time and filtering cost. All evaluations are conducted against *US* and 3d *Uniform* datasets.

Fig. 14 reports the candidate size of *MMD*, *STF*, *PCR* and *APF* when query distance γ grows from 400 to 2000. We set d_{ap} to 30 and 60 for *US* and 3d *Uniform* datasets respectively. As expected, larger γ value results in more candidate data points in the *verification* phase. It is interesting to note that with only a small amount of statistic information, *STF* can significantly reduce the candidate size compared with *MMD*. *PCR* can further reduce the candidate size while *APF* significantly outperforms others especially for the large γ : only 19650 data points need to be further verified for *APF* when $\gamma = 2000$ on *US* dataset, which is 96769, 183387 and 284136 for *PCR*, *STF* and *MMD* respectively.

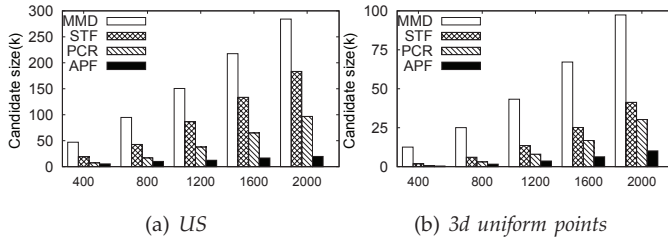


Fig. 14. Candidate Size vs γ

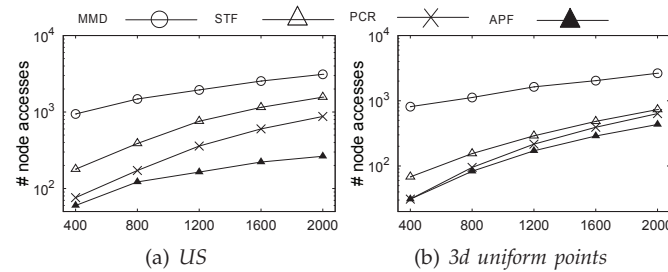


Fig. 15. # node accesses vs γ

Fig. 15 reports the IO cost of the techniques. As expected, *APF* is more IO efficient than other filtering techniques on both datasets.

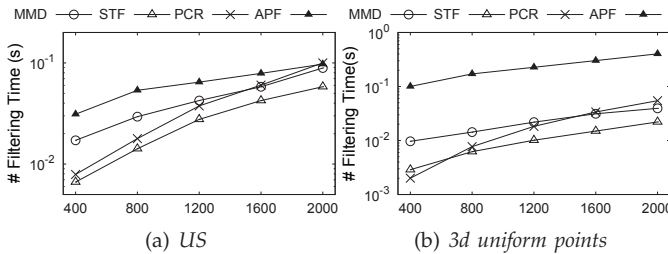


Fig. 16. Filtering Time vs γ

Fig. 16 reports the filtering time of four techniques against different query distance γ . It is shown that *STF* outperforms *MMD* under all settings. This is because both of them need at most two distance computations (i.e., maximal and minimal distance) for each entry, while the filtering capability of *STF* is much better than that of *MMD*. As expected, *APF* takes more time for filtering especially on 3-d *Uniform* dataset because more distance computations are required in *APF* and each costs $O(d)$ time. Nevertheless, it only takes less than 0.4 second for 3d *Uniform* dataset with $\gamma = 2000$, which is only a small portion of the computation cost as shown in Fig. 17.

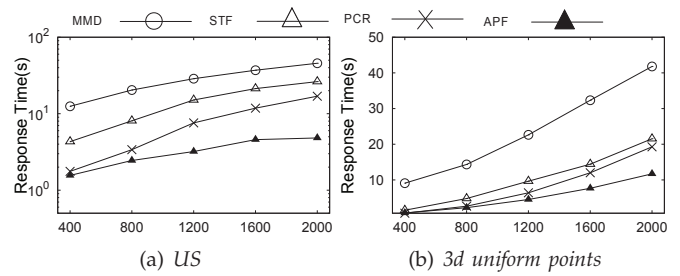


Fig. 17. Query Response Time vs γ

In Fig. 17, we report the average processing time of 200 uncertain queries. It is not surprising to see that *APF* outperforms other filtering techniques on the query response time evaluation because with reasonable filtering cost, *APF* has best filtering capability regarding candidate size and IO cost.

The second set of experiments depicted in Fig. 18 is conducted to investigate the impact of probabilistic threshold θ against the performance of the filtering techniques, where candidate size, the number of nodes assessed and the average query response time are recorded. The performance of *MMD* does not change with θ value on the candidate size and number of nodes accessed because the probabilistic threshold is not considered in *MMD* technique. Intuitively, the larger the threshold, there should be less entries which have a chance to get involved in the computation. This is confirmed by *PCR* and *APF* whose performances are better when θ is large, especially for *PCR*. It is interesting that *STF* has better performance when θ is small, even as good as *PCR* when $\theta = 0.1$.

We investigate the impact of q_r against the performance of the techniques in the third set of experiments. *LB* dataset is employed and uncertain queries follow *Uniform* distribution, while other system parameters are set to default values. Clearly the increment of q_r degrades the performance of all filtering techniques because more objects are involved in the computation. Nevertheless, as illustrated in Fig. 19 *PCR* and *APF* techniques are less sensitive to q_r .

In the last set of experiments, we study the scalability of the techniques against 2d *Uniform* dataset with size varying from 2M to 10M. As shown in Fig. 20, the performance of *PCR* and *APF* are quite scalable towards the size of dataset.

4.3 Summary

It has a much better performance than *MMD* with the help of a small amount of pre-computed statistic information. When more space is available, *PCR* and *APF* have better performance than *MMD* and *STF*. Compared with *PCR*, our experiments show *APF* can achieve a better trade-off between the filtering capability and the space usage.

5 DISCUSSIONS

Sum, min, max and average. Since the aggregate *R* tree is employed to organize the data points, instead of

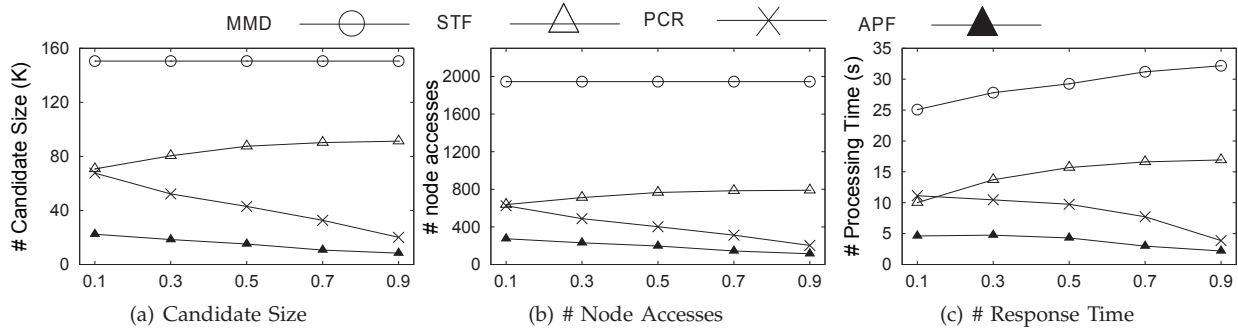


Fig. 18. Performance vs Diff. θ

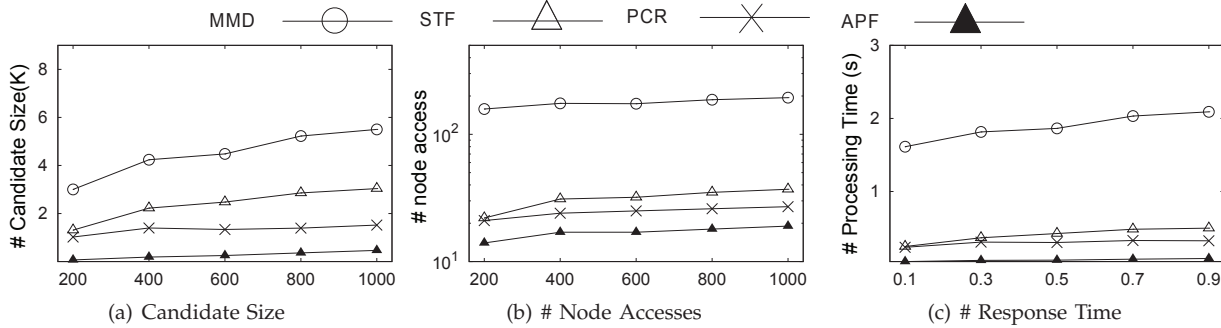


Fig. 19. Performance vs q_r (LB)

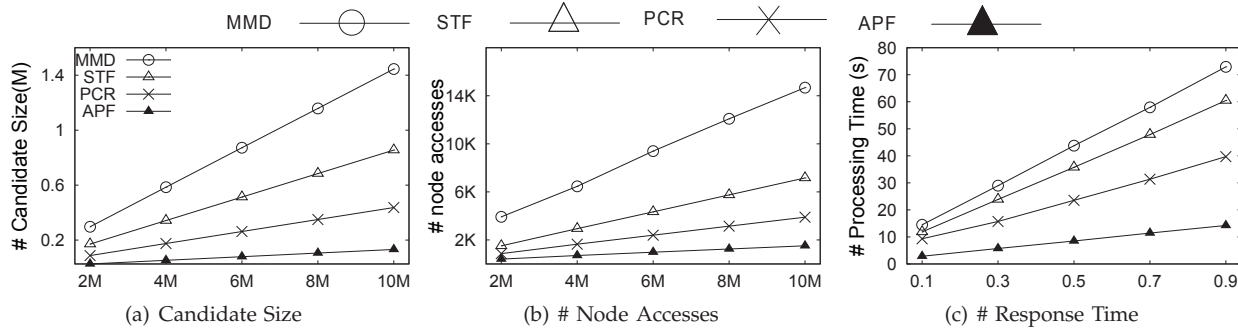


Fig. 20. Performance vs n

the number of descendant data points, other aggregate values like *sum*, *max* and *min* can be kept to calculate corresponding aggregate values. Note that for *average* we need to keep *count* and *sum* for each entry.

Uncertain query with non-zero absent probability. In some applications, the probability sum of the uncertain query might be less than 1 because the query might have a particular probability of being absent; that is, $\int_{x \in Q.region} Q.pdf(x)dx = c$ or $\sum_{q \in Q} P_q = c$ where $0 \leq c \leq 1$. The current techniques can be immediately applied by normalizing probabilities of the instances and the probabilistic threshold. For instances, suppose the uncertain query Q has two instances with appearance probability 0.3 and 0.2 respectively, and the probability threshold θ is set to 0.2. Then we can set the appearance probability of two instances to 0.6 and 0.4 respectively, and θ is set to 0.4.

Computing expected value. In some applications, instead of finding the number objects whose falling proba-

bilities exceed a given probabilistic threshold, users may want to compute the expected number of objects falling in Q regarding γ , denoted by n_{avg} ; that is,

$$n_{avg} = \sum_{p \in P} P_{fall}(p, \gamma) \quad (7)$$

Clearly, a point $p \in P$ can be immediately *pruned* (*validated*) if we can claim $P_{fall}(p, \gamma) = 0$ ($P_{fall}(p, \gamma) = 1$). Then, we need to compute the exact falling probabilities for all points survived, i.e., points not being *pruned* or *validated*. As a future direction, it is interesting to develop efficient algorithm to compute n_{avg} such that the computational cost can be significantly reduced.

Computing range aggregate against mutual exclusion. In the motivating application 1 of the paper, the actual number of civilian objects destroyed may be less than $|Q_{\theta, \gamma}(P)|$ because the events of objects to be destroyed may be mutually exclusive; that is, once the falling position of the missile is determined, two objects p_1 and p_2 cannot be destroyed at same time (e.g., when

$\delta(p_1, p_2) > 2\gamma$). Therefore, in addition to counting the number of civilian objects which are at risk, i.e., with falling probability at least θ , it is also interesting to find out the maximal possible number of objects being destroyed with probability at least θ . Although the algorithm in the paper can provide an upper bound for this problem, the techniques developed cannot be trivially applied for the exact solution because it is computational expensive to check the mutually exclusive property for each pair of objects in the candidate set. Therefore, as a future direction, it is desirable to develop novel techniques to efficiently solve this problem.

6 RELATED WORK

To the best of our knowledge, this is the first work on the problem of computing range aggregates against uncertain location based query. Our study is related to the previous work on querying uncertain objects [13], [21], [30], [8], [10], [1], [26], [28], [6], [29], [12].

Cheng *et al* present a broad classification of probabilistic queries over one-dimensional uncertain data as well as techniques for evaluating probabilistic queries [8]. There are four types in all, and the problem we studied in the paper belongs to the value-based aggregates. In recent years, *probability-thresholding range queries* has attracted much attention of the literature because of the importance of the problem. For the given region r_q and probabilistic threshold θ , such a query returns all objects which appear in r_q with likelihood at least θ . Based on the notation of *x-bounds*, a novel index, *probability threshold index (PTI)*, is proposed in [10] to efficiently process the query against one dimension uncertain objects. Recently, Agarwal *et al* [1] develop novel indexing technique for one dimension uncertain object to support range query with theoretical guarantee.

As a general version of the *x-bounds*, the concept of *probabilistic constrained region (PCR)* is introduced by Tao *et al* [26] to support the range query on uncertain objects in a multi-dimensional space where the pdf of the uncertain object might be an arbitrary function. In order to prune or validate a set of uncertain objects at the same time, the *U-Tree* technique is proposed to index the uncertain objects based on *PCR* technique. Tao *et al* further improve the *PCR* technique in [28] and they also study another range query problem in which locations of query and objects are uncertain. A similar problem is studied by Chen *et al* [6] where the *PCR* technique is also applied to efficiently *prune* or *validate* data points or uncertain objects. Recently, Yang *et al* [29] investigate the problem of range aggregate query processing over uncertain data in which two sampling approaches are proposed to estimate the aggregate values for the range query.

In [12], Dai *et al* investigate the probabilistic range query on existentially uncertain data in which each object exists with a particular probability. [3], [4] study range queries with the constraint that instances of uncertain objects follow *Gaussian* distribution. Results are ranked according to the probability of satisfying range

queries. A more recent work addressing indexing high dimensional uncertain data is [2].

Besides the range query over uncertain data, many other conventional queries are studied in the context of uncertain data such as clustering [17], [20], similarity join [16], [9], top-k queries [25], [14], nearest neighbour queries [7] and skyline query [23], [18]. Among the previous work on uncertain data, [28], [6] and [15] are the most closely related to the problem we studied. However, the target objects considered in [28] are uncertain objects which are organized by *U-tree*, so their algorithms cannot be directly applied to the problem studied in the paper. Our problem definition is similar with the *C-IPQ* problem studied in [6], but their query search region is rectangle which might be less interesting to the applications mentioned in our paper. Moreover, we investigate the range aggregates computation in our paper which is different from [28], [6]. We also show that although the *PCR* technique employed by [28], [6] can be modified to our problem as discussed in the paper, it is less efficient compared with *anchor point* filtering technique proposed in the paper. Although [15] studies the range query in which the location of the query is imprecise, their techniques are based on the assumption that the possible locations of the query follow the *Gaussian* distribution, hence cannot support the general case. Therefore, their techniques cannot be applied to the problem studied in the paper.

7 CONCLUSION

In this paper, we formally define the problem of uncertain location based range aggregate query in a multi-dimensional space; it covers a wide spectrum of applications. To efficiently process such a query, we propose a general filtering and verification framework and two novel filtering techniques, named *STF* and *APF* respectively, such that the expensive computation and *IO* cost for verification can be significantly reduced. Our experiments convincingly demonstrate the effectiveness and efficiency of our techniques.

Acknowledgement. The work was supported by ARC Grant (DP0987557, DP0881035 and DP0666428) and Google Research Award. Yufei Tao was supported by GRF grants 4166/10 and 4169/09 from HKRGC.

REFERENCES

- [1] P. K. Agarwal, S.-W. Cheng, Y. Tao, and K. Yi. Indexing uncertain data. In *Proc. Symp. Principles of Database Systems (PODS)*, 2009.
- [2] C. Aggarwal and P. Yu. On high dimensional indexing of uncertain data. In *Proc. Intl Conf. Data Eng. (ICDE)*, 2008.
- [3] C. Bohm, M. Gruber, P. Kunath, A. Pryakhin, and M. Schubert. Prover: Probabilistic video retrieval using the gauss-tree. In *Proc. Intl Conf. Data Eng. (ICDE)*, 2007.
- [4] C. Bohm, A. Pryakhin, and M. Schubert. Probabilistic ranking queries on gaussians. In *Proc. Intl Conf. Scientific and Statistical Database Management (SSDBM)*, 2006.
- [5] V. Bryant. *Metric Spaces: Iteration and Application*. Cambridge University Press, 1996.
- [6] J. Chen and R. Cheng. Efficient evaluation of imprecise location-dependent queries. In *Proc. Intl Conf. Data Eng. (ICDE)*, 2007.

- [7] R. Cheng, J. Chen, M. F. Mokbel, and C.-Y. Chow. Probabilistic verifiers: Evaluating constrained nearest-neighbor queries over uncertain data. In *Proc. Intl Conf. Data Eng. (ICDE)*, 2008.
- [8] R. Cheng, D. V. Kalashnikov, and S. Prabhakar. Evaluating probabilistic queries over imprecise data. In *Proc. ACM SIGMOD*, 2003.
- [9] R. Cheng, S. Singh, and S. Prabhakar. Efficient join processing over uncertain data. In *Proc. Intl Conf. Information and Knowledge Management (CIKM)*, 2006.
- [10] R. Cheng, Y. Xia, S. Prabhakar, R. Shah, and J. S. Vitter. Efficient indexing methods for probabilistic threshold queries over uncertain data. In *Proc. Intl Conf. Very Large Data Bases (VLDB)*, 2004.
- [11] G. W. Cordner. Police patrol work load studies: A review and critique. *Police Studies*, 2(3):50–60, 1979.
- [12] X. Dai, M. Yiu, N. Mamoulis, Y. Tao, and M. Vaitis. Probabilistic spatial queries on existentially uncertain data. In *Proc. Intl Symp. Large Spatio-Temporal Databases (SSTD)*, 2005.
- [13] E. Frentzos, K. Gratsias, and Y. Theodoridis. On the effect of location uncertainty in spatial querying. *IEEE Trans. Knowl. Data Eng.*, 21(3):366–383, 2009.
- [14] M. Hua, J. Pei, W. Zhang, and X. Lin. Ranking queries on uncertain data: A probabilistic threshold approach. In *Proc. ACM SIGMOD*, 2008.
- [15] Y. Ishikawa, Y. Iijima, and J. X. Yu. Spatial range querying for gaussian-based imprecise query objects. In *Proc. Intl Conf. Data Eng. (ICDE)*, 2009.
- [16] H.-P. Kriegel, P. Kunath, M. Pfeifle, and M. Renz. Probabilistic similarity join on uncertain data. In *Proc. Intl Conf. Database Systems for Advanced Applications (DASFAA)*, 2006.
- [17] H. P. Kriegel and M. Pfeifle. Density-based clustering of uncertain data. In *Proc. ACM SIGKDD*, 2005.
- [18] X. Lian and L. Chen. Monochromatic and bichromatic reverse skyline search over uncertain databases. In *Proc. ACM SIGMOD*, 2008.
- [19] R. Meester. *A Natural Introduction to Probability Theory*. Addison Wesley, 2004.
- [20] W. K. Ngai, B. Kao, C. K. Chui, R. Cheng, M. Chau, and K. Y. Yip. Efficient clustering of uncertain data. In *Proc. Intl Conf. on Data Mining (ICDM)*, 2006.
- [21] J. Ni, C. V. Ravishankar, and B. Bhanu. Probabilistic spatial database operations. In *Proc. Intl Symp. Large Spatio-Temporal Databases (SSTD)*, 2003.
- [22] D. Papadias, P. Kalnis, J. Zhang, and Y. Tao. Efficient olap operations in spatial data warehouses. In *Proc. Intl Symp. Large Spatio-Temporal Databases (SSTD)*, 2001.
- [23] J. Pei, B. Jiang, X. Lin, and Y. Yuan. Probabilistic skyline on uncertain data. In *Proc. Intl Conf. Very Large Data Bases (VLDB)*, 2007.
- [24] G. M. Siouris. *Missile Guidance and Control Systems*. Springer Publication, 2004.
- [25] M. A. Soliman, I. F. Ilyas, and K. C. Chang. Top-k query processing in uncertain databases. In *Proc. Intl Conf. Data Eng. (ICDE)*, 2007.
- [26] 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. Intl Conf. Very Large Data Bases (VLDB)*, 2005.
- [27] Y. Tao and D. Papadias. Range aggregate processing in spatial databases. *IEEE Trans. Knowl. Data Eng.*, 16(12):1555–1570, 2004.
- [28] Y. Tao, X. Xiao, and R. Cheng. Range search on multidimensional uncertain data. *ACM Trans. Database Syst.*, 32(3), 2007.
- [29] S. Yang, W. Zhang, Y. Zhang, and X. Lin. Probabilistic threshold range aggregate query processing over uncertain data. In *Advances in Data and Web Management, Joint International Conferences (APWeb/WAIM)*, 2009.
- [30] X. Yu and S. Mehrotra. Capturing uncertainty in spatial queries over imprecise data. In *Proc. Intl Conf. Database and Expert Systems Applications (DEXA)*, 2003.



Ying Zhang is a Research Fellow in the School of Computer Science and Engineering, the University of New South Wales. He received his BSc and MSc degrees in Computer Science from Peking University, and PhD in Computer Science from the University of New South Wales. His research interests include query processing on data stream, uncertain data and graphs.



Xuemin Lin is a Professor in the School of Computer Science and Engineering, the University of New South Wales. He has been the head of database research group at UNSW since 2002. Before joining UNSW, Xuemin held various academic positions at the University of Queensland and the University of Western Australia. Dr. Lin got his PhD in Computer Science from the University of Queensland in 1992 and his BSc in Applied Math from Fudan University in 1984. During 1984–1988, he studied for PhD in Applied Math at Fudan University. He currently is an associate editor of ACM Transactions on Database Systems. His current research interests lie in data streams, approximate query processing, spatial data analysis, and graph visualization.



Yufei Tao is a Professor at the Department of Computer Science and Engineering, Chinese University of Hong Kong (CUHK). Before joining CUHK in 2006, he was a Visiting Scientist at the Carnegie Mellon University during 2002–2003, and an Assistant Professor at the City University of Hong Kong during 2003–2006. He received a Hong Kong Young Scientist Award 2002 from the Hong Kong Institution of Science. He is currently an associate editor of ACM Transactions on Database Systems (TODS). His research interests focus on algorithms and data structures on massive datasets.



Wenjie Zhang is a research fellow in the School of Computer Science and Engineering, the University of New South Wales, Australia. She received her M.S. degree and B.S. degree both in computer science from Harbin Institute of Technology, China. Her research focuses on uncertain data management and spatio-temporal indexing techniques. She has published papers in conferences and journals including SIGMOD, VLDB, ICDE, VLDB J and TKDE.



Haixun Wang is a researcher at Microsoft Research Asia in Beijing, China. Before joining Microsoft, he had been a research staff member at IBM T. J. Watson Research Center for 9 years. He was Technical Assistant to Stuart Feldman (Vice President of Computer Science of IBM Research) from 2006 to 2007, and Technical Assistant to Mark Wegman (Head of Computer Science of IBM Research) from 2007 to 2009. He received the B.S. and the M.S. degree, both in computer science, from Shanghai Jiao Tong University in 1994 and 1996. He received the Ph.D. degree in computer science from the University of California, Los Angeles in 2000. His main research interest is database language and systems, data mining, and information retrieval. He has published more than 120 research papers in referred international journals and conference proceedings. He was PC Vice Chair of KDD10, ICDM09, SDM08, and KDD08, Demo PC Chair of ICDE09 and ICDM08, Sponsor Chair of SIGMOD08, etc., and he serves on the editorial board of IEEE Transactions of Knowledge and Data Engineering (TKDE), Journal of Computer Science and Technology (JCST), as well as in program committees of various international conferences and workshops in the database field.