Probabilistic Threshold Range Aggregate Query Processing over Uncertain Data

Shuxiang Yang*, Wenjie Zhang+, Ying Zhang*, and Xuemin Lin+

*The University of New South Wales, Australia
+ The University of New South Wales & NICTA, Australia
{syang, zhangw, yingz, lxue}@cse.unsw.edu.au

Abstract. Large amount of uncertain data is inherent in many novel and important applications such as sensor data analysis and mobile data management. A probabilistic threshold range aggregate (PTRA) query retrieves summarized information about the uncertain objects satisfying a range query, with respect to a given probability threshold. This paper is the first one to address this important type of query. We develop a new index structure aU-tree and propose an exact querying algorithm based on aU-tree. For the pursue of efficiency, two techniques *SingleSample* and *DoubleSample* are developed. Both techniques provide approximate answers to a PTRA query with accuracy guarantee. Experimental study demonstrates the efficiency and effectiveness of our proposed methods.

1 Introduction

Many emerging important applications involve dealing with uncertain data, such as data integration, sensor data analysis, market surveillance, trends prediction, mobile data management, etc. Uncertainty is inherent in such data due to various factors like randomness or incompleteness of data, limitations of equipment and delay or loss in data transfer. Extensive research effort has been given to model and query uncertain data recently. Research directions include modeling uncertainty [16], query evaluation [2], indexing [18], top-k queries [8], skyline queries [15], clustering and Mining [10], etc. However, though range aggregate query on uncertain data is very important in practice, this problem remains unexplored.

A range aggregate query (RA query) on certain data returns summarized information about objects satisfying a given query range, such as the total number of qualified objects [19]. This type of query is important since users may be interested only in aggregate information instead of specific IDs. For instance, to monitor traffic volume of a crossroad A in rush hours, query "how many vehicles pass A from 8AM to 9AM today" is of more interest than "which vehicles pass A from 8AM to 9AM today". aR-tree [13] is the most popular index structure to answer RA query on spatial space.

While many sophisticated techniques have been developed to answer RA query over certain data [19], the counter problem of RA query over uncertain data has not attracted much research attention. Modeling and answering RA query over uncertain data require comprehensive analysis of probabilities, as shown in Figure 1. Assume we still use aR-tree to index the uncertain objects. N_i $(1 \leq i \leq 3)$ represents nodes in aR-tree and U_j $(1 \leq j \leq 9)$ represents uncertain objects. Each uncertain object contains a set of uncertain instances. A probabilistic threshold range query q fully covers node N_2 and intersects with node N_3. After accessing node N_3, only part of the instances in uncertain object U_7 is in the query range of q. Clearly in this case we have to retrieve the instances of U_7 to compute the probability for it to satisfy q. If this probability is no less than a probability threshold, then U_7 will be counted in the reulst otherwise it will be excluded from final result. Computing such probability is time consuming when the number of uncertain instances from uncertain objects is large.

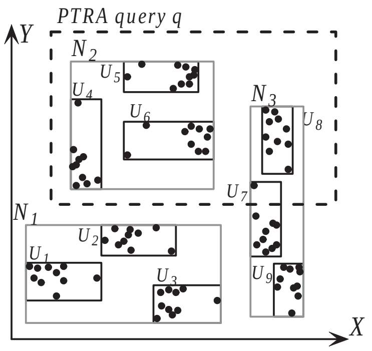


Fig. 1. Uncertain Objects

Challenges and Contributions. Aggregate information retrieval on uncertain objects requires detailed analysis of appearance probabilities from uncertain instances (*discrete case*) or probability density function (PDF) (*continuous case*). Naively computing the probability for every uncertain object to satisfy a $PTRA$ query can be very time consuming. Our contributions in this paper are:

– We formally define $PTRA$ query over uncertain objects.
– A novel index structure, aU-tree is developed to support exactly execut $PTRA$ queries.
– Two techniques, $SingleSample$ and $DoubleSample$ are proposed to approximately answer $PTRA$ queries with accuracy guarantee.
– An extensive experimental study over real and synthetic datasets shows the efficiency and effectiveness of our proposed techniques.

Organization of the paper. The rest of the paper is organized as follows. Section 2 models the problem and introduces preliminaries. Exact and approximate query processing techniques are presented in Section 3 and Section 4, respectively. In Section 5, the efficiency and effectiveness of our proposed techniques are experimentally studied. This is followed by related work in Section 6. We conclude our paper in section 7.

2 Background Information

In this section, we give a formal definition of PTRA query for *discrete* cases. The extension of problem definition and techniques to *continuous* cases will be discussed in the end of this paper in Section 7.

2.1 Problem Statement

A PTRA query q is related with a d-dimensional query range r_q and a probability threshold p_q. In discrete cases, an uncertain object U is represented by a set of uncertain instances $\{u_1, ..., u_l\}$. Each instance u_i is associated with a membership probability $P(u_i)$ and $\sum_{i=1}^{l} P(u_i) = 1$. Let $P_{app}(U, q)$ be the appearance probability that object U satisfies a query with range r_q, then

$$P_{app}(U, q) = \sum_{u \in U, u \vdash r_q} P(u) \tag{1}$$

where $u \vdash r_q$ denotes uncertain instance u is inside r_q.

Given a set of uncertain objects $\mathcal{U}$, a PTRA query q returns the number of uncertain objects in $\mathcal{U}$ with appearance probability no less than p_q:

$$|\{U \in \mathcal{U} | P_{app}(U, q) \geq p_q\}| \tag{2}$$

Example 1. As the example illustrated in Figure 1, to process PTRA query q, we calculate the appearance probability of $P(U_7, q)$ using the formulate 1. If $P(U_7, q) \geq p_q$, then the result for q is 5; otherwise, the result is 4.

2.2 Preliminaries

Possible World Semantics. Given a set of uncertain objects $\mathcal{U} = \{U_1, \cdots, U_n\}$, a *possible world* $W = \{u_1, \cdots, u_n\}$ is a set of n instances - one instance per uncertain object. The probability of W to appear is $P(W) = \prod_{i=1}^{n} P(u_i)$. Let Ω be the set of all possible worlds; that is, $\Omega = U_1 \times U_2 \cdots \times U_n$. Then, $\sum_{W \in \Omega} P(W) = 1$. Namely, Ω enumerates all the possibilities of $\mathcal{U}$.
aR-tree based Range Query Processing. aR-tree is a modification of R-tree [7] by storing the number of objects in each entry. Figure 2 illustrates the structure of 2-level aR-tree. Besides R-tree structure information, entries in root node also keep the number of objects contained, such as entry $e_1 \in root$ contains 3 objects. The dashed rectangle is the range of a RA query q. As shown in Figure 2, leaf node N_3 is fully covered by q, so N_3 will not be accessed but adding 3 to the final result. Node N_1 does not need to be accessed either since it does not intersect with q. The only accessed node is N_2 and object O_5 is detected to satisfy q. So the final result is 4.
U-tree. One of the most popular index structure for multi-dimensional uncertain data with arbitrary PDFs is U-tree [18], which is built based on R*-tree

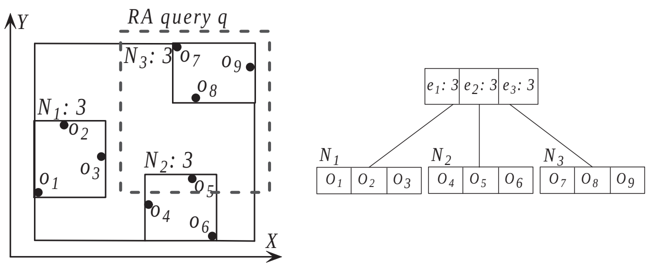


Fig. 2. Certain Objects Indexed by aR-tree

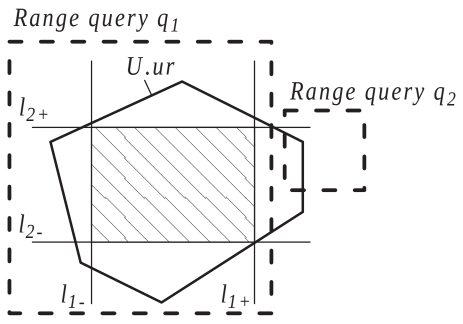


Fig. 3. Pruning/Validating in U-tree

with a set of pruning and validating rules to support range queries over uncertain data. In Figure 3, polygon $U.ur$ is the uncertain region of 2-dimensional object U. For a given probability $p = 0.2$, in the horizontal dimension, two lines are calculated. U has probability p to occur on the left side of line l_{1-}, also probability p to occur on the right side of line l_{1+}. In the vertical dimension, two such lines are also computed according to the PDF of U. The intersection of these four lines is called probabilistically constrained regions (PCRs). Suppose probability threshold of range query q_1 is 0.8, U can be validated without accessing the instances of U since it fully contains PCR(p = 0.2). On the other hand, suppose probability threshold of range query q_2 is 0.2, U will be pruned from the result of q_2 because q_2 does not intersect with PCR(p = 0.2). To trade-off between pruning/validating power and space cost, only a set of m probability values are chosen as representatives to compute their PCRs. These m PCRs are further bounded from the "outside" and from "inside", called $o.cfb_{out}$ and $o.cfb_{in}$, respectively. A U-tree is built by organizing the cfb_{out} and cfb_{in} of uncertain objects.

Min-Skew Partitioning. Min-Skew partitioning skill was proposed by [1] aiming at dividing the data space into a number of buckets according to the spatial distribution of input data points. Two metrics are proposed in [1] to capture the underlying feature of the input data distribution: *spatial density* of a point representing the number of rectangles that include the point; *spatial-skew* of a bucket which is the statistical variance of the spatial densities of all points grouped in that bucket. The partitioning procedure is: use a uniform grid of regions with the spatial density in each grid to represent the spatial density of the input data. The process starts from a single bucket including the whole space. Split is processed along the boundary of the grid which will lead to the largest reduce of spatial skewness. The iteration stops when the number of buckets meets users' specification.

3 Exact Query Processing

In this section, we firstly present aU-tree which is modified based on U-tree by integrating aggregate information; this is followed by the exact query processing algorithm based on aU-tree.

3.1 aU-tree

Similar with the adjustment of aR-tree to R-tree for the RA query over certain data, aU-tree is modified on U-tree by embedding aggregate information in every entry. Updating the aggregate information for aU-tree is similar as for aR-tree: whenever inserting or deleting an object, the aggregate information on entries along the corresponding insertion/deletion path is updated as well.

Specifically, each intermediate entry in an aU-tree keeps the following information: a pointer referencing its child node; two d-dimensional rectangles which are used for pruning as introduced in Section 2; *agg* which is the number of uncertain objects indexed under the subtree rooted at this intermediate entry. Each leaf entry records the following: conservative functional boxes for uncertain object U for both pruning and validating as introduced in Section 2; MBR of the uncertain region of U; uncertain region $U.ur$; and a set of instances to describe the probability distribution. For a leaf entry, its *agg* value is assigned to 1.

3.2 Querying Algorithm

With aU-tree, if the intermediate entry is totally covered by the query range, the aggregate result for the subtree underneath can be retrieved immediately without accessing every uncertain object in it. However, given a probability threshold, U-tree can only be used to prune a subtree in the intermediate level but not validate a subtree unless the query range fully covers it. In such cases, we still have to access the children of the intermediate entry. Algorithm 1 illustrates the steps of exact results retrieval based on aU-tree.

Till here, to process a PTRA query, we need to reach the leaf level if an intermediate entry can not be either pruned or fully covered by the query range. Then on the leaf level, the pruning/validating rules of U-tree are applied on individual uncertain object. If an object can not be pruned or validated, its exact appearance probability in r_q will be computed. Two aspects impede the efficiency of the exact algorithm. Firstly, validating on higher levels in aU-tree is not possible and the pruning power is not powerful enough especially when

Algorithm 1 Exact Query Processing

INPUT: root node N of aU-tree;
probabilistic query q with probability threshold p_q; query range r_q;
OUTPUT: $result$:number of objects inside r_q with probability $\geq p_q$;
Description:

```
 1: if $r_q$ fully covers $N$ then
 2:   $result$ += $N.agg$;
 3: if $r_q$ partially overlaps with $N$ then
 4:   if $N$ is an intermediate node then
 5:     if $N$ can not be pruned w.r.t $p_q$ then
 6:       for each child $child$ of $N$ do
 7:         call Algorithm 1 with $child$ and $q$ as input;
 8:   if $N$ is a leaf node then
 9:     apply PCR technique of U-tree on $N$
10:     if $N$ is validated then
11:       $result$ += 1;
12:     if $N$ is neither pruned nor validated then
13:       compute the exact probability of $P_{app}(N, q)$;
14:       if $P_{app}(N, q) \geq p_q$ then
15:         $result$ += 1;
16: return result;
```

index space is limited; Secondly, computing the appearance probability of an uncertain object is time expensive when the number of instances is large. In the next section, we develop approximate query processing algorithms which are both efficient in time and effective in accuracy.

4 Approximate Query Processing

For a set of uncertain objects $\mathcal{U} = \{U_1, ..., U_n\}$, a possible world consists of n sampled instances – one instance from one uncertain object. Suppose the number of uncertain instance in an uncertain object U_i is $|U_i|$ ($1 \leq i \leq n$), the total number of possible worlds is $\Pi_{i=1}^{n} |U_i|$. This number can be huge when n is large and each uncertain object is represented by a large number of instances. The basic idea of our approximate algorithm is to sample all possible worlds using a small number m of possible worlds S_i ($1 \leq i \leq m$) where each S_i also contains n instances – one per object. Intuitively, for an uncertain object U, if its sampled instance is inside r_q in m' out of m corresponding sampled possible worlds, $P_{app}(U, q)$ can be approximated by m'/m. If $m'/m \geq p_q$, then U is considered a result contributed to q.

We consider two scenarios in this section. The first one is for the case where the number of uncertain objects is relatively small, we sample the possible worlds only. Techniques developed are called *SingleSample*. The second one is deployed when the number of uncertain objects is also large. It will become too time-consuming to get an approximate answer with decent accuracy performance. In this case, in the first step we sample a small number of uncertain objects $\mathcal{U}_s$, in the second step sampling possible worlds is applied on $\mathcal{U}_s$. The technique is thus

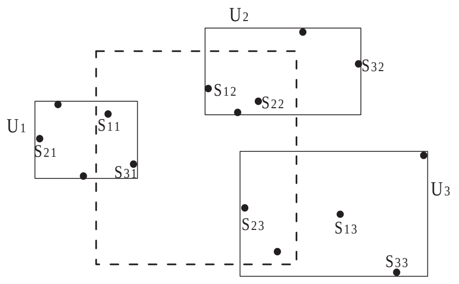


Fig. 4. aU-tree indexing uncertain data.

named *DoubleSample*. Query processing algorithms and accuracy guarantee are presented for both techniques.

4.1 SingleSample

In this section, an aU-tree is built on the MBR of sampling instances from all uncertain objects, as illustrated in Figure 4. In Figure 4, each uncertain object is represented using 5 sampling instances. $S_{i,j}$ represents the sampling instance from the i-th sample worlds $(1 \leq i \leq m)$ and the j-th uncertain object $(1 \leq j \leq n)$.

Approximate Query Algorithm Theoretically, for each sampled possible world S_i $(1 \leq i \leq m)$, we process a PTRA query and record for every uncertain object U_j $(1 \leq j \leq n)$ whose sample instance $S_{i,j}$ is inside r_q. Noticing the fact that $S_{i,j}$ is inside r_q is not affected by the sample instances from the other possible worlds or other uncertain objects, a more efficient query algorithm is developed in this section. As shown in Figure 4, two sampled instances from U_1 are inside the query region, so $P_{app}(U_1, q)$ is approximated by 2/3; similarly for U_2 and U_3, the approximated probability is 2/3 and 1/3, respectively. If the given probability threshold is 1/2, then result for this PTRA is 2 (U_1 and U_2). Algorithm detail is described in Algorithm 2.

Accuracy Guarantee Theorem 1 presents the accuracy guarantee for Algorithm 2.

Theorem 1. *Suppose $s_1 = \frac{1}{\epsilon_1^2} log \delta_1^{-1}$ sampling instances are drawn for each uncertain object, then the following inequation holds with probability at least 1 - δ_1,*

$$|P^E - P| \leq \epsilon_1 \tag{3}$$

where P is the actual appearance probability of the uncertain object and P^E is the probability computed using Algorithm 2.

The theorem can be proved using the Chernoff Hoeffding bound. Proof details are omitted due to space limitation.

Algorithm 2 Query Processing for SingleSample

INPUT: aU-tree indexing the MBR of all uncertain objects;
probabilistic query q with probability threshold p_q; query range r_q;
OUTPUT: $result$:number of objects inside r_q with probability $\geq p_q$;
Description:

```
1: sample $m$ possible worlds from all possible worlds;
2: apply pruning/validating techniques on aU-tree as in Algorithm 1;
3: if a node $N$ is validated then
4:     result += $N.agg$;
5: for every uncertain object $U_i$ that can not be pruned/validated do
6:     record $m'$ as number of sampled instance inside $r_q$;
7:     if $m'/m \geq p_q$ then
8:         result += 1;
9: return result;
```

4.2 DoubleSample

When the number of uncertain objects is huge, the aU-tree in Algorithm 2 may take a large space which prevents the range query from efficiently processing. In this case, we propose a solution to get a sample set of uncertain objects $\mathcal{U}_s$ first, and then sample the possible worlds based on $\mathcal{U}_s$.

Sample Uncertain Objects A naive way to select uncertain objects is to use uniform sampling; however, this may lead to lose of spatial distribution of uncertain objects. Instead, we utilize Min-Skew partitioning technique to select K nodes from the aU-tree indexing the MBR of uncertain objects. K can be a user-specific parameter to meet with the space requirements. We call the selected K nodes best K nodes (BKNs).

The criteria to select BKNs is that the sum of the spatial skewness of the K subtrees is as small as possible meanwhile cover all uncertain objects. To do this, we propose an efficient heuristic. In the first step, identify from aU-tree a level L which has intermediate nodes less than K and the number of intermediate nodes on its child level $L-1$ is larger than K. (Note that if there exists a level L with exactly K nodes then we simply choose them as the K nodes.) Otherwise, each node N_i on level L is split into G_i buckets, making $\sum_{i=1}^{r} G_i = K$, where r is the number of nodes on level L. G_i $(1 \leq i \leq r)$ is computed according to the spatial skewness of each node. For two nodes N_i and N_j, $G_i : G_j = Sk_i : Sk_j$, where Sk represents the skewness of a node. After getting the BKNs, we perform uniform sampling on each selected node and obtain the sampled objects $\mathcal{U}_s$.

Sampling possible worlds is processed the same as in *SingleSample* technique. The difference is we form sample worlds based on $\mathcal{U}_s$ instead of $\mathcal{U}$.

Approximate Query Algorithm
Based on $\mathcal{U}_s$, we process Algorithm 2 which returns $result$. Since we apply sampling twice in DoubleSample, the final result for *DoubleSample* is therefore:

$$result(DoubleSample) = result * \frac{|\mathcal{U}|}{|\mathcal{U}_s|} \tag{4}$$

Accuracy Guarantee
The following theorem states the accuracy guarantee of DoubleSample technique.

Theorem 2. *Let A be number of objects with appearance probability $P_x \geq p_q$. Suppose $s_2 = \frac{1-\delta_1}{\epsilon_2^2} \log \delta_2^{-1}$ sampling objects are drawn. s_1 sampled instances are generated for each sampled uncertain object. Assuming the appearance probability of each uncertain object follows uniform distribution regarding the query, then the following inequality holds with probability at least 1 - δ_2:*

$$|A^E - A| \leq (\epsilon_1 + \epsilon_2) * N \tag{5}$$

where A^E is the estimated value of A and N is the total number of uncertain objects. s_1, ϵ_1 and δ_1 are the same as in Theorem 1.

This theorem can also be proved using Chernoff Hoeffding bound. Limited by space, proof details are omitted.

5 Experimental Analysis

All algorithms are implemented in C++. Experiments are run on *PCs* with Intel P4 2.8GHz CPU and 2G memory under Debian Linux. The page size is fixed to 8192 bytes.

Two real spatial data sets are used in this section. LB with $53K$ points and CA with $62K$ points, presenting locations in the Long Beach country and California. Data domain along each dimension is $[0, 10000]$. An data U is generated with uncertainty region as a circle with radius rad_U 250. For each uncertain object U, 10000 instances are generated and the spatial distribution follows either *uniform* or *Constrained-Gaussian* (*Con-Gau*) distribution. A synthetic Aircraft data set consisting of $53K$ points is also generated to investigate the performance in $3D$ space with instances in *Con-Gau* distribution. All data sets are downloaded from *http://www.cse.cuhk.edu.hk/ taoyf/paper/tods07-utree.html*.

The query region r_q is a square/cube with radius rad_q ranging from 500 to 1500. The probabilistic threshold p_q is ranged from 0.3 to 0.9.

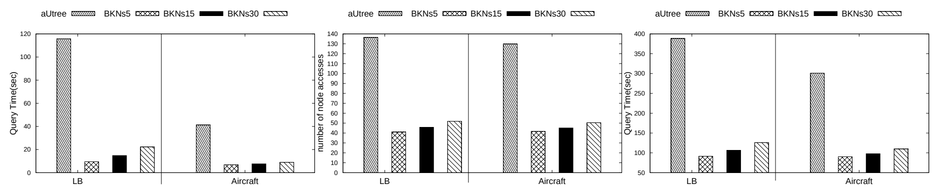
For approximate querying algorithms, the number of sampling instances varies among $[10, 1000, 1000]$. We denote $BKNs$ as the number of buckets in *DoubleSample* technique. $BKNs$ ranges from 5 to 30 and each bucket keeps 250 sampled objects. The default value for sampling instances is 1000 and for $BKNs$ is 30.

5.1 Efficiency Evaluation

Figure 5 illustrates the query efficiency of aU-tree and *DoubleSample* technique in terms of CPU cost, I/O cost and total cost. *DoubleSample* techniques outperforms aU-tree significantly. As $BKNs$ increases, the query time increases too since more uncertain objects are sampled to form $\mathcal{U}_s$.

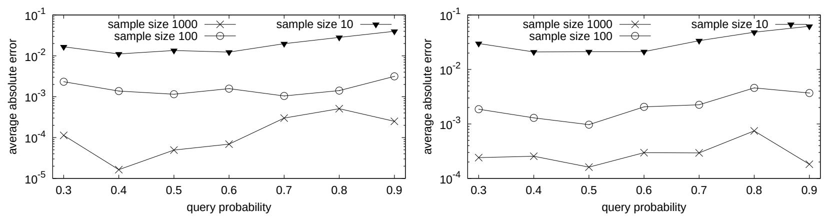
5.2 Accuracy Evaluation

Accuracy is defined as relative error of approximate answer w.r.t. exact answer by the formula: $accuracy = |\frac{R_{ex} - R_{ap}}{R_{ex}}|$, where R_{ex} and R_{ap} present the exact



(a) CPU time. (b) IO cost. (c) Total cost.

Fig. 5. Efficiency Evaluation.



(a) LB. (b) Aircraft.

Fig. 6. Accuracy Evaluation for singleSample.

result and approximate result, respectively. We use 0.4 as a default value for p_q. We evaluate the accuracy for *SingleSample* in Figure 6. As expected, accuracy increases as the sample size gets larger.

In Figure 7, we evaluate the accuracy of *DoubleSample*. We fix the sample instance size at 1000 and vary $BKNs$. Clearly, if more $BKNs$ are deployed, higher accuracy will be obtained. This trend is obvious for 2D dataset LB and less obvious for 3D datasets. We compare the accuracy of *SingleSample* and *Dou-*

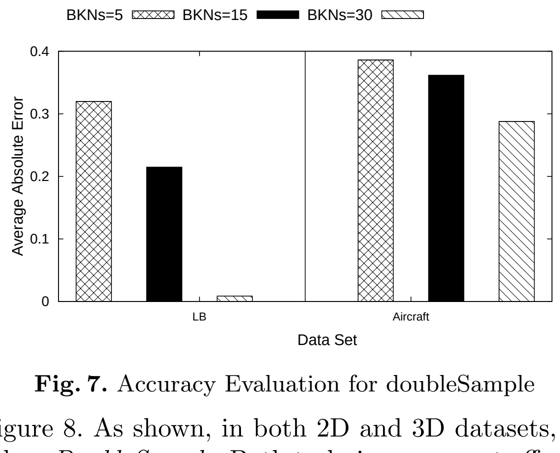
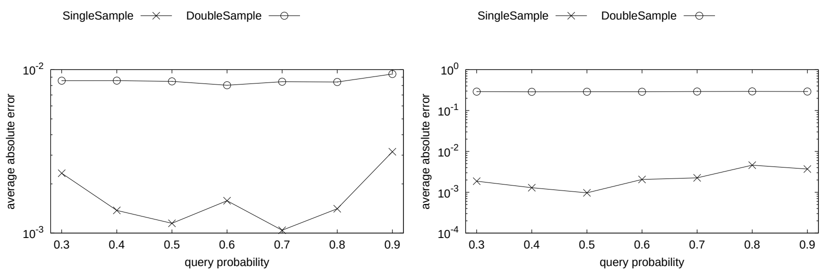


Fig. 7. Accuracy Evaluation for doubleSample

bleSample in Figure 8. As shown, in both 2D and 3D datasets, *SingleSample* is more accurate than *DoubleSample*. Both techniques are not affected significantly by p_q values.



(a) LB. (b) Aircraft.

Fig. 8. Accuracy Comparison.

6 Related Work

Considerable research effort has been put into modeling and managing uncertain data in recent years due to many emerging applications. Sarma *et al* [16] models uncertain data using *possible world semantics* and a prototype of uncertain data management system, *Trio*, is developed by the Stanford Info Lab [12]. General issues in modelling and managing uncertain data are addressed by Dey and Sarkar in [4], Lee in [11], and Antova, Koch, and Olteanu in [6]. Querying uncertain data by the probabilistic paradigm has been investigated by Dalvi and Suciu in [2] and Sen and Deshpande in [17]. Very recently Dalvi and Suciu [3] have shown that the problem of query evaluation over probabilistic databases is either $PTIME$ or *#P-complete*.

A number of problems in querying uncertain data have also been studied, such as indexing [18], similarity join [5], nearest neighbor query [9], skyline query [15], clustering [10], etc. Relatively complete and detailed study of existing techniques on managing uncertainty can be found in [14] and [20].

Range aggregate query over certain data is thoroughly studied in [19]. Range query over uncertain data [18] is the one that is the mostly related with our work, as we introduced in Section 2. To the best of our knowledge, this paper is the first one to address range aggregate query over uncertain data.

7 Conclusion

An important problem, probabilistic threshold range aggregate query over uncertain data is investigated in this paper. After formally defining this problem, we propose a novel index structure aU-tree to retrieve exact answers to $PTRA$ queries. To trade-off between efficiency and accuracy, *SingleSample* and *DoubleSample* methods are developed to approximately answer $PTRA$ queries. Our experimental study confirms the efficiency and effectiveness of the techniques we proposed.

Our techniques proposed can be extended to *continuous cases* directly. Based on the continuous probability density functions of uncertain objects, Monte Carlo sampling technique can be utilized and obtain a set of uncertain instances for each object. Thus both exact and approximate querying algorithms can be applied.

Acknowledgement The work of the authors was partially supported by ARC Grant (DP0881035, DP0666428 and DP0987557) and a Google Research Award.

References

1. S. Acharya, V. Poosala, and S. Ramaswamy. Selectivity estimation in spatial databases. In *SIGMOD 1999*.
2. N. Dalvi and D. Suciu. Efficient query evaluation on probabilistic databases. In *VLDB 2004*.
3. N. Dalvi and D. Suciu. Management of probabilistic data: foundations and challenges. In *PODS 2007*.
4. D. Dey and S. Sarkar. A probabilistic relational model and algebra. In *TODS 1996*.
5. H. P. Kriegel et al. Probabilistic similarity join on uncertain data. In *DASFAA 2006*.
6. L. Antova et al. 10^{10^6} worlds and beyond: Efficient representation and processing of incomplete information. In *ICDE 2007*.
7. A. Guttman. R-trees: A dynamic index structure for spatial searching. In *SIGMOD 1984*.
8. M. Hua, J. Pei, X. Lin, and W. Zhang. Efficiently answering probabilistic threshold top-k queries on uncertain data. In *ICDE 2008*.
9. H.-P. Kriegel, P. Kunath, and M. Renz. Probabilistic nearest-neighbor query on uncertain objects. In *DASFAA 2007*.
10. H. P. Kriegel and M. Pfeifle. Density-based clustering of uncertain data. In *KDD 2005*.
11. S. K. Lee. Imprecise and uncertain information in databases: an evidential approach. In *ICDE 1992*.
12. P.Agrawal, O.Benjelloun, A.Das Sarma, C.Hayworth, S.Nabar, T.Sugihara, and J.Widom. Trio: A system for data, uncertainty, and lineage. In *VLDB 2006*.
13. D. Papadias, P. kalnis, J. Zhang, and Y. Tao. Efficient olap operations in spatial data warehouses. In *SSTD 2001*.
14. J. Pei, M. Hua, Y. Tao, and X. Lin. Query answering techniques on uncertain and probabilistic data. In *SIGMOD 2008*.
15. J. Pei, B. Jiang, X. Lin, and Y. Yuan. Probabilistic skyline on uncertain data. In *VLDB*, 2007.
16. A. D. Sarma, O. Benjelloun, A. Halevy, and J. Widom. Working models for uncertain data. In *ICDE 2005*.
17. P. Sen and A. Deshpande. Representing and querying correlated tuples in probabilistic databases. In *ICDE 2007*.
18. 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 *VLDB*, pages 922–933, 2005.
19. Y. Tao and D. Papadias. Range aggregate processing in spatial databases. *ACM TODS*, 16(12):1555–1570, 2004.
20. W. Zhang, X. Lin, J. Pei, and Y. Zhang. Managing uncertain data: Probabilistic approaches. In *WAIM 2008*.