Probabilistic Skylines on Uncertain Data *

Jian Pei† Bin Jiang‡ Xuemin Lin‡ Yidong Yuan‡
†Simon Fraser University, Canada ‡The University of New South Wales & NICTA, Australia
jpei@cs.sfu.ca, {bjiang, lxue, yyidong}@cse.unsw.edu.au

ABSTRACT

Uncertain data are inherent in some important applications. Although a considerable amount of research has been dedicated to modeling uncertain data and answering some types of queries on uncertain data, how to conduct advanced analysis on uncertain data remains an open problem at large. In this paper, we tackle the problem of *skyline analysis on uncertain data*. We propose a *novel probabilistic skyline model* where an uncertain object may take a probability to be in the skyline, and a p-skyline contains all the objects whose skyline probabilities are at least p. Computing probabilistic skylines on large uncertain data sets is challenging. We develop two efficient algorithms. The bottom-up algorithm computes the skyline probabilities of some selected instances of uncertain objects, and uses those instances to prune other instances and uncertain objects effectively. The top-down algorithm recursively partitions the instances of uncertain objects into subsets, and prunes subsets and objects aggressively. Our experimental results on both the real NBA player data set and the benchmark synthetic data sets show that probabilistic skylines are interesting and useful, and our two algorithms are efficient on large data sets, and complementary to each other in performance.

1. INTRODUCTION

Uncertain data are inherent in some important applications, such as environmental surveillance, market analysis, and quantitative economics research. Uncertain data in those applications are generally caused by data randomness and incompleteness, limitations of measuring equipment, delayed data updates, etc. Due to the importance of those applications and the rapidly increasing amount of uncertain data collected and accumulated, analyzing large collections of uncertain data has become an important task. However, how to conduct advanced analysis on uncertain data, particularly skyline analysis as will be addressed in this study, remains an open problem at large.

1.1 Motivating Examples

Many previous studies (e.g., [4, 6, 14, 20, 24, 23, 26, 29, 32]) showed that skyline analysis is very useful in multi-criteria decision making applications. As an example, consider analyzing NBA players using multiple technical statistics criteria (e.g., the number of assists and the number of rebounds). Ideally, we want to find the perfect player who can achieve the best performance in all aspects. Unfortunately, such a player does not exist. The skyline analysis here is meaningful since it discloses the tradeoff between the merits of multiple aspects.

A player U is in the skyline if there exists no another player V such that V is better than U in one aspect, and is not worst than U in all other aspects. Skyline analysis on the technical statistics data of NBA players can identify excellent players and their outstanding merits.

We argue that skyline analysis is also meaningful on uncertain data. Consider the skyline analysis on NBA players again. Since the annual statistics are used as certain data in the previous studies [24], it has never been addressed in the skyline analysis that players may have different performances in different games. *If the game-by-game performance data are considered, which players should be in the skyline and why?*

For example, let us use the number of assists and the number of rebounds, both the larger the better, to examine the players. The two measures may vary substantially player by player and game by game. Uncertainty is inherent due to many factors such as the fluctuations of players' conditions, the locations of the games, and the support from audience. *How can we define the skyline given the uncertain data?*

While a skyline analysis using the real NBA game records will be reported in Section 6, here we plot a few games of 5 synthesis players in Figure 1 to illustrate several important issues.

The traditional method represents an attribute of each player using an aggregate function such as the mean or the median, and computes the skyline over such aggregate values. However, such aggregate values cannot capture the performance distribution information, and the skyline com-

*The research of Jian Pei is supported in part by an NSERC Discovery Grant. The research of Bin Jiang, Xuemin Lin, and Yidong Yuan is supported in part by the Australian Research Council Discovery Grant DP0666428 and the UNSW Faculty Research Grant Program. All opinions, findings, conclusions and recommendations in this paper are those of the authors and do not necessarily reflect the views of the funding agencies.
We thank the anonymous reviewers and Dr. Benjamin C. M. Kao for their comments.

Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the VLDB copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Very Large Data Base Endowment. To copy otherwise, or to republish, to post on servers or to redistribute to lists, requires a fee and/or special permission from the publisher, ACM.
VLDB '07, September 23-28, 2007, Vienna, Austria.
Copyright 2007 VLDB Endowment, ACM 978-1-59593-649-3/07/09.

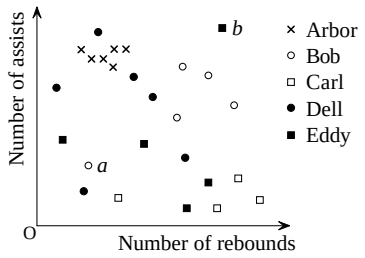


Figure 1: A set of synthesis players.

puted using such aggregate values may not be meaningful.

First, performances in different games may vary differently. For example, in Figure 1, player Arbor's performances are quite consistent while Eddy's performances are quite diverse. Although Eddy's performance in one game (point b in the figure) is better than Arbor's performances in all games in both the number of assists and the number of rebounds, Arbor is generally better than Eddy in the number of assists if all games they played are considered.

Second, some outliers may bias the aggregate of a player. For example, Bob is good in general, but he has an outlier game (point a) of poor performance in both measures.

In order to handle the uncertain data, a naïve approach is to compute the skyline on the game records instead of the players. However, the game records can be regarded as the samples of the players' performances and the samples cannot be complete. A skyline game record may be just an exception of a player (e.g., point b of Eddy in Figure 1). Thus, the skyline of game records may not be meaningful for comparing players.

There can be a large number of players over years and each player may play many games in his career. Therefore, the efficiency of skyline analysis on uncertain data matters.

There are many other application examples for skyline analysis on uncertain data. For example, to evaluate the effect of therapies in medical practice, test cases are collected, and a few measures are used. Generally, the measures may vary, sometimes even substantially, among the test cases of one therapy. Uncertainty is inherent due to the incompleteness of the samples and many other factors (e.g., the physical conditions of patients). Finding the skyline therapies on the uncertain data helps to identify good therapies and understand the tradeoff among multiple factors in question.

In summary, uncertain data pose a few new challenges for skyline analysis and computation. Specifically, we need a meaningful yet simple model for skylines on uncertain data. Moreover, we need to develop efficient algorithms for such skyline computation.

1.2 Challenges and Our Contributions

In this paper, we address two major challenges about skyline analysis and computation on uncertain data.

Challenge 1: Modeling Skylines on Uncertain Data

In a set of uncertain objects, each object has multiple instances, or alternatively, each object is associated with a probability density function. A model about skylines on uncertain data needs to answer two questions: *How can we capture the dominance relation between uncertain objects?* and *What should be the skyline on those uncertain objects?*

Our contributions. We introduce the probabilistic nature of uncertain objects into the skyline analysis. Essentially, to compare the advantages between two objects, we calculate the probability that one object dominates the other. Based on the probabilistic dominance relation, we propose the notion of *probabilistic skyline*. The probability of an object being in the skyline is the probability that the object is not dominated by any other objects.

Given a probability threshold p ($0 \leq p \leq 1$), the *p-skyline* is the set of uncertain objects each of which takes a probability of at least p to be in the skyline.

Comparing to the traditional skyline analysis, probabilistic skyline analysis is more informative on uncertain objects. For example, in a case study (details in Section 6) using the game-by-game technical statistics of $1,313$ NBA players in $339,721$ games, the 0.2-skyline includes 14 players. Among them, Hakeem Olajuwon and Kobe Bryant are not in the traditional skyline where only the aggregate statistics are used. The reason is that their performances vary a lot over games. On the other hand, some players that are in the traditional skyline have a low skyline probability such as Gary Payton (0.126) and Lamar Odom (0.102). This information cannot be obtained using the traditional skyline analysis.

To the best of our knowledge, this paper is the first study conducting the skyline analysis using the detailed instances of uncertain objects.

Challenge 2: Efficient Computation of Probabilistic Skylines

Computing a probabilistic skyline is much more complicated than computing a skyline on certain data. Particularly, in many applications, the probability density function of an uncertain object is often unavailable explicitly. Instead, a set of instances are collected in the hope of approximating the probability density function. Thus, it is challenging to compute probabilistic skylines on uncertain objects each of which is represented by a set of instances.

In this paper, we focus on the discrete case of probabilistic skylines computation, i.e., each uncertain object is represented by a set of instances, while some of our ideas can be applied to the continuous case as well.

First, each uncertain object may have many instances. We have to process a large number of instances. Second, we have to consider many probabilities in deriving the probabilistic skylines. For example, as reported in Section 6, a straightforward method takes more than 1 hour to compute the 0.3-skyline on the NBA data set. *Can we devise efficient methods to compute probabilistic skylines efficiently?*

Our contributions. We develop two algorithms to tackle the problem. The bottom-up algorithm computes the skyline probabilities of some selected instances of uncertain objects, and uses those instances to prune other instances and uncertain objects effectively. The top-down algorithm recursively partitions the instances of uncertain objects into subsets, and prunes subsets and objects aggressively. Our methods are efficient and scalable. As verified by our extensive experimental results, our methods are tens of times faster than the straightforward method.

The rest of the paper is organized as follows. In Section 2, we propose the notion of probabilistic skylines on uncertain data. In Sections 3 and 4, we develop the bottom-up and the top-down methods for probabilistic skyline computation, respectively. We review the related work in Section 5. A systematic performance study is reported in Section 6. We conclude the paper in Section 7.

Notation	Definition
D	data space, $D = (D_1, \ldots, D_n)$
U, V	uncertain objects
u, v	instances of uncertain objects
$\|U\|$	the number of instances of U
$u \prec v$	instance u dominates instance v
$Pr[U \prec V]$	the probability that U dominates V
$Pr(U)$ $(Pr(u))$	the skyline probability of U (u)
$Pr^+(\cdot)$ $(Pr^-(\cdot))$	the upper (lower) bound of $Pr(U)$ or $Pr(u)$
U_{min} (U_{max})	the minimum (maximum) corner of the minimum bounding box of U
$u.key$	the key of an instance, $u.key = \sum_{i=1}^{n} u.D_i$
N	a node in a partition tree
$N.MBB$	the MBB of a node in a partition tree
N_{min} (N_{max})	the minimum (maximum) corner of $N.MBB$

Table 1: The summary of notations.

2. PROBABILISTIC SKYLINES

In this section, we present the probabilistic skyline mode. For reference, a summary of notations is given in Table 1.

2.1 Skylines on Certain Objects

By default, we consider points in an n-dimensional numeric space $D = (D_1, \ldots, D_n)$. The dominance relation is built on the preferences on attributes $D_1, \ldots, D_n$. Without loss of generality, we assume that, on $D_1, \ldots, D_n$, smaller values are more preferable.

For two points u and v, u is said to *dominate* v, denoted by $u \prec v$, if for every dimension D_i $(1 \leq i \leq n)$, $u.D_i \leq v.D_i$, and there exists a dimension D_{i_0} $(1 \leq i_0 \leq n)$ such that $u.D_{i_0} < v.D_{i_0}$.

Given a set of points S, a point u is a *skyline point* if there exists no another point $v \in S$ such that v dominates u. The *skyline* on S is the set of all skyline points.

Example 1 (Dominance and skyline). Consider the points in Figure 2. According to the definition of dominance, point c dominates d and e, c dominates b. Points a, c and f are not dominated by any other points in the set. Thus, these 3 points form the skyline of this data set. ∎

2.2 Probabilistic Skylines

An *uncertain object* is conceptually described by a *probability density function* (PDF) f in the data space D. Generally, $f(u) \geq 0$ for any point u in the data space D, and $\int_{u \in D} f(u)du = 1$.

Practically, the probability density function of an uncertain object is often unavailable explicitly. Instead, a set of samples are drawn or collected in the hope of approximating the probability density function. Correspondingly, we model an uncertain object U as a set of multiple points in the data space as its *instances*, denoted by $U = \{u_1, \ldots, u_l\}$. It can be regarded as the *discrete case*. The number of instances of an uncertain object U is written as $|U| = l$.

To keep our model simple, we assume that *uncertain objects are independent*. That is, an instance of an object does not depend on the instances of any other objects. Moreover, we assume that, *for an uncertain object, each instance carries the same probability to happen*. Although the rest of this paper bears the above two assumptions, our model can be extended to cases where dependencies (e.g., correlations or anti-correlations) exist among objects and instances carry different weights. Limited by space, we omit the details.

Now let us extend the dominance relation to uncertain objects, and check whether such an extension can straightforwardly define skylines on uncertain objects.

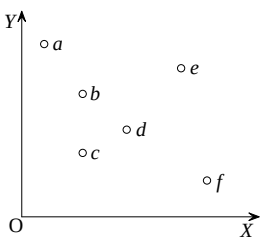


Figure 2: A set of certain points.

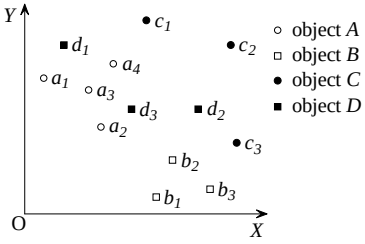


Figure 3: A set of uncertain objects.

Let U and V be two uncertain objects, and f and f' be the corresponding probability density functions, respectively. Then, the probability that V dominates U is

$$\begin{aligned} Pr[V \prec U] &= \int_{u \in D} f(u)(\int_{v \prec u} f'(v)\, dv)\, du \\ &= \int_{u \in D} \int_{v \prec u} f(u)f'(v)\, dv\, du \end{aligned} \quad (1)$$

In the discrete case, let $U = \{u_1, \ldots, u_{l_1}\}$ and $V = \{v_1, \ldots, v_{l_2}\}$ be two uncertain objects and their instances. The probability that V dominates U is given by

$$\begin{aligned} Pr[V \prec U] &= \sum_{i=1}^{l_1} \frac{1}{l_1} \cdot \frac{|\{v_j \in V | v_j \prec u_i\}|}{l_2} \\ &= \frac{1}{l_1 l_2} \sum_{i=1}^{l_1} |\{v_j \in V \mid v_j \prec u_i\}| \end{aligned} \quad (2)$$

Since any two points u and v in the data space must have one of the following three relations: $u \prec v$, $v \prec u$, or u and v do not dominate each other, for two uncertain objects U and V, $Pr[U \prec V] + Pr[V \prec U] \leq 1$.

Example 2 (Probabilistic dominance relation). Consider the set of 4 uncertain objects in Figure 3. Object C has 3 instances: c_1 and c_2 are dominated by every instance of A, and c_3 is not dominated by any instance of A. Thus, the probability that A dominates C is $Pr[A \prec C] = \frac{2}{3}$. Similarly, we can calculate $Pr[B \prec C] = \frac{2}{3}$.

To calculate $Pr[A \prec D]$, we notice that $a_1 \prec d_1$, $a_2 \prec d_2$, and $a_2 \prec d_3$. According to Equation (2), we have $Pr[A \prec D] = \frac{1}{4 \times 3} \times 3 = \frac{1}{4}$.

Consider the three instances of C. Since c_1 is dominated by every instance of object A, c_2 is dominated by every instance of A and B, and c_3 is dominated by every instance of B, the probability that C is dominated by A or B is 1. In other words, C cannot be in the skyline.

An important observation here is that, although $Pr[A \prec C] = \frac{2}{3} < 1$ and $Pr[B \prec C] = \frac{2}{3} < 1$, the probability of C being dominated by A or B is 1. Moreover,

$Pr[(A \not\prec C) \wedge (B \not\prec C)] \neq (1 - Pr[A \prec C]) \cdot (1 - Pr[B \prec C])$. ∎

The observation in Example 2 indicates that *the probabilistic dominance relation cannot be used straightforwardly to define skylines on uncertain objects*. Then, *what is the probability that an uncertain object is in the skyline?*

Example 3 (Probabilistic skyline). Consider the objects in Figure 3 again. For object A, every instance of the object is not dominated by any instances of objects B, C or D. Thus, the probability that A is dominated by any object is 0, and the probability that A is in the skyline is 1. Similarly, the probability that B is in the skyline is also 1.

For object D, instance d_1 is dominated by a_1, d_2 is dominated by a_2, b_1 and b_2, and d_3 is dominated by a_2. Thus,

the probability that D is not dominated by any other object can be calculated as

$$\begin{aligned}
&\tfrac{1}{3} \times (&& D \text{ has three instances} \\
&\quad (1-\tfrac{1}{4}) + && \text{case of } d_1 \\
&\quad (1-\tfrac{1}{4}) \times (1-\tfrac{2}{3}) + && \text{case of } d_2 \\
&\quad (1-\tfrac{1}{4})) && \text{case of } d_3 \\
&= \tfrac{7}{12}
\end{aligned}$$

Thus, D takes a probability of $\frac{7}{12}$ to be in the skyline. ∎

Generally, consider an uncertain object U with probability density function f. The probability that U appears at position u in data space D is given by $f(u)$. For any other object $V \neq U$ with probability density function f', the probability that V dominates u is $\int_{v \prec u} f'(v)dv$. Thus, the probability that u is not dominated by any other object is $\prod_{V \neq U}(1 - \int_{v \prec u} f'(v)dv)$. The probability that U is in the skyline is

$$Pr(U) = \int_{u \in D} f(u) \prod_{V \neq U} (1 - \int_{v \prec u} f'(v)dv)du \qquad (3)$$

$Pr(U)$ is called the *skyline probability* of U.

In the discrete case, let $U = \{u_1, \ldots, u_l\}$ be an uncertain object and its instances. Equation (3) can be written as

$$Pr(U) = \frac{1}{l} \sum_{i=1}^{l} \prod_{V \neq U} (1 - \frac{|\{v \in V \mid v \prec u_i\}|}{|V|}) \qquad (4)$$

Moreover, for an instance $u \in U$,

$$Pr(u) = \prod_{V \neq U} (1 - \frac{|\{v \in V \mid v \prec u\}|}{|V|}) \qquad (5)$$

$Pr(u)$ is the probability that u is not dominated by any other objects, i.e., u is in the skyline. It is called the *skyline probability* of instance u. Equation (4) can be written as

$$Pr(U) = \frac{1}{l} \sum_{u \in U} Pr(u) \qquad (6)$$

Intuitively, let $U_1, \ldots, U_m$ be the set of uncertain objects in question, where each object U_i $(1 \leq i \leq m)$ takes the probability of $\frac{1}{|U_i|}$ to appear as one of its instances. Then, the number of possible worlds of the data set is $\prod_{i=1}^{m} |U_i|$. For an instance $u_{i,j} \in U_i$, the probability of $u_{i,j}$ being in the skyline is the ratio of the number of worlds where $u_{i,j}$ is in the skyline against all the possible worlds. Moreover, the probability that U_i is in the skyline is the expectation of the probability that an instance of U_i is in the skyline.

An uncertain object may take a probability to be in the skyline. It is natural to extend the notion of skyline to *probabilistic skyline*. For a set of uncertain objects S and a *probability threshold* p $(0 \leq p \leq 1)$, the *p-skyline* is the subset of objects in S each of which takes a probability of at least p to be in the skyline. That is,

$$Sky(p) = \{U \in S | Pr(U) \geq p\}.$$

Problem definition. Given a set of uncertain objects S and a probability threshold p $(0 \leq p \leq 1)$, the problem of *probabilistic skyline computation* is to compute the p-skyline on S.

Particularly, in this paper we tackle the discrete case. That is, given a set of uncertain objects where each object is a set of sample instances and a probability threshold p, compute the p-skyline. ∎

Although we will focus on the discrete case in this paper, some of our ideas can be applied to handle the general case, which will be discussed briefly in Section 7.

3. THE BOTTOM-UP METHOD

We develop two methods of computing the p-skyline over a set of uncertain objects. They both follow the *bounding-pruning-refining* iteration.

Bounding: For an instance of an uncertain object, we compute an upper bound and a lower bound on its skyline probability. Then, using Equation (6) we obtain an upper bound and a lower bound on the skyline probability of an uncertain object.

Pruning: For an uncertain object U, if the lower bound of $Pr(U)$ is larger than p, the probability threshold, then U is in the p-skyline. If the upper bound of $Pr(U)$ is smaller than p, then U is not in the p-skyline.

Refining: If p is between the lower bound and the upper bound, then we need to get tighter bounds of the skyline probabilities by the next iteration of bounding, pruning and refining.

The above iteration goes on until for every uncertain object we can determine whether it is in the p-skyline or not. The two methods we propose differ in how to compute and refine the bounds and how to prune uncertain objects.

Particularly, in the bottom-up method presented in this section, we compute and refine the bounds of instances of uncertain objects by selectively computing the skyline probabilities of a small subset of instances. An uncertain object may be pruned using the skyline probabilities of its instances, or those of some other objects. This method is called *bottom-up* since the bound computation and refinement start from instances (bottom) and go up to skyline probabilities of objects.

3.1 Bounding Skyline Probabilities

Given an uncertain object U and an instance u of U, trivially, we have $0 \leq Pr(U) \leq 1$ and $0 \leq Pr(u) \leq 1$.

Let $U_{min} = (\min_{i=1}^{|U|}\{u_i.D_1\}, \ldots, \min_{i=1}^{|U|}\{u_i.D_n\})$ and $U_{max} = (\max_{i=1}^{|U|}\{u_i.D_1\}, \ldots, \max_{i=1}^{|U|}\{u_i.D_n\})$ be the minimum and the maximum corners of the minimum bounding box (MBB for short) of U, respectively. Note that, U_{min} and U_{max} are not necessary two actual instances of U. In this case, we treat them as virtual instances and define their skyline probabilities following equation (5). That is, $Pr(U_{min}) = \prod_{V \neq U}(1 - \frac{|\{v \in V \mid v \prec U_{min}\}|}{|V|})$, and $Pr(U_{max}) = \prod_{V \neq U}(1 - \frac{|\{v \in V \mid v \prec U_{max}\}|}{|V|})$.

LEMMA 1 (BOUNDING SKYLINE PROBABILITIES). *Let $U = \{u_1, \ldots, u_l\}$ be an uncertain object where $u_1, \ldots, u_l$ are the instances of U.*

(1) If $u_{i_1} \prec u_{i_2}$ $(0 \leq i_1, i_2 \leq l)$, then $Pr(u_{i_1}) \geq Pr(u_{i_2})$.

(2) $Pr(U_{min}) \geq Pr(U) \geq Pr(U_{max})$

Proof sketch. Dominance relations on instances are transitive: for instances x, y, and z, if $x \prec y$ and $y \prec z$, then $x \prec z$. Since $u_{i_1} \prec u_{i_2}$, for any instance v of other object V, if $v \prec u_{i_1}$ then $v \prec u_{i_2}$. Applying this observation to Equation (5), we have

$$\begin{aligned}
Pr(u_{i_1}) &= \prod_{V \neq U}(1 - \tfrac{|\{v \in V \mid v \prec u_{i_1}\}|}{|V|}) \\
&\geq \prod_{V \neq U}(1 - \tfrac{|\{v \in V \mid v \prec u_{i_2}\}|}{|V|}) \\
&= Pr(u_{i_2})
\end{aligned}$$

According to item 1 in this lemma, for any u_i $(1 \leq i \leq l)$, $Pr(U_{min}) \geq Pr(u_i) \geq Pr(U_{max})$. Item 2 in the lemma follows with the above inequality and Equation (6). ∎

Lemma 1 provides a means to compute the upper bounds and the lower bounds of instances and uncertain objects using the skyline probabilities of other instances.

According to the first inequality in the lemma, the skyline probability of an instance can be bounded by those of other instances dominating or dominated by it. In other words, when the skyline probability of an instance is calculated, the bounds of the skyline probabilities of some other instances of the same object may be refined accordingly.

The second inequality in the lemma indicates that the minimum and the maximum corners of the MBB can play important roles in bounding the skyline probability of a set of instances.

3.2 Pruning Techniques

If the skyline probability of an uncertain object or an instance of an uncertain object is computed, can we use this information to prune the other uncertain instances or objects? Following with Lemma 1, we immediately have the following rule to determine the p-skyline membership of an uncertain object using its minimum or maximum corners.

PRUNING RULE 1. *For an uncertain object U and probability threshold p, if $Pr(U_{min}) < p$, then U is not in the p-skyline. If $Pr(U_{max}) \geq p$, then U is in the p-skyline.* ∎

Moreover, we can prune an uncertain object using the upper bounds and the lower bounds of the skyline probabilities of instances.

PRUNING RULE 2. *Let U be an uncertain object. For each instance $u \in U$, let $Pr^+(u)$ and $Pr^-(u)$ be the upper bound and the lower bound of $Pr(u)$, respectively. If $\frac{1}{|U|}\sum_{u \in U} Pr^+(u) < p$, then U is not in the p-skyline. If $\frac{1}{|U|}\sum_{u \in U} Pr^-(u) \geq p$, then U is in the p-skyline.* ∎

We can also use the information about one uncertain object to prune other uncertain instances or objects. First, if an instance u of an uncertain object U is dominated by the maximum corner of another uncertain object V, then u cannot be in the skyline.

PRUNING RULE 3. *Let U and V be uncertain objects such that $U \neq V$. If u is an instance of U and $V_{max} \prec u$, then $Pr(u) = 0$.* ∎

By pruning some instances in an uncertain object using the above rule, we can reduce the cost of computing the skyline probability of the object.

When the skyline probabilities of some instances of an uncertain object are computed, we can use the information to prune some other uncertain objects.

PRUNING RULE 4. *Let U and V be two uncertain objects and $U' \subseteq U$ be a subset of instances of U such that $U'_{max} \preceq V_{min}$. If $\frac{|U-U'|}{|U|} \cdot \min_{u \in U'}\{Pr(u)\} < p$, then $Pr(V) < p$ and thus V is not in the p-skyline.*

Proof. Figure 4 illustrates the situation. Since V_{min} is dominated by all instances in U'. An instance of V can be in the skyline only if U does not appear as any instance in U'. Even no instance in $(U - U')$ dominates any instance of V, since $U'_{max} \preceq V_{min}$ and $\frac{|U-U'|}{|U|} \cdot \min_{u \in U'}\{Pr(u)\} < p$, the probability that V is in the skyline still cannot reach the probability threshold p. Thus V cannot be in the p-skyline.

Formally, since every instance of V is dominated by all instances in U', only when U takes an instance in $(U - U')$, V may have a chance of not being dominated by U. The probability that an instance of V is not dominated by an instance of U cannot be more than $(1 - \frac{|U'|}{|U|}) = \frac{|U-U'|}{|U|}$. Moreover, since $U'_{max} \preceq V_{min}$, all instances of objects other than U and V dominating U'_{max} also dominate V_{min}.

Thus, $Pr(V) \leq Pr(V_{min}) \leq (1 - \frac{|U'|}{|U|}) \cdot Pr(U'_{max}) = \frac{|U-U'|}{U} \cdot Pr(U'_{max}) \leq \frac{|U-U'|}{U} \cdot \min_{u \in U'}\{Pr(u)\} < p$.

As a special case, if there exists an instance $u \in U$ such that $Pr(u) < p$ and $u \preceq V_{min}$, then $Pr(V) < p$ and V can be pruned. ∎

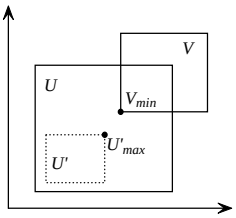


Figure 4: An illustration of Pruning Rule 4.

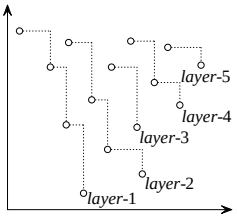


Figure 5: The layers of an uncertain object.

The pruning rule is powerful since even an uncertain object partially computed can be used to prune other objects.

3.3 Refinement Strategies

For an uncertain object U, we want to determine whether U is in the p-skyline by computing the skyline probabilities of as few instances of U as possible. Finding an optimal subset of instances to compute is a very difficult online problem since, without computing the probabilities of the instances, we do not know their distribution. Here, we propose a layer-by-layer heuristic method.

3.3.1 Layers of Instances

According to the first inequality in Lemma 1, among all instances of an object U, we can first compute the skyline probabilities of the instances that are not dominated by any other instances, i.e., the skyline instances in the object. Those instances are the *layer*-1 instances, as illustrated in Figure 5. The skyline probabilities of the instances at *layer*-1 can serve as the upper bounds of the skyline probabilities of other instances, and generate an upper bound of the skyline probability of U.

If the upper bounds using the *layer*-1 instances are not enough to qualify or disqualify U in the p-skyline, we need to refine the upper bounds. We can compute the skyline probabilities for instances at *layer*-2 which are dominated only by instances at *layer*-1, as shown in Figure 5. Similarly, we can partition the instances of an object into layers.

Formally, for an uncertain object U, an instance $u \in U$ is at *layer*-1 if u is not dominated by any other instance in U. An instance v is at *layer-k* $(k > 1)$ if, v is not at the 1st, ..., $(k-1)$-th layers, and v is not dominated by any instances except for those at the 1st, ..., $(k-1)$-th layers.

The advantage of partitioning instances of an object into layers is that, once the skyline probabilities of all instances

at one layer are calculated, the probabilities can be used as the upper bounds of the instances at the higher layers.

LEMMA 2. *In an uncertain object U, let $u_{1,1}, \ldots, u_{1,l_1}$ be the instances at layer-k_1, and $u_{2,1}, \ldots, u_{2,l_2}$ be the instances at layer-k_2, $k_1 < k_2$. Then, for any instance at layer-k_2 u_{2,j_2} $(1 \leq j_2 \leq l_2)$, there exists an instance at layer-k_1 u_{1,j_1} $(1 \leq j_1 \leq l_1)$ such that $Pr(u_{1,j_1}) \geq Pr(u_{2,j_2})$. Moreover, $\max_{i=1}^{l_1}\{Pr[u_{1,i}]\} \geq \max_{j=1}^{l_2}\{Pr(u_{2,j})\}$.*

Proof sketch. Since $k_1 < k_2$, instance u_{2,j_2} must be dominated by an instance at *layer-k_1*. Otherwise, u_{2,j_2} is at *layer-k_1* or some lower layer. Let u_{1,j_1} be an instance at *layer-k_1* that dominates u_{2,j_2}. Then, the first inequality follows with Lemma 1. The second inequality follows with the first inequality. ∎

3.3.2 Partitioning Instances to Layers

How can instances of an object be assigned quickly into layers?

For each instance u, we define the *key* of the instance as the sum of its values on all attributes, that is, $u.key = \sum_{i=1}^{n} u.D_i$. Then, we sort all the instances in the key ascending order. This is motivated by the SFS algorithm [10]. The sorted list of instances has a nice property: for instances u and v such that $u \prec v$, u precedes v in the sorted list.

We scan the sorted list once. The first instance has the minimum key value, and is assigned to *layer*-1. We compare the second instance with the first one. If the second one is dominated, then it is assigned to *layer*-2; otherwise it is assigned to *layer*-1.

Generally, when we process an instance u, suppose at the time there already exist h layers. We compare u with the instances currently at *layer-*$\lceil \frac{h}{2} \rceil$. One of the two cases may happen. If u is dominated by an instance at that layer, then u must be at some layer higher than $\lceil \frac{h}{2} \rceil$. Otherwise, u is neither dominated by, nor dominates, any instance at that layer. Then, u must be at that layer or some lower layer. We conduct this binary search recursively until u is assigned to a layer.

Lemma 1 indicates that the minimum corner of the MBB of an uncertain object leads to the upper bounds of the skyline probabilities of all instances as well as the object itself. As a special case, we assign this minimum corner as a virtual instance at *layer*-0.

The above partitioning method has a nice property: all instances at a layer are sorted in the key ascending order.

3.3.3 Scheduling Objects

From which objects should we start the skyline probability computation?

In order to use the pruning rules discussed in Section 3.2 as much as possible, those instances in uncertain objects that likely dominate many other objects or instances should be computed early. Heuristically, those instances which are close to the origin may have a better chance to dominate other objects and instances.

The instances of an uncertain object are processed layer by layer. Within each layer, the instances are processed in the key ascending order. As discussed in Section 3.2, some pruning rules enable us to use the partial information of some uncertain objects to prune other objects and instances, we interleave the processing of different objects.

Technically, all instances of an uncertain object are kept in a list. The minimum corner of its MBB is treated as a special instance and put at the head of the list. The heads of lists of all uncertain objects are organized into a heap. We iteratively process the top instance in the heap. If an object cannot be pruned after its minimum corner is processed, we organize the rest of instances in its list in the layer and key value ascending order. Once an instance from an object is processed, the object sends the next instance into the heap if its skyline membership is not determined. The proper pruning rules are triggered if the conditions are satisfied.

Input: a set of uncertain objects S; probability threshold p;
Output: the p-skyline in S;
Method:

```
1:  SKY = ∅;
2:  FOR EACH object U ∈ S DO
3:    Pr+(U) = 1; Pr−(U) = 0;
4:    compute U_min, the minimum corner of its MBB;
    END FOR EACH
5:  build an R-tree to store U_min for all U ∈ S;
6:  build a heap H on U_min for all U ∈ S;
7:  WHILE H ≠ ∅ DO
8:    let u ∈ U be the top instance in H;
9:    IF u is from a non-skyline object THEN NEXT;
10:   IF u is dominated by another object THEN
11:     GOTO Line 20; // Pruning Rule 3
12:   IF u is the minimum corner of U THEN
13:     find possible dominating objects of U; // Section 3.4.1
14:     compute Pr(u); // Section 3.4.2
15:     IF Pr(u) ≥ p THEN
16:       partition instances of U to layers; // Section 3.3.2
17:     ELSE U is pruned; // Pruning Rule 1
      ELSE
18:     compute Pr(u); // Section 3.4.2
19:     Pr−(U) = Pr−(U) + (1/|U|) Pr(u);
20:     IF u is the last instance at a layer THEN
21:       update U.Pr_max;
22:     Pr+(U) = Pr−(U) + U.Pr_max · (|Ũ|/|U|);
23:     IF (|U−U'|/|U|) · min_{u∈U'}{Pr(u)} < p THEN
24:       apply Pruning Rule 4 to prune other objects;
25:     IF Pr−(U) ≥ p THEN
26:       SKY = SKY ∪ {U}; NEXT; // Pruning Rule 2
27:     IF Pr+(U) ≥ p THEN
28:       insert the next instance of U into H;
    END WHILE
29: RETURN SKY;
```

Figure 6: The bottom-up algorithm.

3.4 Algorithm and Implementation

The bottom-up algorithm is shown in Figure 6. We explain some critical implementation details here.

3.4.1 Finding Possible Dominating Objects

For an object U, we want to find all other objects that may contain some instances dominating U. Those objects are called the *possible dominating objects* of U. The skyline membership of U depends on only those possible dominating objects. All other objects that do not contain any instances dominating U do not need to be considered.

To speed up the search of possible dominating objects, we organize the minimum corners of MBBs of all objects into a global R-tree. To find the possible dominating objects of U, we issue a window query with the origin and U_{max} as the opposite corners on the global R-tree. The possible dominating objects for an object are computed only when the minimum corner of the object is popped from the heap.

If an object U does not have any possible dominating objects, then every instance of U is not dominated by any

instance of other objects. In other words, the skyline probability of U is 1.

3.4.2 Computing Skyline Probability

To compute the skyline probability $Pr(u)$ for an instance $u \in U$, we compare u with the possible dominating objects of U one by one. To facilitate the comparison, we incrementally maintain a local R-tree T_V for each object V. T_V is set to empty in the initialization. When an instance $u \in U$ is compared with object V, we insert into T_V the instances in V that have a key value less than $u.key$, since only those instances in V may dominate u. Then, we issue a window query with the origin and u as the opposite corners to compute $|\{v \in V | v \prec u\}|$.

After comparing u with all possible dominating objects of U, by Equation (6), we can calculate $Pr(u)$. We also update the lower bound of the probability of object U immediately as $Pr^-(U) = Pr^-(U) + \frac{1}{|U|}Pr(u)$.

Once all instances in a layer are processed, as discussed in Lemma 2, we set the maximum probability of instances in this layer to be the upper bound (denoted by $U.Pr_{max}$) of the probabilities of instances in the higher layers. Moreover, the upper bound of the probability of U is updated as $Pr^+(U) = Pr^-(U) + U.Pr_{max} \cdot \frac{|\widetilde{U}|}{|U|}$, where $\widetilde{U} \subseteq U$ is the set of instances whose probabilities are not calculated yet.

3.4.3 Using Pruning Rule 4

In order to use Pruning Rule 4 to prune other objects, for each object U, we maintain U' as the set of instances which precede the current processing instance in its instance list. The skyline probability of those instances are already computed. Once U' satisfies the condition in the rule, we compute U'_{max}, the maximum corner of the MBB of U', and issue a window query on the global R-tree described in Section 3.4.1 with U'_{max} and the maximum corner of the MBB of all objects in the data set as the opposite corners. For each minimum corner returned from this query, the corresponding uncertain object satisfies the pruning rule and thus is not in the p-skyline. We note that for each object, this rule is applied at most once. This is because once this condition is satisfied, it will be always satisfied afterwards. But there is no more object can be pruned except for those pruned at the first time.

3.4.4 Complexity of the Algorithm

It can be immediately verified that the cost of the bottom-up algorithm is predominated by computing the skyline probabilities of instances as presented in Section 3.4.2. Suppose that R is the average cost of querying the local R-trees of possible dominating objects, with all pruning techniques are applied, for computing the skyline probabilities of instances. Let W_{total} denote the number of instances whose skyline probabilities are computed in the algorithm. Then, the average cost of the algorithm is $O(W_{total} \cdot R)$.

As shown in our experimental results, in practice many instances and objects can be pruned sharply. The bottom-up algorithm only has to compute a small portion of instances. That is, W_{total} is much smaller than the total number of instances. Thus, the bottom-up algorithm has good scalability on large data sets.

4. THE TOP-DOWN METHOD

In this section, we present a top-down method for probabilistic skyline computation. The method starts with the whole set of instances of an uncertain object. The skyline probability of the object can be bounded using the maximum and the minimum corners of the MBB of the object. To improve the bounds, we can recursively partition the instances into subsets. The skyline probability of each subset can be bounded using its MBB in the same way. Facilitated by Equation (6), the skyline probability of the uncertain object can be bounded as the weighted mean of the bounds of subsets. Once the p-skyline membership of the uncertain object is determined, the recursive bounding process stops.

4.1 Partition Trees

To facilitate the partitioning process, we use a *partition tree* data structure for each uncertain object. A partition tree is binary. Each leaf node contains a set of instances and the corresponding MBB. Each internal node maintains the MBB of all instances in its descendants and the total number of instances.

The construction of a partition tree for an uncertain object is somewhat similar to that of kd-trees [2]. We start with a tree of only one node – the root node which contains all instances of the object and the MBB. The tree grows in rounds. In each round, a leaf node with l instances ($l > 1$) is partitioned into two children nodes according to one attribute such that the left child and the right child contain $\lceil \frac{l}{2} \rceil$ and $\lfloor \frac{l}{2} \rfloor$ instances, respectively.

We take a simple round robin method to choose the attributes to grow a partition tree. The attributes are sorted into $D_1, \ldots, D_n$ in an arbitrary order. The root node (level-0) is partitioned into two children on attribute D_1, those children (level-1) are split into grand-children on attribute D_2, and so on. To split the nodes at level-n, attribute D_1 is used again.

The time complexity to grow one level of the tree for an uncertain object U is $O(|U|)$. The cost to fully grow a partition tree (i.e., each leaf node contains only one instance) is $O(|U| \log_2 |U|)$ since the tree has at most $\log_2 |U|$ levels.

4.2 Bounding Using Partition Trees

For a node N in a partition tree, we also use N to denote the set of instances allocated to N. Let $N.MBB$ be the MBB of the instances allocated to N, and N_{max} and N_{min} be the maximum and the minimum corners, respectively. Then, by Lemma 1, for any instance $u \in N$, the skyline probability of u can be bounded by

$$Pr(N_{max}) \leq Pr(u) \leq Pr(N_{min}). \tag{7}$$

Moreover, if the partition tree of uncertain object U has l leaf nodes $N_1, \ldots, N_l$, then

$$\frac{1}{|U|}\sum_{i=1}^{l} |N_i| \cdot Pr(N_{i,max}) \leq Pr(U) \leq \frac{1}{|U|}\sum_{i=1}^{l} |N_i| \cdot Pr(N_{i,min}), \tag{8}$$

where $N_{i,max}$ and $N_{i,min}$ are the maximum and the minimum corners of $N_i.MBB$, respectively, and $|N_i|$ is the number of instances in N_i.

Computing the exact skyline probabilities for all corners can be costly. Instead, we estimate the bounds. To bound the skyline probabilities for N_{min} and N_{max} for a node N in the partition tree of uncertain object U, we query the possible dominating objects of U (Section 3.4.1). We traverse the partition tree of each possible dominating object V of U in the depth-first manner. When a node M in the partition tree of V is met, one of the following three cases happens.

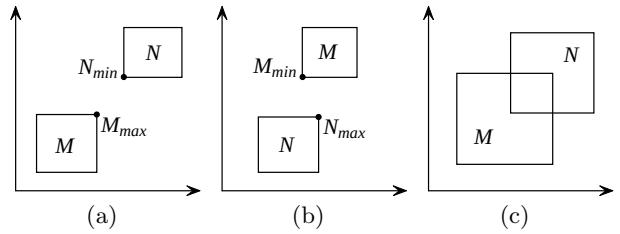


Figure 7: Three cases of bounding $Pr(N)$.

First, if M_{max} dominates N_{min} (as shown in Figure 7(a)), then N_{min} and N_{max} are dominated by all instances in M.

Second, if M_{min} does not dominate N_{max} (as shown in Figure 7(b)), then all instances in M cannot dominate either N_{min} or N_{max}.

Last, if the above two situations do not happen, then some instances in M may dominate some instances in N (as shown in Figure 7(c)). If M is an internal node, we traverse the left and the right children of M recursively. Otherwise, M is a leaf node. Then, we estimate a lower bound of $Pr(N_{max})$ by assuming all instances in M dominate N_{max}, and an upper bound of $Pr(N_{min})$ by assuming no instance in M dominates N_{min}.

By traversing all partition trees of the possible dominating objects, we apply Equation (5) to compute the upper bound for $Pr(N_{min})$ and the lower bound for $Pr(N_{max})$. With the two bounds and Inequality (8), we can immediately bound the skyline probability of object U. We use only the maximum and the minimum corners of the MBBs, and never compute the skyline probability of any one in a subset of instances.

4.3 Pruning and Refinement Using Partition Trees

When one level is grown for the partition trees of all uncertain objects whose skyline memberships are not determined, the possible dominating objects of them are also partitioned to the same level. For all new leaf nodes grown in this round, we bound their probabilities by traversing the partition trees of the corresponding possible dominating objects. We note that the computation of such bounding for the leaf nodes which have the same MBB can be shared.

After that, we check whether some uncertain objects or some leaf nodes in some partition trees may be pruned. That is, their skyline probabilities do not need to be computed any more. Pruning those nodes can make the skyline computation faster.

Consider a node N in the partition tree of uncertain object U. If there exists another uncertain object $V \neq U$ such that N_{min} is dominated by V_{max}, then any instance in N cannot be in the skyline. In other words, $Pr(u) = 0$ for any $u \in N$. We do not need to compute any subset of N anymore since the instances there cannot contribute to the skyline probability of U.

Pruning Rule 5. *Let N be a node in the partition tree of uncertain object U. If there exists an object $V \neq U$ such that $V_{max} \preceq N_{min}$, then node N can be pruned.* ∎

Moreover, if $Pr(N_{min}) = Pr(N_{max})$, according to Inequality (7), the skyline probability of any instance in N is determined. N can be pruned.

Input: a set of uncertain objects S; probability threshold p;
Output: the p-skyline in S;
Method:

```
1:  SKY = ∅;
2:  FOR EACH object U ∈ S DO
3:    initialize a partition tree T_U with only the root node;
    END FOR EACH
4:  let L be the set of all partition trees; i = 0;
5:  WHILE L ≠ ∅ DO
6:    FOR EACH partition tree T_U ∈ L DO
7:      FOR EACH leaf node N of level-i in T_U DO
8:        bound Pr(N_min) and Pr(N_max); // Section 4.2
9:        bound Pr(U); // Inequality (8)
10:       IF Pr(U) ≥ p THEN
11:         SKY = SKY ∪ {U}; L = L − {T_U}; NEXT;
12:       ELSE IF Pr(U) < p THEN
13:         L = L − {T_U}; NEXT; // Pruning Rule 7
          ELSE
14:         apply Pruning Rules 5 and 6 to N if applicable;
15:         partition N to level-(i + 1) if it cannot be pruned;
        END FOR EACH
      END FOR EACH
16:   i = i + 1;
    END WHILE
17: RETURN SKY;
```

Figure 8: The top-down algorithm.

Pruning Rule 6. *Let N be a node in the partition tree of uncertain object U. If $Pr(N_{min}) = Pr(N_{max})$, then for each $u \in N$, $Pr(u) = Pr(N_{min}) = Pr(N_{max})$ and node N can be pruned.* ∎

Last, once the skyline probability of an uncertain object can be bounded at least p or less than p, then whether the object is in the p-skyline is determined. We do not need to compute the probability of this object anymore.

Pruning Rule 7. *Let p be the probability threshold. If the partition tree of an uncertain object U has l leaf nodes $N_1, \ldots, N_l$, and $N_{i,max}$ and $N_{i,min}$ are the maximum and the minimum corners of $N_i.MBB$, respectively, and $\frac{1}{|U|}\sum_{i=1}^{l} |N_i| \cdot Pr(N_{i,max}) \geq p$, then U is in the p-skyline. On the other hand, if $\frac{1}{|U|}\sum_{i=1}^{l} |N_i| \cdot Pr(N_{i,min}) < p$, then U is not in the p-skyline. In both cases, the partition tree of U can be pruned.* ∎

After the pruning step using the above rules, only the partition trees of those uncertain objects which cannot be determined in the p-skyline or not are left. In such trees, only those nodes whose skyline probabilities are not determined survive.

As the refinement steps, we partition those surviving leaf nodes and their possible dominating objects to one more level. With the refinement, the bounds of skyline probabilities are tighter.

4.4 Algorithm and Cost Analysis

The top-down algorithm is shown in Figure 8. In the implementation, we also use an R-tree to index the minimum corners of the MBBs of all objects so that the search of possible dominating objects can be conducted efficiently.

Let P be the average cost of querying partition trees of possible dominating objects for bounding the skyline probabilities of the minimum and maximum corners of MBBs, and M_{total} be the number of tree nodes whose skyline probabilities are bounded in the algorithm. Then, the average cost of the algorithm is $O(M_{total} \cdot P)$.

As shown in our experimental results, many nodes can be pruned sharply by the pruning rules. The top-down algorithm only has to grow a small number of tree nodes (i.e., M_{total} is small), and has good scalability with respect to cardinality of the data sets.

5. RELATED WORK

To the best of our knowledge, this is the first study about skyline analysis on uncertain data. Our study is related to the previous work on querying uncertain data and skyline computation.

Modeling and querying uncertain data have attracted considerable attention from database research community (see [25, 12] and the references therein). Recently, a lot of work has been engaged to management and query processing of uncertain data in sensor databases [9] and especially in spatial-temporal databases [8, 11, 28, 17].

Cheng *et al.* [9] proposed a broad classification of probabilistic queries over uncertain data, and developed novel techniques for evaluating probabilistic queries. Cheng *et al.* [8] are the first to study probabilistic range queries. They developed two auxiliary index structures to support querying uncertain intervals effectively. Tao *et al.* [28] investigated probabilistic range queries on multi-dimensional space with arbitrary probability density functions. They identified and formulated several pruning rules and proposed a new access method to optimize both I/O cost and CPU time. Dai *et al.* [11] introduced an interesting concept of *ranking* probabilistic spatial queries on uncertain data which selects the objects with highest probabilities to qualify the spatial predicates. On the uncertain data indexed by R-tree, several efficient algorithms were developed to support ranking probabilistic range queries and nearest neighbor queries. Kriegel *et al.* [17] proposed to use probabilistic distance functions to measure the similarity between uncertain objects. They presented both the theoretical foundation and some effective pruning techniques of probabilistic similarity joins.

Different from the previous work on querying uncertain data, our study introduces skyline queries and analysis to uncertain data. As shown in Sections 1 and 6, skyline queries are meaningful for uncertain data and can disclose some interesting knowledge that cannot be identified by the existing queries on uncertain data.

Computing skylines was first investigated by Kung *et al.* [18] in computational geometry. Bentley *et al.* [3] proposed an efficient algorithm with an expected linear runtime if the data distribution on each dimension is independent.

Börzsönyi *et al.* [4] introduced the concept of skylines in the context of databases and proposed a SQL syntax for skyline queries. They also developed the skyline computation techniques based on *block-nested-loop* and *divide-and-conquer* paradigms, respectively. Chomicki *et al.* [10] proposed another block-nested-loop based computation technique, SFS (*sort-filter-skyline*), to take the advantages of pre-sorting. The SFS algorithm was further significantly improved by Godfrey *et al.* [13].

The first *progressive* technique that can output skyline points without scanning the whole dataset was delveloped by Tan *et al.* [27]. Kossmann *et al.* [16] presented another progressive algorithm based on the nearest neighbor search technique, which adopts a divide-and-conquer paradigm on the dataset. Papadias *et al.* [22] proposed a *branch-and-bound* algorithm (BBS) to progressively output skyline points on datasets indexed by an R-tree. One of the most important properties of BBS is that it minimizes the I/O cost.

Variations of skyline computation have been explored, including computing skylines in a distributed environment [1, 14], continuously processing skyline queries in data streams [20, 30], computing skylines for partially-ordered value domains [5], skyline cube computation [24, 32, 31], computing subspace skylines [29], approximate skyline computation [15, 7, 6], and materializing dominance relationships [19].

All the previous studies on skyline computation and analysis focus on certain data. Our study extends the skyline computation and analysis to uncertain data. As shown in the previous section, extending skyline queries to uncertain data is far from straightforward. It involves both the development of skyline models and the design of novel algorithms for efficient computation.

6. EMPIRICAL STUDY

In this section, we report an extensive empirical study to examine the effectiveness and the efficiency of probabilistic skyline analysis on uncertain data. All the experiments were conducted on a PC with Intel P4 3.0GHz CPU and 2GB main memory running Debian Linux operating system. All algorithms were implemented in C++.

6.1 Effectiveness of Probabilistic Skylines

To verify the effectiveness of probabilistic skylines on uncertain data, we use a real data set of the NBA game-by-game technical statistics from 1991 to 2005 downloaded from `www.NBA.com`. The NBA data set contains $339,721$ records about $1,313$ players. We treat each player as an uncertain object and the records of the player as the instances of the object. Three attributes are selected in our analysis: number of points, number of assists, and number of rebounds. The larger those attribute values, the better.

Table 2 shows the 0.1-skyline players in the skyline probability descending order. We also conducted the traditional skyline analysis. We calculated the average statistics for each player on each attribute. That is, each player has only one record in the aggregate data set. We computed the skyline on the aggregate data set, which is called the *aggregate skyline* for short hereafter. All skyline players in the aggregate skyline are annotated by a "*" sign in Table 2. We obtain several interesting observations.

First, the top-12 players with the largest skyline probability are also in the aggregate skyline. All of them are great players. Those players not only have good average performance so that they are in the aggregate skyline, but also performed outstandingly in some games so that they have a high skyline probability.

Second, some players that are not in the aggregate skyline may still have a high skyline probability. There are 22 players who are not in the aggregate skyline, but have a higher skyline probability than Odom who is a skyline player in the aggregate data set.

Olajuwon is an example. In the aggregate data set, he is dominated by four other players: O'Neal, Barkley, Duncan, and Webber. In Figure 9, we sample the records of the five players with rate 5% (so that the figure is readable) and plot their number of rebounds and number of points. Olajuwon has some records (e.g., 40 points and 19 rebounds) dominate most records of other players. On the other hand, he also has some records (e.g., 0 point and 3 rebounds) that are dominated by many records of other players. Comparing to

	Name	Skyline Probability		Name	Skyline Probability		Name	Skyline Probability
*	LeBron James	0.350699		Dwyane Wade	0.199065		Steve Francis	0.131061
*	Dennis Rodman	0.327592		Tracy Mcgrady	0.198185		Dirk Nowitzki	0.130301
*	Shaquille O'Neal	0.323401	*	Grant Hill	0.191164		Paul Pierce	0.127079
*	Charles Barkley	0.309311	*	John Stockton	0.183591	*	Gary Payton	0.126328
*	Kevin Garnett	0.302531		David Robinson	0.177437		Baron Davis	0.125298
*	Jason Kidd	0.293569	*	Stephon Marbury	0.16683		Vince Carter	0.122946
*	Allen Iverson	0.269871	*	Tim Hardaway	0.166206		Antoine Walker	0.121745
*	Michael Jordan	0.250633	*	Magic Johnson	0.151813		Steve Nash	0.115874
*	Tim Duncan	0.241252	*	Chris Paul	0.149264		Andre Miller	0.11275
*	Karl Malone	0.239737		Gilbert Arenas	0.142883		Isiah Thomas	0.11076
*	Chris Webber	0.22153		Clyde Drexler	0.138993		Elton Brand	0.10966
*	Kevin Johnson	0.208991		Patrick Ewing	0.13577		Scottie Pippen	0.108941
	Hakeem Olajuwon	0.203641		Rod Strickland	0.135735		Dominique Wilkins	0.104323
	Kobe Bryant	0.200272		Brad Daugherty	0.133572	*	Lamar Odom	0.101803

Table 2: 0.1-skyline players in skyline probability descending order.

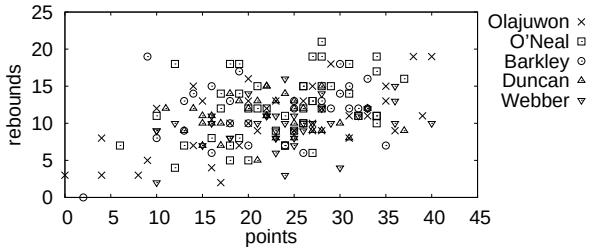


Figure 9: Olajuwon's and some other players' records.

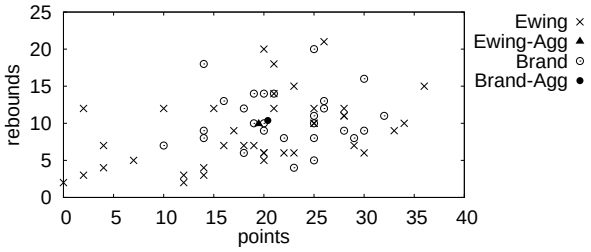


Figure 10: Ewing's and Brand's records.

the four players dominating him, Olajuwon's performance has a larger variance.

Comparing to the aggregate skyline, the probabilistic skyline finds not only players consistently performing well, but also outstanding players with large variances in performance.

Third, a player A may have a higher skyline probability than a player B who dominates A in the aggregate data set. As an example, Ewing has a higher skyline probability than Brand, though Ewing is dominated by Brand in the aggregate data set. We plot a sample with ratio 5% of both players in Figure 10. Their aggregate values are also shown in the figure. The performance of Ewing is more diverse than that of Brand. Ewing played very well in a few games, which explains why Ewing has a higher skyline probability.

In summary, probabilistic skylines disclose interesting knowledge about uncertain data which cannot be captured by traditional skyline analysis, and provide a more comprehensive view on advantages of uncertain objects than skylines using only the aggregate of such objects. Interestingly, we can rank uncertain objects using skyline probabilities, while the skyline on aggregate of uncertain data cannot reflect the differences on the opportunities of uncertain objects not to be dominated by other objects. This is another significant advantage of probabilistic skyline analysis.

6.2 Performance Evaluation

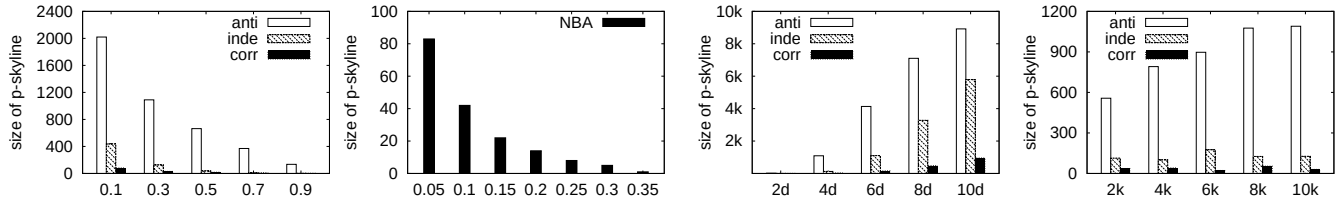
To verify the efficiency and the scalability of our algorithms, we use the NBA real data set as well as synthetic data sets in anti-correlated, independent, and correlated distributions. For the synthetic data sets, the domain of each dimension is $[0, 1]$. The dimensionality d by default is 4. The cardinality (i.e., number of uncertain objects) m by default is $10,000$. We first generated the centers of all uncertain objects using the benchmark data generator described in [4]. Then, for each uncertain object, we use the center to generate a hyper-rectangle region where the instances of the object appear. The edge size of the hyper-rectangle region follows a normal distribution in range $[0, 0.2]$ with expectation 0.1 and standard deviation 0.025. The instances of the object distributed uniformly in the region. The number of instances of an uncertain object follows uniform distribution in range $[1, l]$, where l is 400 by default. Thus, in expectation, each object has $\frac{l}{2}$ instances, and the total number of instances in a data set is $\frac{ml}{2}$ ($2,000,000$ by default). The probability threshold p is 0.3 unless otherwise specified.

6.2.1 Probabilistic Skyline Size

Figure 11 shows the size of probabilistic skylines (i.e., the number of objects in a probabilistic skyline) with respect to three important factors: the probability threshold, the dimensionality and the cardinality. Generally, anti-correlated data sets have the largest skyline size. Correlated data sets have the smallest skyline size. This is similar to the situations of skylines on certain objects. As shown in Figure 11(a), the higher the probability threshold, the smaller the skyline size. This is because a p-skyline contains a p'-skyline if $p < p'$. Figure 11(b) shows the results on the NBA data set, which is in a consistent trend. Figures 11(c) and (d) show that the skyline size increases with higher dimensionality and larger cardinality, which is also similar to the situations of skylines on certain data sets. As the dimensionality increases, the data set becomes sparser. An object has a better opportunity not to be dominated in all dimensions. As the cardinality increases, more objects may have chances not to be dominated.

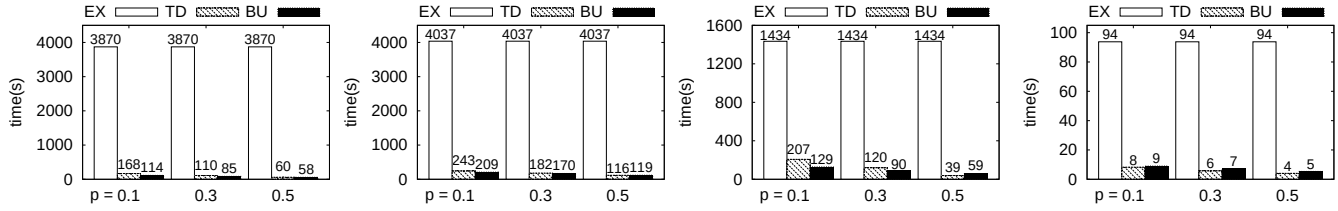
6.2.2 Efficiency and Scalability

Figure 12 investigates the runtime of the algorithms with respect to the probability threshold. The numbers on the



(a) Effect of p (b) Effect of p (c) Effect of d ($p = 0.3$) (d) Effect of m ($p = 0.3$)

Figure 11: The size of p-skyline with respect to probability threshold p, dimensionality d, and cardinality m.



(a) NBA (b) anti-correlated (c) independent (d) correlated

Figure 12: Scalability with respect to probability threshold.

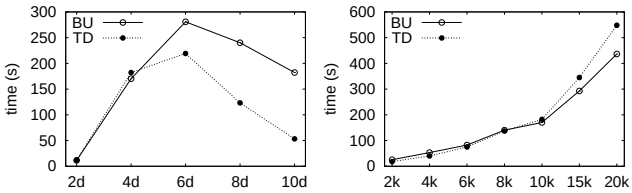
bars give the exact runtime of algorithms on the data sets. Three algorithms are evaluated. In addition to the bottom-up (BU) and the top-down (TD) algorithms described in the paper, for benchmarking we also made up an exhaustive algorithm (EX). To compute the p-skyline on a data set, without any pruning techniques, EX has to compute the skyline probability for each uncertain object. Therefore, it is insensitive to the change of probability threshold.

Both BU and TD are much faster than EX. The results clearly indicate that the pruning techniques in those two methods significantly save the cost of computing the exact skyline probabilities of many instances and objects.

Computing skylines on anti-correlated data sets is much more challenging. In the rest of this section, we focus on analyzing in detail the performance of BU and TD on anti-correlated data sets.

Figure 13 compares BU and TD with respect to dimensionality and cardinality. Both algorithms follow similar trends. The runtime of both algorithms increases when the dimensionality increases from 2 to 6, but decreases afterward. On the one hand, the cost of dominance testing between two instances, which is the basic operation in both algorithms, increases as the dimensionality increases. On the other hand, the average number of possible dominating objects for an uncertain object decreases since the data set becomes sparser when the dimensionality increases. The trend of runtime reflects the compromise of the two factors.

The higher the dimensionality, the sparser the data set. The larger the cardinality, the denser the data set. Figure 13 indicates that TD performs better when the data set is sparser. In sparse data set, the subset instances of uncertain objects may have a smaller chance to overlap, and a better chance to be pruned by some subset instances of other objects. Thus, TD has better performance. On the other hand, in dense data sets, the skyline probability of an instance may improve the bounds of the probabilities of more other instances and objects, and BU performs better.



(a) Effect of d ($p = 0.3$) (b) Effect of m ($p = 0.3$)

Figure 13: Runtime on anti-correlated data sets.

6.2.3 *Effectiveness of Pruning Techniques*

The performance of BU mainly depends on the efficiency of pruning instances and objects so that their skyline probabilities do not need to be computed. Figure 14 counts, for each object, the percentage of instances whose skyline probabilities are computed by BU. We group the objects by the percentage in 6 ranges, and count the proportion of each group in the whole data set. It is clear that more than 90% of the objects in the NBA, independent, and correlated data sets are pruned after 20% of the instances are processed. Even for anti-correlated data sets, the corresponding figure is 66%. The pruning is more effective on independent and correlated data sets. That explains the difference of runtime on synthetic data sets.

Figure 15 counts the percentage of objects pruned by pruning rules in BU (Section 3.2). Rules 3 is not counted since it prunes instances only. Every rule takes effect in some situations. Rules 4 is particularly effective on independent and correlated data sets where 84% and 97% objects are pruned, respectively. In those data sets, it is more likely that an object is completely dominated by another.

Figures 16 and 17 examine the effectiveness of the pruning techniques in TD. Figures 16(a) and (b) show the runtime of each round of partitioning on the NBA data set and the synthetic data sets, respectively. On all data sets, the run-

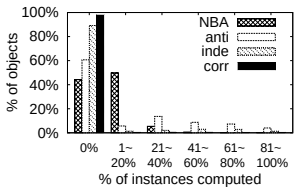


Figure 14: Pruning effect in BU.

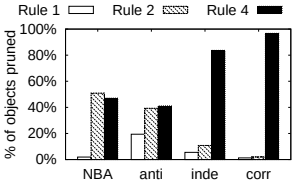
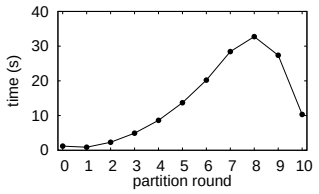
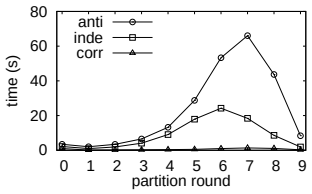


Figure 15: Effectiveness of pruning rules in BU.

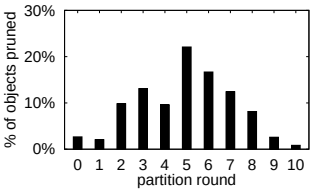


(a) NBA data set

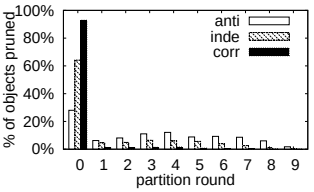


(b) Synthetic data sets

Figure 16: Pruning effect in TD – runtime in each round.



(a) NBA data set



(b) Synthetic data sets

Figure 17: Pruning effect in TD – Percentage of objects pruned in each round.

time increases at first, since after such a round the leaf nodes not pruned are partitioned into more nodes. The runtime decreases in the later rounds. This is because the effectiveness of pruning in TD becomes stronger when the leaf nodes are smaller, such that the numbers of remaining objects and nodes decrease significantly.

Figure 17 shows in each round the number of objects whose probabilistic skyline memberships are determined. On the NBA data set, rounds 2-8 prune most of the objects, while on the synthetic data sets, most of the objects are pruned in the first round. Again, the pruning is more effective on independent and correlated data sets.

In summary, our two algorithms are effective and efficient in computing probabilistic skylines. They are also scalable on large data sets containing millions of instances.

7. DISCUSSION AND CONCLUSIONS

In this paper, we extended the well-known skyline analysis to uncertain data, and developed two efficacious algorithms to tackle the problem of computing probabilistic skylines on uncertain data. Using real data sets and synthetic data sets, we illustrated the effectiveness of probabilistic skylines and the efficiency and scalability of our algorithms.

Although we focused on the discrete case, some of our ideas can be applied to handle the continuous case, i.e., each uncertain object is represented by a probability density function. For example, in the top-down algorithm, for each uncertain object, we can initially partition the space into 2 regions such that the probability of the object in each region is 0.5. Each region can be represented by a bounding box. We can estimate the skyline probabilities of the bounding boxes and recursively partition the bounding boxes into smaller ones until the skyline probabilities of uncertain objects can be determined against the threshold. Limited by space, we omit the details here.

8. REFERENCES

[1] W.-T. Balke *et al.* Efficient distributed skylining for web information systems. In *EDBT'04*.

[2] J. L. Bentley. Multidimensional binary search trees used for associative searching. *Commun. ACM*, 18(9):509–517, 1975.

[3] J. L. Bentley *et al.* On the average number of maxima in a set of vectors and applications. *JACM*, 25(4):536–543, 1978.

[4] S. Börzsönyi *et al.* The skyline operator. In *ICDE'01*.

[5] C. Y. Chan *et al.* Stratified computation of skylines with partially-ordered domains. In *SIGMOD'05*.

[6] C. Y. Chan *et al.* Finding k-dominant skylines in high dimensional space. In *SIGMOD'06*.

[7] C. Y. Chan *et al.* On high dimensional skylines. In *EDBT'06*.

[8] R. Cheng *et al.* Efficient indexing methods for probabilistic threshold queries over uncertain data. In *VLDB'04*.

[9] R. Cheng *et al.* Evaluating probabilistic queries over imprecise data. In *SIGMOD'03*.

[10] J. Chomicki *et al.* Skyline with presorting. In *ICDE'03*.

[11] X. Dai *et al.* Probabilistic spatial queries on existentially uncertain data. In *SSTD'05*.

[12] N. N. Dalvi *et al.* Efficient query evaluation on probabilistic databases. In *VLDB'04*.

[13] P. Godfrey *et al.* Maximal vector computation in large data sets. In *VLDB'05*.

[14] Z. Huang *et al.* Skyline queries against mobile lightweight devices in MANETs. In *ICDE'06*.

[15] V. Koltun *et al.* Approximately dominating representatives. In *ICDT'05*.

[16] D. Kossmann *et al.* Shooting stars in the sky: An online algorithm for skyline queries. In *VLDB'02*.

[17] H.-P. Kriegel *et al.* Probabilistic similarity join on uncertain data. In *DASFAA'06*.

[18] H. T. Kung *et al.* On finding the maxima of a set of vectors. *JACM*, 22(4):469–476, 1975.

[19] C. Li *et al.* DADA: A data cube for dominant relationship analysis. In *SIGMOD'06*.

[20] X. Lin *et al.* Stabbing the sky: Efficient skyline computation over sliding windows. In *ICDE'05*.

[21] R.J. Lipton *et al.* Practical selectivity estimation through adaptive sampling. In *SIGMOD'90*.

[22] D. Papadias *et al.* An optimal and progressive algorithm for skyline queries. In *SIGMOD'03*.

[23] J. Pei *et al.* Computing Compressed Skyline Cubes Efficiently. In *ICDE'07*.

[24] J. Pei *et al.* Catching the best views of skyline: A semantic approach based on decisive subspaces. In *VLDB'05*.

[25] A. D. Sarma *et al.* Working models for uncertain data. In *ICDE'06*.

[26] M. Sharifzadeh *et al.* The spatial skyline queries. In *VLDB'06*.

[27] K.-L. Tan *et al.* Efficient progressive skyline computation. In *VLDB'01*.

[28] Y. Tao *et al.* Indexing multi-dimensional uncertain data with arbitrary probability density functions. In *VLDB'05*.

[29] Y. Tao *et al.* SUBSKY: Efficient computation of skylines in subspaces. In *ICDE'06*.

[30] Y. Tao *et al.* Maintaining sliding window skylines on data streams. *TKDE*, 18(2):377–391, 2006.

[31] T. Xia *et al.* Refreshing the sky: The compressed skycube with efficient support for frequent updates. In *SIGMOD'06*.

[32] Y. Yuan *et al.* Efficient computation of the skyline cube. In *VLDB'05*.