Skyline Query Processing for Uncertain Data *

Mohamed E. Khalefa
University of Minnesota
Minneapolis, MN, USA
khalefa@cs.umn.edu

Mohamed F. Mokbel
University of Minnesota
Minneapolis, MN, USA
mokbel@cs.umn.edu

Justin J. Levandoski
University of Minnesota
Minneapolis, MN, USA
justin@cs.umn.edu

ABSTRACT

Recently, several research efforts have addressed answering skyline queries efficiently over large datasets. However, this research lacks methods to compute these queries over *uncertain* data, where uncertain values are represented as a range. In this paper, we define skyline queries over continuous uncertain data, and propose a novel, efficient framework to answer these queries. Query answers are probabilistic, where each object is associated with a probability value of being a query answer. Typically, users specify a probability threshold, that each returned object must exceed, and a tolerance value that defines the allowed error margin in probability calculation to reduce the computational overhead. Our framework employs an efficient two-phase query processing algorithm.

Categories and Subject Descriptors: H.2.4 Database ManagementSystems

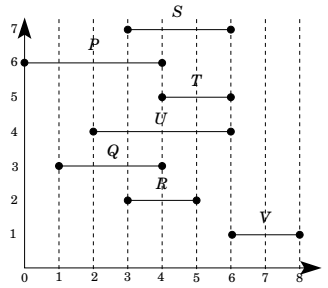
General Terms: Algorithms, Design.

1. INTRODUCTION

Since its introduction in databases [2], skyline query processing has received formidable interest in the literature (e.g., see [6, 7, 11, 15, 16]). This interest is mainly due to the skyline's importance in realizing useful, non-trivial applications ranging from multi-criteria decision-making tools to *personalized* databases [9, 12]. Given a set of multi-dimensional objects, skyline queries find the set of interesting (i.e., non-dominated) objects. An n-dimensional object P dominates another object Q if P is better than or equal to Q in $n-1$ dimensions, and strictly better than Q in at least one dimension. An example of skyline queries is *"find my best restaurants based on minimal price and minimal distance"*, which returns those restaurants that are not dominated by other restaurants based on price and distance.

*This work is supported in part by the National Science Foundation under Grants IIS-0811998, IIS-0811935, CNS-0708604, IIS-0952977 (NSF CAREER) and by a Microsoft Research Gift.

CIKM'10, October 26–30, 2010, Toronto, Ontario, Canada.



(a) Running Example

Point	Skyline
P	60%
Q	91%
R	100%
S	0.5%
T	0%
U	16%
V	100%

(b) Probabilities

Figure 1: Skyline example over data with uncertain ranges

Throughout the paper, we are using our running example consisting of seven uncertain objects presented in Figure 1a where the x dimension of each object is presented as a continuous range while the y dimension is a single certain point. We assume lower is better for all dimensions. Figure 1b gives the probability that each uncertain object is a skyline, assuming the probability density function within the uncertainty range is uniform, and lower is better for all dimensions.

With the growing number of applications that include uncertainty, e.g., sensor readings, human reading errors, and data imperfection, it became essential to support skyline queries over uncertain data. Existing work in this scope is limited only to the case where uncertain data is represented as a set of discrete values [16], i.e., a finite set of instances a_1, a_2, $\cdots$, a_n. The exact probability of object P (in Figure 1a) to be a skyline is the probability that point p in the uncertainty range of P, i.e., 0 to 4, is less than all other objects in the dataset, formally, this can be represented as $\int_0^4 f_P(x)[\Pi_{J \neq P} \int_x^\infty f_J(y)dy]dx$, where f_A is the probability density function of object A.

In this paper, we propose an efficient framework that supports skyline queries for uncertain data represented as a continuous range. The answer of the skyline query is presented, similar to Figure 1, as the probability of each object to be a skyline. Two user parameters are introduced in our framework, namely a user-defined threshold H where the user is only interested in those objects that have a probability more than H to be a skyline, and a tolerance parameter δ that represents a trade-off between the accuracy in probability calculation, and the computational overhead. We mainly propose three bounding methods, namely, *uncertainty reduction*, *pairwise comparison*, and *bound tightening*. The first two methods give an upper bound probability of each

Table 1: Related Work

Related work	Query	Continuous	Threshold	Tolerance
[5]	Range & 1-NN	√	—	—
[1]	1-NN	√	√	—
[3]	1-NN	√	√	√
[4]	k-NN	√	√	—
[14, 18]	Rank	√	—	—
[13]	Reserve Skyline	√	√	—
[19]	Top-k	—	—	—
[8]	Top-k	—	√	—
[17]	Rank	—	—	—
[16]	Skyline	—	√	—
Our work	**Skyline**	√	√	√

object to be in the skyline, while the last method gives an upper and lower bound probability.

We present a general two-phase framework that employs our proposed three bounding methods and follows a filter-refine approach, where the first phase (filtering) prunes a set of objects from consideration before reaching a more expensive second phase (refinement). In particular, Phase I is responsible on applying *uncertainty reduction* and *pairwise comparison* to derive an initial upper-bound probability of each object along with preparing a data structure that will be used in the second phase. Given this initial upper-bound value, Phase I may determine that an object can *never* be a skyline where it is immediately discarded (i.e., filtered). In Phase II, we continue pruning objects using the *bound tightening* which is iteratively applied to each object O until we either confirm that O is not in the skyline (i.e., O's probability is below the threshold H) or we have O as a skyline with the required probabilistic accuracy δ. Thus, we avoid expensive exact probability calculations until a final answer is found.

2. RELATED WORK

Table 1 divides existing work on preference queries for uncertain data with respect to: (a) the query type, (b) the uncertainty model, (c) utilizing a threshold value, and (d) utilizing a tolerance value in probability calculation.

In contrary to our work in this paper (represented in the last line of Table 1), there is no previous work that addresses the case of skyline for continuous uncertainty model. The two closest work to ours are [3] and [16]. The first one [3] supports continuous uncertainty model, threshold, and tolerance values, yet it is limited only to the simpler case of nearest-neighbor queries while our work supports skyline queries. Moreover, it is limited to one uncertain dimension. The second one [16] supports skyline queries with threshold value, however, it is limited only to the case of discrete uncertainty model while our work supports the case of continuous uncertainty model with a tolerance value in addition to the threshold value.

3. BOUNDING OBJECT PROBABILITY

As computing the exact probability for each object to be in the skyline set $\mathcal{S}$ is prohibitively expensive [10], we present three methods that gradually bound the probability of each object to be in $\mathcal{S}$. These bounds are used to early prune objects that are not of interest, i.e., those objects that have a probability less than H. The main objective is to calculate the exact probability, within tolerance δ, to only those objects that will have a probability more than H. These three methods are: (1) *Uncertainty reduction* (Section 3.1) aims to place an upper bound on the object probability to be in $\mathcal{S}$ while reducing the object uncertainty region, (2) *Pairwise comparison* (Section 3.2) gives a tighter upper bound probability, and (3) *Adaptive Bound tightening* (Section 3.3) iteratively tightens the lower and upper bound probabilities of an object to be within the required tolerance δ.

3.1 Uncertainty Reduction

Affected Objects. Uncertainty reduction is computed in a *pairwise* fashion. An ordered pair of objects (Q,P) qualifies to *uncertainty reduction* only if the *endpoint* of Q dominates the *endpoint* of P. The endpoint e_P of an object P is formulated by substituting P's uncertainty range by the upper value on each uncertain dimension, formally $e_P = (d_1^u, d_2^u, \cdots, d_i^u, d_{i+1}, \cdots, d_m)$. Notice that it is straightforward to determine the dominance relation between e_P and e_Q because both of them are exact points. As the dominance relation is not reflexive, if (Q,P) qualifies for *uncertainty reduction*, then (P,Q) does not qualify.

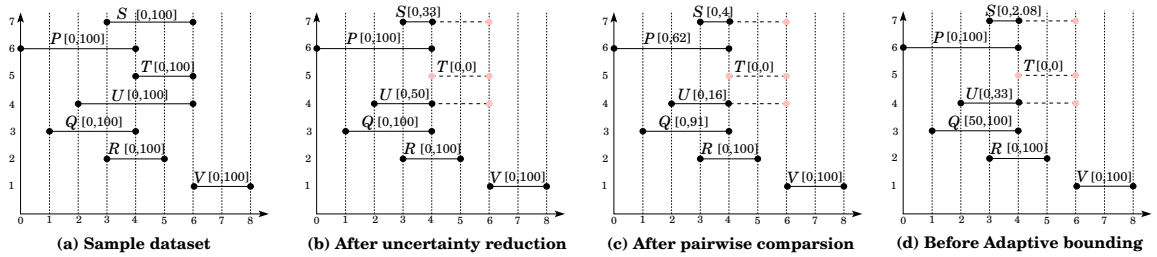
Main Idea. The main idea of *uncertainty reduction* is to reduce the upper bound probability for an object P by removing a portion of its uncertainty range in which if P exists in, it would have a zero probability being a skyline object. In other words, *uncertainty reduction* is applied to the dominated object where we: (a) limit its upper bound probability, and (b) reduce the uncertainty range $[d_1^l\text{-}d_1^u] \times [d_2^l\text{-}d_2^u] \times \ldots \times [d_i^l\text{-}d_i^u]$ to a smaller range.

Example. As an example, we apply the *uncertainty reduction* over all the uncertain objects in Figure 2a starting from V with the lowest certain value. For object V, as the endpoint of the uncertainty range e_V does not dominate any other endpoint, V does not result in any uncertainly reduction for any other object. For object R, e_R dominates e_U, e_T, and e_S, so, the pairs (R,U), (R,T), and (R,S) qualify for *uncertainty reduction*. This results in reducing the uncertainty range of U to be [2-5] instead of [2-6]. Since the reduced range is one quarter of the original range, the upper bound probability of U is set to 75%. Similarly, the uncertainty ranges of T and S are reduced to [4-5] and [3-5] with an upper bound probability of 50% and 66%, respectively. Figure 2b gives the result of all points after the *uncertaintyreduction* with their upper probability bounds, pruning objects S, and T from any further processing for 50% threshold.

3.2 Pairwise Comparison

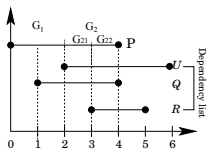
Affected Objects. Similar to the case of *uncertainty reduction*, *pairwise comparison* is computed in a *pairwise* fashion. However, the condition that two objects qualify for pairwise comparison is different from that of the uncertainty reduction where an ordered pair of objects (Q,P) qualifies to *pairwise comparison* only if: (a) the uncertainty ranges of P and Q overlap, and (b) the certain part of Q either is equal to or dominates the certain part of P. The certain part of object P, $C(P)$, is formulated by removing P's uncertain dimensions, formally, $C(P)$=$\{d_{i+1}, \ldots, d_n\}$. Then, the upper bound probability of object P can be bounded. As the qualifying condition includes equality, it could be the case that both ordered pairs (P,Q) and (Q,P) qualify for *pairwise comparison*. This case takes place if $C(P) = C(Q)$.

Main Idea. The main idea of *pairwise comparison* is that for a qualified ordered pair of objects (Q,P), since Q already dominates or equal to P in the certain dimensions, if Q would also dominate P in the uncertain dimension, then P would have no chance to be in the answer set, and it will be discarded using the uncertainty reduction. This means that



(a) Sample dataset (b) After uncertainty reduction (c) After pairwise comparsion (d) Before Adaptive bounding

Figure 2: Bounding Objects Probabilities



(a) Spliting

Split at 2

G_i	s_i	e_i	lower	upper	error
G_1	0	2	0.67	1	0.165
G_2	2	4	0	0.67	0.335

Split at 3

G_i	s_i	e_i	lower	upper	error
G_1	0	2	0.67	1	0.165
G_{12}	2	3	0.25	0.67	0.105
G_{22}	3	4	0	0.25	0.06

(b) Bound tightening

Figure 3: Segmentation of Object P

the only chance for P to still be a skyline object is to be *not* dominated by Q in the uncertain dimension. Thus, we can have an upper bound probability for object P to be in the answer set as the probability that the uncertain range of P is not dominated by the uncertain range of Q. if the certain dimensions of Q is equal to those of P, we compute an upper bound probability of Q.

To compute a tighter upper bound probability for object P for a qualified pair (Q,P), we compose uncertain object QP to be from the start point of object Q, Q_s, to the start point of object P, P_s. let Q_{dom} be the portion of Q that fully dominates P, i.e., Q_{dom}= $QP \cap Q$. The upper bound of P can be computed that the probability that object Q is in Q_{dom} multiplied by the upper bound of P. If this computed upper bound is lower than the user-given threshold, H, we discard P. Otherwise, we proceed by iterating over all uncertain dimensions and finding the minimum upper bound of object P.

Example. Continuing the example shown in Figure 2b. Object V does not qualify for *pairwise comparison* with any other object as its uncertainty range does not overlap with others. For object R, the ordered pairs (R,Q), (R,U), (R,P), and (R,S) qualify for *pairwise comparison*. For (R,Q), the upper bound probability of Q is reduced to 1-$\frac{1}{2}*\frac{1}{3}*\frac{1}{2}$=91%. Similarly, U upper bound is reduced to 31%, P upper bound is reduced to 87%, and S upper bound is reduced to 25%. Figure 2c gives the result of all points after the *pairwisecomparisons* with their upper probability bounds, where object U is discarded.

3.3 Adaptive Bound Tightening

Affected Objects. In contrast to *uncertainty reduction* and *pairwise comparison* that are applied for a pair of objects, *Adaptive bound tightening* is applied for a given object P and a list of objects DL_P, termed the *dependency list* of P. An object Q belongs to DL_P if and only if the pair (Q,P) qualifies for *pairwise comparison*. In other words, DL_P includes all objects in the data set that affect the probability of P being a skyline object. For example, in Figure 2a, $DL_P = \{R, Q, U\}$.

Main Idea. We divide the uncertain object P into segments adaptively to get better accuracy bounds. Initially, we set the segment to be the whole uncertainty range. To do so in a simple and efficient way, we choose the segment that is causing the largest difference between the upper and lower bounds for uncertain object P. And then we *split* it into halves along the dimension that is causing the largest difference in probability. This will increase the number of segments by one more segment, it will also tighten the current probability bounds [$P_{lower} - P_{upper}$]. We keep doing so, iteratively, till we reach to the stopping condition, ($P_{upper} < H$ OR ($P_{lower} > H$ AND $P_{upper} - P_{lower} < \delta$). The faster we approach to the stopping condition, the more efficient our scheme will be. So, *bound tightening* always selects the segment to split, G_s, as the one that has the largest weighted difference in its upper and lower bound probabilities, i.e., $G_s = MAX_{\forall G_i}((G_{i_{upper}} - G_{i_{lower}}) * Pr\{P \in G_i\})$. By doing so, *bound tightening* will reach to its stopping conditions in less iterations. It is important to note that splitting G_s into two halves G_{s1} and G_{s2} results in calculating only the probability $G_{s1_{lower}}$ as it is also equal to $G_{s2_{upper}}$ for each uncertain dimension, choosing the dimension that reduce the weighted difference for G_{s1} and G_{s2}. The other bounds are inherited from G_s where $G_{s1_{upper}} = G_{s_{upper}}$ and $G_{s2_{lower}} = G_{s_{lower}}$. Finally, P_{lower} and P_{upper} are updated incrementally to reflect the new probability bounds.

Example. Consider object P in Figure 3, the last column in the table of Figure 3b gives the difference between the upper and lower bound probability for each segment of object P multiplied with the probability of the segment after splitting P at 2 then at 3. We divide P at 2. Then, of all segments, G_2 has the largest calculation error (i.e., $(0.67 - 0.0) * \frac{1}{2} = 0.335$). Thus, G_2 is chosen to be split into two halves G_{21} and G_{22}. Then, only *one* probability value needs to be calculated, $G_{21_{lower}} = G_{22_{upper}}$ =0.25. Consecutively, P_{lower} and P_{upper} are updated to be 39.8% and 73%, respectively.

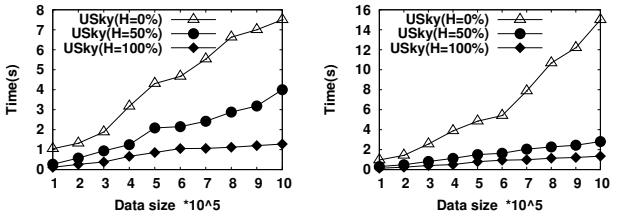
4. SKYLINE QUERY PROCESSING FOR UNCERTAIN DATA

Phase I: Skyline and Dependency Lists Phase I scans the input data set and applies the *uncertainty reduction* over each object Q against existing objects that are candidate to be skylines, i.e., not yet pruned by our algorithm. We keep tracks of objects with upper bound less than given threshold, denoted as $ThresholdDominated$ set. Based on the result of the *uncertainty* reduction, if Q's upper bound probability becomes less than H, we immediately prune Q, otherwise, we apply *pairwise comparison* on Q against each candidate object P. As *pairwise comparison* may affect the probability of both Q and P, we check if the probability of any of them becomes less than H. If this is the case, we prune such object. Finally, we compute the *dependency list* (P_{DL}) for each object P, and add it to the candidate skyline set.

Example. Table 2 gives the result of applying Phase I to

Table 2: Example for Phase I

	Skyline List	ThDom
U	(U,100%,{})	{}
R	(R,100%,{}), (U,50%,{R})	{}
S	(R,100%,{}), (U,50%,{R})	{S}
V	(V,100%,{}), (R,100%,{}), (U,50%,{R})	{S}
Q	(V,100%,{}), (R,100%,{}), (Q,91%,{R})	{U,S}
T	(V,100%,{}), (R,100%,{}), (Q,91%,{R})	{U,S}
P	(V,100%,{}), (R,100%,{}), ((Q,91%,{R}), (P,62%,{R, Q})	{U,S}



(a) One uncert. dimension (b) Three uncert. dimensions

Figure 4: Scalability of USky: One & Three uncertain dimensions

our running example in Figure 2a when $H = 50\%$ and the data is read in the order of U, R, S, V, Q, T, and P.

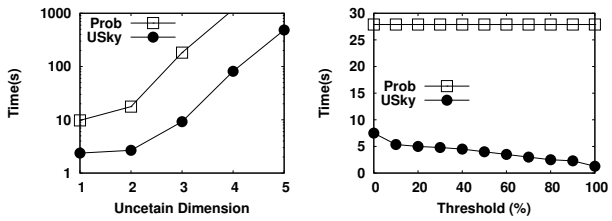
Phase II: Final Probability Calculation Phase II starts after the completion of Phase I and scans each object Q in the candidate skyline set. If the dependency list of Q is empty, we immediately add Q to the final answer set with a probability 100%, otherwise, we proceed with computing a lower and upper bound probability for Q. If the difference between the upper and lower bound probabilities of Q is within the accepted tolerance δ, we report Q as an answer along with its probabilities. If we still did not achieve the required accuracy, we iteratively apply the *bound tightening* method until we conclude that either Q is not an answer (i.e., its upper bound probability is less than H), or it is an answer with an accepted accurate probability calculation within δ.

Example. Continuing on *bound tightening* for P, we will need 10 iterations to have $P_{lower} = 57\%$ and $P_{upper} = 61\%$. Object Q needs 6 iterations to have $Q_{lower} = 89.5\%$ and $Q_{upper} = 93.7\%$.

5. EXPERIMENTAL RESULTS

Figure 4 gives the effect of the various threshold values (0%, 50%, and 100%) on our proposed algortihm denoted as USky as dataset sizes increase. Even for the case of $H = 0\%$, USky performance is three times better without adaptive bound tightening. When $H = 100\%$, for two uncertain dimensions, USky runtime is 1.3 seconds in comparsions to 8 and 2.8 seconds for H= 0% and 50%, respectively, as it immediately prunes objects that are dependent on other objects found using *pairwise comparison*. For 50% threshold, the performance is close to 100% as the number of points with thresholds between 50% and 100% are close.

We compare our proposed algorithm USky without adaptive bound tightening, we denote this variant as Prob. Figure 5a gives the effect of increasing the threshold H for our synthetic. The speed up of USky over Prob reaches up to 28 for a 100% threshold and three uncertain dimensions. Figure 5b studies the effect of increasing the number of uncertain dimensions from one to five, while having one certain dimension, on the total runtime (presented in log scale) for the USky and Prob algorithms. For all number of uncertain dimensions, USky has about an order of magnitude better performance than Prob.



(a) Uncertain Dimensions (b) Threshold

Figure 5: Effect of dimensionality and threshold

6. CONCLUSION

We have defined skyline queries over continuous uncertain data, and proposed a novel, efficient framework to answer these queries. Query answers are probabilistic, where each object is associated with a probability value of being in skyline objects. Users can specify a probability threshold, that each object in the answer set must exceed, and a tolerance that defines the allowed error margin in probability calculation. We have described our framework in the context of *skyline* in which we have proposed three methods to bound each object probability for being a preferred object, namely, we have proposed *uncertainty reduction*, *pairwise comparison*, and *bound tightening*. Then, we presented a two-phase framework that encapsulates our three proposed methods together using a filter-refine approach.

7. REFERENCES

[1] G. Beskales, M. A. Soliman, and I. F. Ilyas. Efficient Search for the Top-k Probable Nearest Neighbors in Uncertain Databases. In *VLDB*, 2008.

[2] S. Börzsönyi, D. Kossmann, and K. Stocker. The Skyline Operator. In *ICDE*, 2001.

[3] R. Cheng, J. Chen, M. F. Mokbel, and C.-Y. Chow. Probabilistic Verifiers: Evaluating Constrained Nearest-Neighbor Queries over Uncertain Data. In *ICDE*, 2008.

[4] R. Cheng, L. Chen, J. Chen, and X. Xie. Evaluating probability threshold k-nearest-neighbor queries over uncertain data. In *EDBT*, 2009.

[5] R. Cheng, D. V. Kalashnikov, and S. Prabhakar. Evaluating Probabilistic Queries over Imprecise Data. In *SIGMOD*, 2003.

[6] J. Chomicki, P. Godfrey, J. Gryz, and D. Liang. Skyline with Presorting. In *ICDE*, 2003.

[7] P. Godfrey, R. Shipley, and J. Gryz. Maximal vector computation in large data sets. In *VLDB*, 2005.

[8] M. Hua, J. Pei, W. Zhang, and X. Lin. Efficiently Answering Probabilistic Threshold Top-k Queries on Uncertain Data. In *ICDE*, 2008.

[9] W. Kießling. Foundations of Preferences in Database Systems. In *VLDB*, 2002.

[10] C. Koch and D. Olteanu. Conditioning Probabilistic Databases. In *VLDB*, 2008.

[11] D. Kossmann, F. Ramsak, and S. Rost. Shooting Stars in the Sky: An Online Algorithm for Skyline Queries. In *VLDB*, 2002.

[12] G. Koutrika and Y. E. Ioannidis. Personalization of Queries in Database Systems. In *ICDE*, 2004.

[13] X. Lian and L. Chen. Monochromatic and bichromatic reverse skyline search over uncertain databases. In *SIGMOD*, 2008.

[14] X. Lian and L. Chen. Probabilistic ranked queries in uncertain databases. In *EDBT*, 2008.

[15] D. Papadias, Y. Tao, G. Fu, and B. Seeger. Progressive skyline computation in database systems. *TODS*, 30(1):41–82, 2005.

[16] J. Pei, B. Jiang, X. Lin, and Y. Yuan. Probabilistic Skylines on Uncertain Data. In *VLDB*, 2007.

[17] C. Re, N. Dalvi, and D. Suciu. Efficient top-k query evaluation on probabilistic data. In *ICDE*, 2007.

[18] M. A. Soliman and I. F. Ilyas. Ranking with uncertain scores. In *ICDE*, 2009.

[19] M. A. Soliman, I. F. Ilyas, and K. C.-C. Chang. Top-k Query Processing in Uncertain Databases. In *ICDE*, 2007.