IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661, p-ISSN: 2278-8727
PP 53-57
www.iosrjournals.org

Semantics Representation of Probabilistic Data by Using Topk-Queries for Uncertain Data

[1]R.G.NishaaM.E (SE),[2] N.GayathriM.E(SE)
[1] Saveetha engineering college, [2]SSN engineering college

Abstract: Database systems for uncertain and probabilistic data promise to have many applications. Query processing on uncertain data occurs in the contexts of data warehousing, data integration, and of processing data extracted from the Web. Data cleaning can be fruitfully approached as a problem of reducing uncertainty in data and requires the management and processing of large amounts of uncertain data. Decision support and diagnosis systems employ hypothetical queries. Scientific databases, which store outcomes of scientific experiments, frequently contain uncertain data such as incomplete observations or imprecise measurements. Sensor and RFID data is inherently uncertain. Applications in the contexts of fighting crime or terrorism, tracking moving objects, surveillance, and plagiarism detection essentially rely on techniques for processing and managing large uncertain datasets. Beyond that, many further potential applications of probabilistic databases exist and will manifest themselves once such systems become available. Frequent items in a large data set are commonly referred to as heavy hitters. More precisely, the heavy hitters in a data set are those items whose relative frequency exceeds a specified threshold. We can easily adapt our algorithms to answer top-k queries, returning the k items with the largest confidence of being heavy hitters.

Keywords: Uncertain data, knowledge representation Expected rank, topk values.

I. INTRODUCTION ABOUT THE KNOWLEDGE REPRESENTATION:

The semantics of ranking in uncertain databases are unclear, due to the fact that both scores and probabilities of tuples must be accounted for in the ranking. For example, it is unclear whether it is better to report highly ranked items with a relatively low probability of existence or a lower-ranked set of items with a high probability of existence. Thus, the definition of the semantics of top-*k* queries when the data is uncertain is an important issue. Finding frequent items in probabilistic data turn out to be much more difficult. We first propose exact algorithms for offline data with either quadratic or cubic time. Next, we design novel sampling-based algorithms for streaming data to find all approximately likely frequent items with theoretically guaranteed high probability and accuracy. To consume sub linear memory and exhibit excellent scalability. The effectiveness and efficiency of our algorithms using both real and synthetic data sets with extensive experimental evaluations. Ranking queries are a powerful concept in focusing attention on the most answers to a query. To deal with massive quantities of data, such as multimedia search, streaming data, web data and distributed systems, tuples from the underlying database are ranked by a score, usually computed based on a user defiend scoring function. Data items in the output of such operations are usually associated with a confidence, reflecting how well they are matched with other records from different data sources. In applications that handle measurement data, e.g., sensor readings and distances to a query point, the data is inherently noisy, and is better represented by a probability distribution rather than a single deterministic value.

II. PROPOSED SYSTEM:

The data's of uncertain, probably of measuring system the result for the query is uncertain. For e.g. while determining distance between the two query points the result will be uncertain, so it's better to determine the result based on the probabilistic database management systems. In this project the results for the massive data query is determined using both the ranking approach and probability database approach.Exact-k: In this the tuple present in a location of k-list should not vary with k'-list conntainment: The top-(k+1) list should contain all items in top-k list.Unique: The tuple should be giv a unique location.Stability: A result present in the top order should not be removed from list.Value Invariance: The scores should determine the behavior of the tuple not its order.

III. CALCULATION OF TOP-K PROBABILITIES OF PATH:

Now, we present how to compute top-k probabilities of a path, i.e., PrkðPÞ, offline. For a keyword indexed in g, we use BLINKS to determine distinct root paths E ¼ fP1; . . . ;Png3 that reach the keyword, in a nonincreasing score order.Each path Pi 2 E has an existence probability, and thus, weuse the traditional top-k algorithms for uncertain relationaldatabases to compute. These methods all assume tuples or attributes probabilistic independence. However, paths in E overlap and, thus, the existence probabilities are correlated. We should develop a new approach to calculate PrkðPÞ. In this section, we propose an efficient sampling algorithm to calculate pruning.Consider an uncertain graph g with edge set E and vertex set V . Associated each edge e 2 E with a Boolean variable Xe. The probability.

IV. IMPLEMENTATION

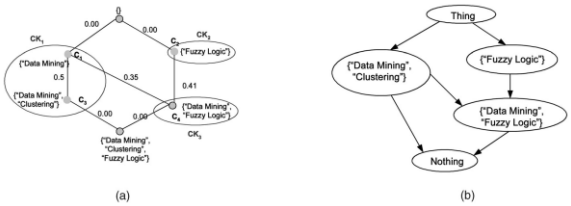
Module is a bounded contiguous group of statements having a single name and that can be treated as a unit. In other words,a single block in a pile of blocks. Make sure modules perform a single task, have a single entry point, and have a single exit point. Isolate input output routines into a small number of standard modules that can be shared system wide. Isolate system dependent functions in the application to ease possible future conversions to other computer platforms or to accommodate future operating system revisions. The main modules are: Exact-k. The top-k list should contain exactly k items.. Containment. The top-ðk þ 1Þ list should contain all items in the top-k.. Unique ranking. Within the top-k, each reported item should be assigned exactly one position: the sameitem should not be listed multiple times within thetop-k.. Stability. Making an item in the top-k list more likelyor more important should not remove it from the list. Value invariance. The scores only determine the relative behavior of the tuples: changing the scorevalues without altering the relative ordering should not change the topk.

Ranking method	Exact-k	Containment	Unique-Rank	Value-Invariant	Stability
U-top$k$ [37]	×	×	✓	✓	✓
U-kRanks [37], [25]	✓	✓	×	✓	×
PT-k [18]	×	weak	✓	✓	✓
Global-top$k$ [43]	✓	×	✓	✓	✓
Expected score	✓	✓	✓	×	✓
Expected rank	✓	✓	✓	✓	✓

Ranking methods for uncertain data

V. PRUNING ALGORITHM:

The pruning algorithm scans the tuples in order. After seeing tn, it can compute exactly using E½jWj_ and qn in Oð1Þ time based on (8). It also maintains r, the kth smallest among all the tuples that have been retrieved. This can be done with a priority queue in time per tuple. A lower bound on K for any ‘ > n is computed as follows Thus, when qn _ 1, we know for sure that there are at least k tuples among the first n with expected ranks smaller than all unseen tuples.



At this point, we can safely terminate the search. In addition, recall that for all the scanned tuples, their expected ranks are calculated exactly by (8). Hence, this algorithm—which we dub T-ERank- Prune—can simply return the current top-k tuples. From theabove analysis, its time cost is O log kÞ where n is potentially much smaller than N. The important properties that guide the definition of ranking queries in deterministic databases and analyze characteristics of existing top-k ranking queries for probabilistic data. These properties naturally lead to the ranking approach that is based on the rank distribution for a tuple across all possible worlds in an uncertain domain. Efficient algorithms for two major uncertainty models ensure the practicality of our approach. The project demonstrate that ranking by expected ranks is efficient in both attribute-level and tuple-level uncertainty models.

tuples	score
t_1	$\{(100, 0.4), (70, 0.6)\}$
t_2	$\{(92, 0.6), (80, 0.4)\}$
t_3	$\{(85, 1)\}$

world W	$\Pr[W]$
$\{t_1 = 100, t_2 = 92, t_3 = 85\}$	$0.4 \times 0.6 \times 1 = 0.24$
$\{t_1 = 100, t_3 = 85, t_2 = 80\}$	$0.4 \times 0.4 \times 1 = 0.16$
$\{t_2 = 92, t_3 = 85, t_1 = 70\}$	$0.6 \times 0.6 \times 1 = 0.36$
$\{t_3 = 85, t_2 = 80, t_1 = 70\}$	$0.6 \times 0.4 \times 1 = 0.24$

An example of possible worlds for attribute-level uncertainty

tuples	score	$p(t)$
t_1	100	0.4
t_2	92	0.5
t_3	80	1
t_4	70	0.5

rules	
τ_1	$\{t_1\}$
τ_2	$\{t_2, t_4\}$
τ_3	$\{t_3\}$

world W	$\Pr[W]$
$\{t_1, t_2, t_3\}$	$p(t_1)p(t_2)p(t_3) = 0.2$
$\{t_1, t_3, t_4\}$	$p(t_1)p(t_3)p(t_4) = 0.2$
$\{t_2, t_3\}$	$(1 - p(t_1))p(t_2)p(t_3) = 0.3$
$\{t_3, t_4\}$	$(1 - p(t_1))p(t_3)p(t_4) = 0.3$

An example of possible worlds for tuple-level uncertainty model. Comparison of Expected and Median Ranks We also studied the similarity of top-k lists for the expected and median ranks on different data sets as k varies. We extend the averaging Kendall distance from [12] and define the normalized averaging Kendall distance which we use in our comparisons. In general, our results show that the top-k lists for both median and expected ranks are rather different which show median and expected ranks enable us toemphasize different characteristics of a rank distribution. We also study the similarity of the top-k lists of the median ranks and different quantile ranks as k varies. We observe that this similarity is very stable, with quantiles closer to the median having more similar top-k lists. The complet

i.Segmentation:sensor reading

V. CONCLUSION

We have studied semantics of ranking queries in probabilistic data. We adapt important properties that guide the definition of ranking queries in deterministic databases and analyze characteristics of existing top-k ranking queries for probabilistic data. These properties naturally lead to the ranking approach that is based on the rank distribution for a tuple across all possible worlds in an uncertain domain. Efficient algorithms for two major uncertainty models ensure the practicality of our approach. Our experiments demonstrate that ranking by expected ranks, median ranks, and quantile ranks is efficient in bothattribute-level and tuple-level uncertainty models.

INPUT:
Sensor reading for temperature**:**

EXPECTED RANK:

REFERENCES

[1]. P. Agrawal, O. Benjelloun, A. Das Sarma, C. Hayworth, S. Nabar,T. Sugihara, and J. Widom, "Trio: A System for Data,Uncertainty, and Lineage," Proc. Int'l Conf. Very Large Data Bases (VLDB), 2006.
[2]. L. Antova, C. Koch, and D. Olteanu, "10106 Worlds and Beyond:
[3]. Efficient Representation and Processing of Incomplete Information,"Proc. IEEE Int'l Conf. Data Eng. (ICDE), 2007.
[4]. L. Antova, T. Jansen, C. Koch, and D. Olteanu, "Fast and Simple Relational Processing of Uncertain Data," Proc. IEEE Int'l Conf.
[5]. Data Eng. (ICDE), 2008. O. Benjelloun, A.D. Sarma, A. Halevy, and J. Widom, "ULDBs:
[6]. Databases with Uncertainty and Lineage," Proc. Int'l Conf. Very Large Data Bases (VLDB), 2006.
[7]. T. Bernecker, H.-P. Kriegel, M. Renz, F. Verhein, and A. Zuefle, "Probabilistic Frequent Itemset Mining in Uncertain Databases,"
[8]. Proc. ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (KDD), 2009.
[9]. G. Beskales, M.A. Soliman, and I.F. Ilyas, "Efficient Search for the Top-k Probable Nearest Neighbors in Uncertain Databases," Proc.
[10]. VLDB Endowment, vol. 1, pp. 326-339, 2008.
[11]. S. Chaudhuri, K. Ganjam, V. Ganti, and R. Motwani, "Robust and Efficient Fuzzy Match for Online Data Cleaning," Proc. ACM
[12]. SIGMOD Int'l Conf. Management of Data (SIGMOD), 2003.
[13]. R. Cheng, D. Kalashnikov, and S. Prabhakar, "Evaluating Probabilistic Queries over Imprecise Data," Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD), 2003.
[14]. N. Dalvi and D. Suciu, "Efficient Query Evaluation on Probabilistic Databases," VLDB J., vol. 16, no. 4, pp. 523-544, 2007.
[15]. A. Deshpande, C. Guestrin, S. Madden, J. Hellerstein, and W. Hong, "Model-Driven Data Acquisition in Sensor Networks,"
[16]. Proc. Int'l Conf. Very Large Data Bases (VLDB), 2004. R. Fagin, A. Lotem, and M. Naor, "Optimal Aggregation
[17]. Algorithms for Middleware," Proc. ACM SIGMOD-SIGACTSIGART Symp. Principles of Database Systems (PODS), 2001.
[18]. R. Fagin, R. Kumar, and D. Sivakumar, "Comparing Top k Lists," Proc. ACM-SIAM Symp. Discrete Algorithms, 2003.

[19]. A. Fuxman, E. Fazli, and R.J. Miller, "ConQuer: Efficient Management of Inconsistent Databases," Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD), 2005.
[20]. [T. Ge, S. Zdonik, and S. Madden, "Top-k Queries on Uncertain Data: On Score Distribution and Typical Answers," Proc. SIGMOD
[21]. Int'l Conf. Management of Data (SIGMOD), 2009. A. Halevy, A. Rajaraman, and J. Ordille, "Data Integration: The Teenage Year," Proc. Int'l Conf. Very Large Data Bases (VLDB),2006.
[22]. M.A. Hernandez and S.J. Stolfo, "Real-World Data Is Dirty: Data Cleansing and the Merge/Purge Problem," Data Mining and Knowledge Discovery, vol. 2, no. 1, pp. 9-37, 1998.
[23]. M. Hua, J. Pei, W. Zhang, and X. Lin, "Efficiently Answering Probabilistic Threshold Top-k Queries on Uncertain Data," Proc. IEEE Int'l Conf. Data Eng. (ICDE), 2008.
[24]. 1916 IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 23, NO. 12 DECEMBER 2011M. Hua, J. Pei, W. Zhang, and X. Lin, "Ranking Queries on Uncertain Data: A Probabilistic Threshold Approach," Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD), 2008.
[25]. I.F. Ilyas, W.G. Aref, A.K. Elmagarmid, H. Elmongui, R. Shah, and J.S. Vitter, "Adaptive Rank-Aware Query Optimization in Relational Databases," ACM Trans. Database Systems, vol. 31, pp. 1257- 1304, 2006.
[26]. I.F. Ilyas, G. Beskales, and M.A. Soliman, "Survey of Top-k Query Processing Techniques in Relational Database Systems," ACM
[27]. Computing Surveys, vol. 40, pp. 1-58, 2008.
[28]. R. Jampani, F. Xu, M. Wu, L.L. Perez, C.M. Jermaine, and P.J. Haas, "MCDB: A Monte Carlo Approach Managing Uncertain Data," Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD), 2008.
[29]. B. Kanagal and A. Deshpande, "Online Filtering, Smoothing and Probabilistic Modeling of Streaming Data," Proc. IEEE Int'l Conf. Data Eng. (ICDE), 2008.