Efficient and Scalable Monitoring and Summarization of Large Probabilistic Data

Mingwang Tang
(Supervised by Feifei Li)
School of Computing
University of Utah, Salt Lake City, Utah, USA
tang@cs.utah.edu
Expected graduation date: Fall 2013

ABSTRACT

In numerous real applications, uncertainty is inherently introduced when massive data are generated. Modern database management systems aim to incorporate and handle data with uncertainties as a first-class citizen, where uncertain data are represented as probabilistic relations. In my thesis, my work has focused on monitoring and summarization of large probabilistic data. Specifically, we extended the distributed threshold monitoring problem to distributed probabilistic data. Instead, we actually need to monitor the aggregated value (e.g. sum) of distributed probabilistic data against both the score threshold and the probability threshold, which make the techniques designed for deterministic data are not directly applicable. Our algorithms have significantly reduced both the communication and computation costs as shown by an extensive experimental evaluation on large real datasets. On the other hand, building histograms to summarize the distribution of certain feature in a large data set is a fundamental problem in data management. Recent work have extended this studies to probabilistic data [6, 7] but their methods suffer from the limited scalability. We present novel methods to build scalable histograms over large probabilistic data using distributed and parallel algorithms. Extensive experiments on large real data sets have demonstrated the superb scalability and efficiency achieved by our implementations in MapReduce, when compared to the existing, state-of-the-art centralized methods.

Categories and Subject Descriptors

H.4 [**Data Management**]: Systems-Distributed probabilistic data

General Terms

Algorithms, Experimentation, Performance

Keywords

monitor; summarize; uncertainty; probabilistic data

SIGMOD'13 PhD Symposium, June 23, 2013, New York, NY, USA.

1. INTRODUCTION

In numerous applications, uncertainty and imprecision exists in the data due to various reasons. For instance, sensor/RFID readings are inherently noisy in measurement [4, 11, 16]; data integration produces fuzzy matches [12, 30] and many more. In the era of big data, massive data are produced or collected across different applications, cloud, and science domains, which only increases uncertainties that come with data. Many research efforts have been devoted to explicitly represent and manage data with uncertainty in probabilistic database management systems (PDBMSs), such as MystiQ [3], Trio [1], MayBMS [24], SPROUT^2 [14], Orion [29], among others.

In a probabilistic database, uncertainties in tuple level or attribute level are typically modeled by probability distribution functions (pdfs). Such a probabilistic database characterizes a probability distribution of an exponential number of possible worlds and each possible world is an realization (deterministic instance) of the probabilistic database. Meanwhile, the query result on a probabilistic database essentially determines a distribution of possible query answers across all possible worlds. Given this possible worlds semantic, the goal of my thesis mainly focus on two instance problems on monitoring and summarization of large probabilistic data respectively.

Following the guidance from my advisor, Professor Feifei Li, I extend the study of distributed threshold monitoring problem (DTM) to probabilistic data. The most natural and popular way of extending threshold queries to probabilistic data is probabilistic-threshold semantics [4,9,30], which introduces another threshold on the probability of the query answer in addition to the threshold on the score value of the results. We refer to them as the *distributed probabilistic threshold monitoring* (DPTM) problem. That said, we have to continuously monitor the distributed probabilistic data against two thresholds in DPTM, which bring up challenges to reduce both the computation and communication costs.

Recently, Cormode and Garofalakis has extended the definition of the well-known V-optimal histogram (a form of bucketization over a set of one dimension values) [19], and wavelet histogram [25] to probabilistic data [7], followed by the work by Cormode and Deligiannakis [6]. They have explored both the *expectation-based* [7] and the *probabilistic-based* [6] semantics of extending histogram definitions in certain data to data with uncertainty. Note that histogram construction can be an expensive operation, even for certain data, e.g., the exact algorithm for building a V-optimal histogram is based on a dynamic programming formulation, which runs in $O(kN^2)$ for constructing k buckets over N data values [19]. Not surprisingly, building histograms as such on probabilistic data is even more expensive, when one has to work with the (expo-

nential) number of possible worlds. Thus, existing methods [6, 7], though being ground-breaking, suffer from limited scalability and do not scale to large probabilistic data.

In my dissertation, we aim to design techniques to tackle the challenges arising from monitoring and summarization of large probabilistic data. Specifically, we investigate the distributed probabilistic threshold monitoring which has been demonstrated in [26] and scalable histogram summary on large probabilistic data respectively. The goal of DPTM is to monitor the aggregate score (e.g. sum) of distributed probabilistic data continuously and generate an alarm when the thresholds are violated. The challenge is that computing the exact distribution of the aggregate score (e.g. sum, mean) over distributed probabilistic data is very expensive. We leverage on the tail bounds of the distribution of the aggregated scores to help assert alarms in the monitoring instance. Meanwhile, adaptive threshold monitoring techniques are incorporated to the monitoring instances to further reduce the communication cost. When the tail bounds check fail to assert the alarms, we also introduce several novel sampling-based techniques to approximate the exact distribution. Dalvi et al also shown that answering queries w.r.t. all possible worlds is in #P-hard complexity [10]. This motivates me to explore building a compact synopsis or summary, e.g. histogram, of the distribution of the underlying probabilistic database. We investigate the problem of scaling up histogram constructions on large probabilistic data. In particular, we focus on the extension of the V-optimal histogram to probabilistic databases using the *expectation-based* semantics [7]. Our goal is to tap the distributed and parallel processing power offered by a cluster of commodity machines, which enables scalable construction of such histograms over large data.

2. MONITORING DISTRIBUTED PROBABILISTIC DATA

2.1 Problem Formulation

Given g distributed clients $\{c_1, \ldots, c_g\}$, and a centralized server H. We consider the *flat model* for the organization of distributed sites as shown in Figure 1. We assume that data from different sites are independent; each tuple has one attribute and every uncertain attribute has a pdf with bounded size for its value distribution (see Figure 1). At each time instance t, for $t = 1, \ldots, T$, client c_i reports a tuple $d_{i,t}$ with a score $X_{i,t}$ to server H.

tuples	attribute *score*
d_1	X_1
d_2	X_2
.	...
d_t	X_t

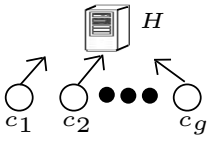


Figure 1: Attribute-level uncertain tuple and the flat model.

Definition 1 (DPTM) Given $\gamma \in \mathbb{R}^+$ and $\delta \in [0, 1)$, let $Y_t = \sum_{i=1}^{g} X_{i,t}$, for $t = 1, \ldots, T$. The goal is to raise an alarm at H, whenever for any $t \in [1, T]$ $\Pr[Y_t > \gamma] > \delta$.

In our definition, DPTM monitors the *sum* constraint. Monitoring the *average* constraint is equivalent to this case, as well as any other types of constraints that can be expressed as a linear combination of one or more *sum* constraints.

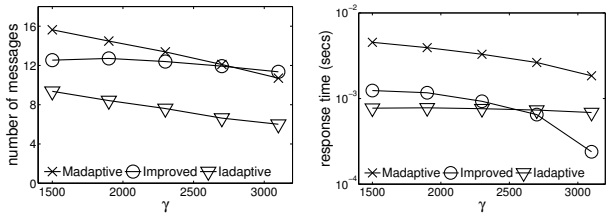
The goal is to minimize both the *overall* communication and computation costs, at the end of all time instances. We measure the communication cost using both the total number of bytes transmitted and the total number of messages sent. Lastly, when the context is clear, we omit the subscript t from Y_t and $X_{i,t}$.

2.2 Methods and Evaluation

Naive method. Naively, we can ask each client c_i to send its score X_i to H and compute $\Pr[Y > \gamma]$ exactly at H. Let n denote the max pdf size of X_i. When each X_i is represented by discrete pdf, clearly, we can compute the distribution of Y by enumerating all combinations of possible values from X_i's in $O(n^g)$ time. We can reduce this cost to $O(n^{g/2} \log n^{g/2})$ by combining the pdf of $Y_{1,\ldots,g/2}$ and the cdf of $Y_{g/2+1,\ldots,g}$. When X_i's are represented by continuous pdfs, we leverage on the characteristic functions of X_i's to compute Y exactly. Overall, the exact computing of $\Pr[Y > \gamma]$ is very expensive on both computation and communication costs.

Filtering by tail bounds. By the Markov inequality, we have $\Pr[Y > \gamma] \leq \frac{\mathbf{E}(Y)}{\gamma}$. Given that $\mathbf{E}(Y) = \sum_{i=1}^{g} \mathbf{E}(X_i)$, each client c_i only sends $\mathbf{E}(X_i)$, and H can check if $\frac{\mathbf{E}(Y)}{\gamma} < \delta$. If this is true, no alarm should be raised for sure; otherwise, we can then ask for X_i's, and apply the exact algorithm. We can improve this further. Since $\mathbf{E}(Y) = \sum_{i=1}^{g} \mathbf{E}(X_i)$ and our goal is to monitor if $\mathbf{E}(Y) < \gamma\delta$ by the Markov inequality, we can leverage on the adaptive thresholds algorithm for the DTM problem in deterministic data [23] to monitor if $\sum_{i=1}^{g} \mathbf{E}(X_i) < \gamma\delta$ continuously. We dub this method as Madaptive. We further improve the number of asserted cases by leveraging more sophisticated combinations of more accurate tail bounds, i.e. Chebyshev and Chernoff bounds. We dub it as the Improved method. However, the improved method still need to send g messages every time instance. Similar to Madaptive, we propose an improved adaptive method based on the moment generating functions and adaptive monitoring techniques, which we dub as Iadaptive method. Whenever the tail bound cannot assert an alarm at a time instance t, clients transmit X_i's to H and server applies the exact algorithm.

Estimating the threshold. In a lot of situations, the exact computation of $\Pr[Y > \gamma]$ is expensive and impractical due to both the communication and computation costs involved. Instead, we can approximate the probabilities of raising an alarm by a Monte Carlo approach where H asks each c_i for a sample x_i from X_i. He then computes a value $y = \sum_{i=1}^{g} x_i$; this is a unbiased estimation of $\Pr[Y > \gamma]$. We can repeat this to amplify success by several distribute random/deterministic sampling techniques.



(a) Messages. (b) Response time.

Figure 2: vary score threshold: γ.

Evaluations. Our methods are evaluated by real datasets from the SAMOS project [28]. Figure 2 shows the message communication cost and response time of *Madaptive*, *Improved*, and *Iadaptive* when we vary γ from 1500 to 3100 for the sum of wind direction observations from 10 sites. The number of messages reduce for all algorithms while γ increases, since probabilistic tail bounds become tighter for larger γ values. Figure 2(a) indicates that *Iadaptive* communicates the least number of messages, and it uses only half to one-third number of messages compared to *Madaptive* and *Improved* in most cases. In fact, it sends less than one message per client per time instance in most cases. Figure 2(b) shows the response time of these methods when γ varies. Clearly,

all methods take less time as γ increases, since there are fewer instances where they need to call the exact method (which is costly). *Improved* and *Iadaptive* are much more efficient than *Madaptive*. When $\gamma = 3100$, *Iadaptive* takes less than 0.001 second, and *Improved* takes close to 0.0003 second.

2.3 Contribution

In this work, we present a comprehensive study on the threshold monitoring problem over distributed probabilistic data with a focus on the sum constraint and the flat model. The main contributions are as follows:

- We extended the distributed threshold monitoring problem to work on the probabilistic data and formalize the distributed probabilistic threshold monitoring problem.
- We propose baseline methods based on the Markov tail bound, which improve over the naive method of sending all tuples at each time instance.
- We design more efficient and effective monitoring methods that leverage moment generating functions in chernoff bound and adaptive filters to significantly reduce both the communication and computation costs.
- When an exact solution is not absolutely necessary, we introduce novel sampling-based methods to further reduce the communication and the cpu costs.
- We extensively evaluate all proposed methods on large real data obtained from research vessels in the SAMOS project [28]. The results have shown that our monitoring methods significantly outperform the baseline approach. They also indicate that our sampling method is very effective when it is acceptable to occasionally miss one or two alarms with very small probability.

3. SUMMARIZATION OF LARGE PROBABILISTIC DATA

3.1 Problem Formulation

The tuple and value models are two common probabilistic models, which express the data uncertainty in terms of the tuple level and attribute level respectively. For both models, we consider the items are drawn from the integer domain $[N] = \{1, ..., N\}$ and use $\mathcal{W}$ to represent all the possible worlds for the tuple model and the value model respectively. For $\forall W \in \mathcal{W}$, let $g_i(W)$ denote the frequency of domain value i in possible world W. Given a fixed possible world W, the histogram of data distribution $G(W) = \{g_1(W), ..., g_N(W)\}$ partitions the ordered domain $[N]$ into B non-overlapping consecutive buckets (s_k, e_k), where $s_1 = 1, e_B = N$ and $s_{k+1} = e_k + 1$. In each bucket b_k, any frequency value within $\{g_{s_k}(W), \ldots, g_{e_k}(W)\}$ is represented (approximated) by a single representative $\hat{b}_k$ and we describe it as $(s_k, e_k, \hat{b}_k)$. It's easy to observe that, with the same B buckets setting, the error of histogram approximation on all possible worlds $\mathcal{W}$ is a random variable. So we focus on the optimal piece-wise constant histogram on probabilistic data that minimize the expected error over all possible worlds $\mathcal{W}$ as Cormode et al shown in [7].

To evaluate the accuracy of a given histogram, the error metric must be specified. Here, we focus on the Sum-Square-Error (SSE) error metric, which is one of the most commonly used metrics. The corresponding histogram using the SSE error metric and a limited number of buckets is known as V-Optimal histogram in [18]. Formally, the V-Optimal histogram for probabilistic data is:

Definition 2 Given the frequency sequence of random variables $\{g_1, \ldots, g_N\}$ defined by the probabilistic data models (e.g. the value model or the tuple model), the problem seeks to construct a B (far smaller than N) constant bucket representatives such that the expected summation of bucket errors over all possible worlds is minimized, i.e.,

$$\min\{E_{\mathcal{W}}\left[\sum_{k=1}^{B}\sum_{i=s_k}^{e_k}(g_i - \hat{b}_k)^2\right]\} \tag{1}$$

In equation (1), the expectation of the sum of bucket errors is equal to the sum of expectations of bucket errors considering the independence of each bucket error. Consequently, the optimal histogram could be derived using a dynamic programming (DP) formulation. We dub this optimal algorithm as OPTHIST. Cormode et al. [7] show that, given any bucket boundary pair (s, e) the minimal error of the bucket $b = (s, e, \hat{b})$ is achieved by setting the representative $\hat{b} = \frac{1}{e-s+1}E_{\mathcal{W}}\left[\sum_{i=s}^{e} g_i\right]$. The corresponding bucket error is as follows:

$$SSE(b, \hat{b}) = \sum_{i=s}^{e} E_{\mathcal{W}}[g_i^2] - \frac{1}{n_b}E_{\mathcal{W}}[(\sum_{i=s}^{e} g_i)^2] \tag{2}$$

Arrays storing the prefix sum of separate component(s) in equation (2) are used for the value model and the tuple model respectively in order to respond the minimal bucket error $SSE(s, e, \hat{b})$ query in constant time. Thus, the optimal bucket representatives could be computed in $O(BN^2)$ time using dynamic programming. Clearly, OPTHIST does not scale up to large data sets, i.e. when N is large. Our goal is to build scalable histograms over large probabilistic data using distributed and parallel algorithms.

3.2 Methods and Evaluation

Inspired by the DNS algorithm for sequence segmentation problem [32], we propose the PMERGE algorithm implemented on the MapReduce framework. The PMERGE algorithm includes two processing phases, namely the partition phase and the merge phase. In the partition phase, we partition the domain $[N]$ equally into m subdomains (intervals) $[s_1, e_1], \ldots, [s_m, e_m]$ where $s_1 = 1, e_m = N$ and $s_{k+1} = e_k + 1$ and the OPTHIST algorithm is performed on each individual subdomain in parallel. Then, in the merge phase, the representatives weighted by the associated boundary intervals over all subdomains are merged into one sequence which serves as the input of another DP and produces the final B representatives and bucket boundaries. Eventually, the PMERGE algorithm produces a 3-approximation. We illustrate the process of PMERGE algorithm in Figure 3 using the blue circles and red circles to represent two different possible worlds and $B = 2$.

We can speed up the computation of prefix arrays for each subdomain by partitioning the tuple sequence into multiple data splits on the mapper side. Thus, each element in the prefix array of a subdomain is a distributed sum of its counterparts from the associated mappers. The difference between the tuple model and the value model is the way they compute its corresponding prefix arrays for the DP in the partition phase of PMERGE. We can recursively apply this partition & merge idea if Bm is still large for the DP in terms of computation cost. We dub the algorithm using l depth of partition & merge processes as l-PMERGE, which gives a $6^{(\frac{1}{2})}$ approximation. For short, we denote 1-PMERGE as PMERGE.

To reduce the communication costs of PMERGE, we can approximate the prefix arrays using sampling based methods. We attempt the following two methods to approximate the prefix arrays in the

value model: 1) We can view each element $A[i]$ of the prefix array A as a distributed sum, which could be approximated by distributed frequency estimation method in [35]. 2) Alternatively, we can view each element $A[i]$ of the prefix array as the rank of item i, i.e. the number of values that is less and equal than item i in the prefix array A. Thus, each $A[i]$ can be approximated using the distributed value-to-rank quantile estimation as shown in [34]. The most challenge part is some component in the prefix arrays of the tuple model, which is a F2 moment. We design a novel method by combining the dyadic intervals and GCS sketch techniques to approximate it. We dub the sampling based method as l-PMERGES.

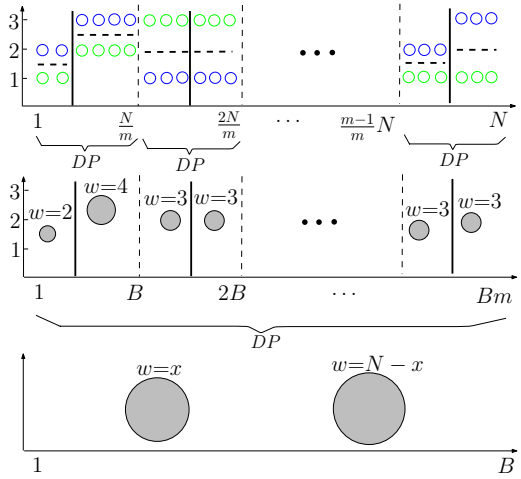
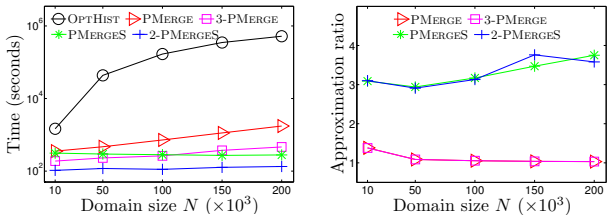


Figure 3: Partition and Merge on PMERGE.

Evaluation. Our methods are evaluated by real datasets from the WorldCup data [2]. Figure 4 shows the runtime and approximation ratio of all methods when we vary N on the value model. The runtime improvement of our approximate methods is more than orders of magnitude over OPTHIST and the trend becomes more obvious as N increases. Specifically, at $N = 200000$, 3-PMERGE took less than 500 seconds while the OPTHIST took around 6 days. Meanwhile, our l-PMERGE methods also achieve a satisfactory approximation quality. PMERGE and 3-PMERGE show a very good approximation ratio, close to 1, across all N values. The approximation quality of the sampling based PMERGES and 2-PMERGES degrades slowly as N increases.



(a) Runtime. (b) Approximation ratio

Figure 4: Value model: vary N.

3.3 Contribution

In this work, we presents a comprehensive study on the scalable histogram for large probabilistic data sets in MapReduce. We focus on the V-optimal histogram based on the expectation-based semantic for the value model and the tuple model respectively. The main contributions are as follows:

- We design a scalable histogram construction method for both the tuple and the value models on the MapReduce. Our method significantly reduces the computation cost and provide an constant approximation. Also, the method could be easily adopted by any distributed and parallel computation framework.
- When the approximate method is expensive in communication cost, we introduce sampling-based techniques to further reduce the communication cost and the approximation quality doesn't degrade much.
- We conduct extensive experiments on large real probabilistic data sets in a Hadoop cluster with 17 nodes. The results suggest that our approximate algorithms achieved significant (orders of magnitude) runtime improvement and communication costs savings as well as a satisfactory approximation quality.

4. RELATED WORK

4.1 Distributed Probabilistic Threshold Monitoring

To our knowledge, aggregate constraint monitoring on distributed data with uncertainty has not been explored before.

Monitoring centralized uncertain data for top-k and similarity queries were studied in [17,33]. On the other hand, due to their importance and numerous applications, constraint and function monitoring with thresholds on deterministic distributed data were examined extensively, e.g., [8,23] In our study, we have leveraged on the adaptive thresholds algorithm for the deterministic (sum) constraint monitoring from [23]. This choice is independent from the design of our adaptive algorithms for the DPTM problem: any adaptive algorithms for the (sum) constraint monitoring in deterministic data can be used in our *Iadaptive* and *Madaptive* methods.

Our study is also related to aggregation queries in probabilistic data, e.g., [20,21]. However, monitoring both score and probability thresholds on aggregate constraints continuously over distributed probabilistic data is clearly different from these studies. Probabilistic threshold queries in uncertain data are also relevant [5,9,27], as they are also concerned with the probability thresholds on the query results. However, they mostly focus on one-shot query processing over centralized, offline probabilistic data.

4.2 Histogram Construction

To our knowledge, building histograms on large probabilistic data in MapReduce has not been explored before. Building optimal histograms on probabilistic data has been recently studied in [7], [6] from expectation-based and probabilistic-based semantics respectively. However, as we have pointed out in Section 1, the expensive construction time rendering it unsuitable for large probabilistic datasets. In particular, we focus on scalable histogram for large probabilistic datasets using the expectation-based semantic [7].

Our study is related to work on the sequence segmentation problem [32]. The work in [32] has focused on addressing the segmentation on deterministic sequence data and it targets for the computation on a single machine, which is different from our data models and computation setting. Our study is also related to work on building wavelet histograms on large data in MapReduce [22]. However, [22] focused on how to summarize massive deterministic data using wavelet histograms in MapReduce.

The optimal and approximate algorithms of histogram construction on deterministic data have been extensively studied, e.g., [13, 15, 19, 31]. In our study, we have leveraged on the optimal histogram algorithm on probabilistic data in [7] in the partition phase

and the optimal histogram algorithm on deterministic data in [19] in the merge phase. The choice is independent from the design of the l-PMERGE algorithms: any corresponding optimal algorithms can be used in our l-PMERGE method.

5. CONCLUSION

My dissertation aims to design techniques to tackle challenges arising from monitoring and summarization of large probabilistic data. Specifically, we study two instance problems: 1) distributed probabilistic threshold monitoring, 2) scalable histograms on large probabilistic data. Our extensive experiments on large real data sets have shown the superb performance of our techniques.

For the future work, I'm interested in studying the extensions of our monitoring model from flat model to hierarchical model that is often used in a sensor network; handle the case when data from different sites are correlated; combine the histogram summary techniques to reduce the size of distribution and computation cost in DPTM problem; and handle the scalable histogram using other error metric, such as Sum-Squared-Relative-Error, that doesn't satisfy the approximation guarantee.

6. ACKNOWLEDGMENTS

This work was partially supported by NSF grants IIS-0916488 and IIS-1053979. Thanks to my thesis advisor Feifei Li for numerous helpful discussions.

7. REFERENCES

[1] P. Agrawal, O. Benjelloun, A. D. Sarma, C. Hayworth, S. Nabar, T. Sugihara, and J. Widom. Trio: A system for data, uncertainty, and lineage. In *VLDB*, 2006.

[2] M. Arlitt and T. Jin. A workload characterization study of the 1998 world cup web site. In *Technical report, IEEE Network*, 1999.

[3] J. Boulos, N. Dalvi, B. Mandhani, S. Mathur, C. Re, and D. Suciu. Mystiq: A system for finding more answers by using probabilities. In *SIGMOD*, 2005.

[4] R. Cheng, D. Kalashnikov, and S. Prabhakar. Evaluating probabilistic queries over imprecise data. In *SIGMOD*, 2003.

[5] R. Cheng, Y. Xia, S. Prabhakar, R. Shah, and J. S. Vitter. Efficient indexing methods for probabilistic threshold queries over uncertain data. In *VLDB*, 2004.

[6] G. Cormode and A. Deligiannakis. Probabilistic histograms for probabilistic data. In *VLDB*, 2009.

[7] G. Cormode and M. Garofalakis. Histograms and wavelets on probabilistic data. In *ICDE*, 2009.

[8] G. Cormode, S. Muthukrishnan, and K. Yi. Algorithms for distributed functional monitoring. In *SODA*, 2008.

[9] N. Dalvi and D. Suciu. Efficient query evaluation on probabilistic databases. In *VLDB*, 2004.

[10] N. Dalvi and D. Suciu. Efficient query evaluation on probabilistic databases. In *VLDB*, 2004.

[11] A. Deshpande, C. Guestrin, S. Madden, J. Hellerstein, and W. Hong. Model-driven data acquisition in sensor networks. In *VLDB*, 2004.

[12] X. Dong, A. Y. Halevy, and C. Yu. Data integration with uncertainty. In *VLDB*, 2007.

[13] P. K. Felix Halim and R. H. C. Yap. Fast and effective histogram construction. In *CIKM*, 2009.

[14] R. Fink, A. Hogue, D. Olteanu, and S. Rath. SPROUT^2: a squared query engine for uncertain web data. In *SIGMOD Conference*, pages 1299–1302, 2011.

[15] A. C. Gilbert, S. Guha, P. Indyk, Y. Kotidis, S. Muthukrishnan, and M. J. Strauss. Fast, small-space algorithms for approximate histogram maintenance. In *STOC*, 2002.

[16] L. Gruenwald, H. Chok, and M. Aboukhamis. Using data mining to estimate missing sensor data. In *ICDMW*, 2007.

[17] M. Hua and J. Pei. Continuously monitoring top-k uncertain data streams: a probabilistic threshold method. *DPD*, 26(1):29–65, 2009.

[18] Y. E. Ioannidis and V. Poosala. Balancing histogram optimality and practicality for query result size estimation. In *SIGMOD*, 1995.

[19] H. V. Jagadish, N. Koudas, S. Muthukrishnan, V. Poosala, K. Sevcik, and T. Suel. Optimal histograms with quality guarantees. In *VLDB*, 1998.

[20] T. S. Jayram, S. Kale, and E. Vee. Efficient aggregation algorithms for probabilistic data. In *SODA*, 2007.

[21] T. S. Jayram, A. McGregor, S. Muthukrishnan, and E. Vee. Estimating statistical aggregates on probabilistic data streams. In *PODS*, 2007.

[22] K. Y. Jeffrey Jestes and F. Li. Building wavelet histograms on large data in mapreduce. In *VLDB*, 2011.

[23] S. Jeyashanker, S. Kashyap, R. Rastogi, and P. Shukla. Efficient constraint monitoring using adaptive thresholds. In *ICDE*, 2008.

[24] C. K. Lyublena Antova and D. Olteanu. 10^{10^6} worlds and beyond: Efficient representation and processing of incomplete information. In *ICDE*, 2007.

[25] Y. Matias, J. S. Vitter, and M. Wang. Wavelet-based histograms for selectivity estimation. In *SIGMOD Conference*, pages 448–459, 1998.

[26] J. M. P. Mingwang Tang, Feifei Li and J. Jestes. Efficient threshold monitoring for distributed probabilistic data. In *ICDE*, 2012.

[27] L. Perez, S. Arumugam, and C. Jermaine. Evaluation of probabilistic threshold queries in MCDB. In *SIGMOD*, 2010.

[28] SAMOS. Shipboard Automated Meteorological and Oceanographic System. http://samos.coaps.fsu.edu.

[29] S. Singh, C. Mayfield, S. Mittal, S. Prabhakar, S. E. Hambrusch, and R. Shah. Orion 2.0: native support for uncertain data. In *SIGMOD Conference*, pages 1239–1242, 2008.

[30] D. Suciu, D. Olteanu, C. Ré, and C. Koch. *Probabilistic Databases*. Synthesis Lectures on Data Management. Morgan & Claypool Publishers, 2011.

[31] N. K. Sudipto Guha and K. Shim. Approximation and streaming algorithms for histogram construction problems. *TODS*, 31(1):396–438, March 2006.

[32] E. Terzi and P. Tsaparas. Efficient algorithms for sequence segmentation. In *SIAM SDM*, 2006.

[33] H. Woo and A. K. Mok. Real-time monitoring of uncertain data streams using probabilistic similarity. In *RTSS*, 2007.

[34] K. Y. Zengfeng Huang, Lu Wang and Y. Liu. Sampling based algorithms for quantile computation in sensor networks. In *SIGMOD*, 2011.

[35] Y. L. Zengfeng Huang, Ke Yi and G. Chen. Optimal sampling algorithms for frequency estimation in distributed data. In *IEEE INFOCOM*, 2011.