Improving the accuracy of continuous aggregates and mining queries on data streams under load shedding

Yan-Nei Law

Bioinformatics Institute,
30 Biopolis Street, #07-01,
Matrix, 138671, Singapore
Fax: +65 6478 9048 E-mail: lawyn@bii.a-star.edu.sg

Carlo Zaniolo*

UCLA Computer Science Department,
4732 Boelter Hall,
Los Angeles, CA 90095, USA
Fax: +1 310 825 2273
E-mail: zaniolo@cs.ucla.edu
*Corresponding author

Abstract: Random samples are common in data streams applications due to limitations in data sources and transmission lines, or to load-shedding policies. Here we introduce a formal error model and show that, besides providing accurate estimates, it improves query answer accuracy by exploiting past statistics. The method is general, robust in the presence of concept drift, and minimises uncertainties due to sampling with negligible time and space overhead. We describe the application of the method, and the results obtained for SQL window aggregates, statistical aggregates such as quantiles, and data mining functions such as k-means clustering and naive Bayesian classifiers.

Keywords: load shedding; data stream; query processing; sampling.

Reference to this paper should be made as follows: Law, Y.N. and Zaniolo, C. (2008) 'Improving the accuracy of continuous aggregates and mining queries on data streams under load shedding', *Int. J. Business Intelligence and Data Mining*, Vol. x, No. x, pp.xxx–xxx.

Biographical notes: Yan-Nei Law received her PhD in Computer Science at the University of California, Los Angeles. She is a post doctoral research fellow in Bioinformatics Institute, Singapore. Her research interests include issues related to bioimaging with emphasis on developing image processing and data mining techniques for high throughput images.

Carlo Zaniolo is a Professor of Computer Science with the University of California at Los Angeles, where he occupies the N.E. Friedmann Chair in Knowledge Science. His current research interests include mining data bases and data streams, data stream management systems, data bases and decision support systems, archival information systems, temporal databases, schema evolution, data bases and knowledge bases, nonmonotonic reasoning.

1 Introduction

Many data stream applications require complex queries involving aggregates, analysis, and mining functions (Babcock et al., 2002). Examples include financial analysis (Chen et al., 2000), network monitoring (Sullivan, 1996), sensor networks (Chandrasekaran et al., 2003), web logs and click-streams, telecommunication data management (Cortes et al., 2000) and many others. These applications are often characterised by high data rates, and a demand for real-time (or near real-time) responses. Traditional database management systems were designed to manage persistent databases and transient queries, and they are not suitable for stream applications that instead require support for persistent queries and transient data. Data Stream Management Systems (DSMS) are therefore being developed to satisfy the unique requirements and technical challenges posed by such applications. Foremost among these, we find the problems created by the bursty nature and unpredictably high arrival rates of data streams, which often make it impossible to provide exact real-time answers to all inputs and continuous queries. To deal with such overload situations, the DSMS must incorporate intelligent *load shedding* policies, to minimise the loss of quality of the query answers in the presence of such overloads. This problem has motivated interesting research work seeking techniques to minimise the loss of information caused by load shedding (Kang et al., 2003; Das et al., 2003; Tatbul et al., 2003; Babcock et al., 2004).

Effective solutions are often made possible by the fact that many data stream applications and DSMS tasks, such as aggregate queries and stream mining methods, only require approximate answers – provided that these satisfy certain statistical accuracy requirements. Therefore, the interesting problem arises of designing optimal policies to optimise answer accuracy when a certain amount of load shedding is needed on the input. The problem of designing load-shedding policies for aggregate queries was treated in Babcock et al. (2004), where an optimum policy based on random sampling was proposed. The interesting approach proposed in Babcock et al. (2004) is based on statistical properties of the incoming data streams. In this paper, we show that, by the same statistical properties used in Babcock et al. (2004), it is also possible to improve the very quality of the answers, since more accurate estimates can now be derived from random samples. We first present the theory that makes these improvements possible and then introduce a method that realises them and yields significant benefits in many queries of practical interest. We first consider the traditional aggregates, such as SUM, COUNT and AVG, studied in Babcock et al. (2004), and then we extend our approach to more complex aggregate-like queries, such as quantiles, and mining queries.

It should be noted that the benefits of the proposed theory and method are even more important on statistical functions than on simple aggregates, where a ‘naive’ uniform sampling of the content of the current sliding window tends to be reasonably effective. However, for complex mining and analytic functions, the errors in the approximate answers can be very large, unless sophisticated estimation techniques are used.

In this paper therefore, we will propose a technique to improve the accuracy of these queries given a sample of the window. Our method is not limited to the load shedding scenarios, but is also applicable to all situations where a random sample of data is available. For instance, random sampling might be caused by faulty data sources or transmission lines.

Related work. Most recent work has focused on computing approximate query answers. A first line of research concentrates on computing sliding window joins (Das et al., 2003; Srivastava and Widom, 2004) and aggregates (Dobra et al., 2002). Another interesting study presented in Arasu and Manku (2004) focused on summarising the content of the continuous sliding windows for computing approximate sliding window quantiles, but they did not consider the scenarios where load shedding is used to cope with system overloads. In Jain et al. (2004), the stream resource management problem was considered as a filtering problem, in which the objective is to filter out as much as data as possible, provided a required precision standard.

In Babcock et al. (2004), a systematic approach was proposed to load-shedding, with the objective of maximising query accuracy. The referenced paper showed that uniform random samples of the window contents should be used since this policy produces the best results on aggregation queries over data streams. However, correlation between answers of a query at different points in time was not considered in Babcock et al. (2004), although it can be very useful, as we will show in this paper.

Our approach. Unlike traditional databases, which deal with relatively static records, DSMS monitor data continuously over time. In typical data stream applications, such as sensor data and internet traffic, there is a strong temporal correlation between answers of a query at different points in time (Vuran et al., 2004), and we can obtain statistical information about the current answer from the past answers. Therefore, this information can be incorporated into the approximation process, and used in data quality evaluation techniques to improve the accuracy of the results with only minimal cost in system resources.

Example 1: Consider the problem of continuously finding the daily average temperature in Los Angeles. Everyday we get the temperature reading for every hour from www.weather.com and return the average of that day at midnight. In particular, the true average of March 9, 2006 is 12.88 (all temperatures are in centigrade). Suppose that, due to some transmission problems, we only received the following four readings for the day: {12.0, 13.0, 16.0, 12.0}. Based on these sample readings, we could use their average, i.e., 13.25, as our estimate of the actual average $\hat{A}$. But for a better estimate, we can take into account the history of the previous answers on the daily temperatures for last week, which were: {11.83, 11.13, 12.13, 12.67, 13.38, 13.79}. In particular, we may simply return a linear combination of the estimated result and the mean of last week daily averages ($\mu = 12.49$), i.e., $0.5\hat{A} + 0.5\mu$, as our final answer. Then our improved answer will be 12.87, which is much closer to the true average than $\hat{A}$.

Data quality evaluation approaches exploiting spatio-temporal correlations that exist among sensors are frequently used in sensor networks to reduce communication overheads and effects of data loss and data contamination caused by non-performing sensors (Jeffery et al., 2006). For instance, some statistical information of a sensor could be learned from the readings of its neighbours. Another source is the past readings of the sensor itself. The idea is to reduce the effect of noise on sensor readings (data) through their error model obtained by sensor testings. However, previous work on processing data streams has not used similar techniques to improve the approximate answers.

In this paper, we attempt to use the historical answers to improve the accuracy of the output which is degraded by sampling or load shedding. In Example 1, we arbitrarily

assigned weighting factors for the historical information and the raw answer. However, our objective is to adjust the current answer by the history in a more advanced way: When the sampling rate is high, the observed answer will be more accurate. Hence, the improved answer should be less dependent on the history. On the other hand, when the sampling rate is low, the observed answer will be less accurate. Hence, the improved answer should be more dependent on the history. To achieve our objective, it is necessary to find the relationship between the sampling rate and the error of the answer. Therefore, we first develop a technique to obtain an error model of different queries given a sampling rate to analyse the accuracy of the answer. Previous work on processing data streams has not used similar techniques. This paper represents the first attempt to apply these ideas to improve the accuracy of answers under load shedding. The main contributions of this paper are:

- We provide an in-depth study of the error models of different kinds of queries including sum, quantile and mining data streams.
- We propose a Bayesian approach which can automatically adjust the degree of involvement of the historical information. This can improve the accuracy of aggregate query answers by reducing their uncertainty.
- We apply and test our method on a range of different applications, including queries to derive window medians, quantiles and stream mining.

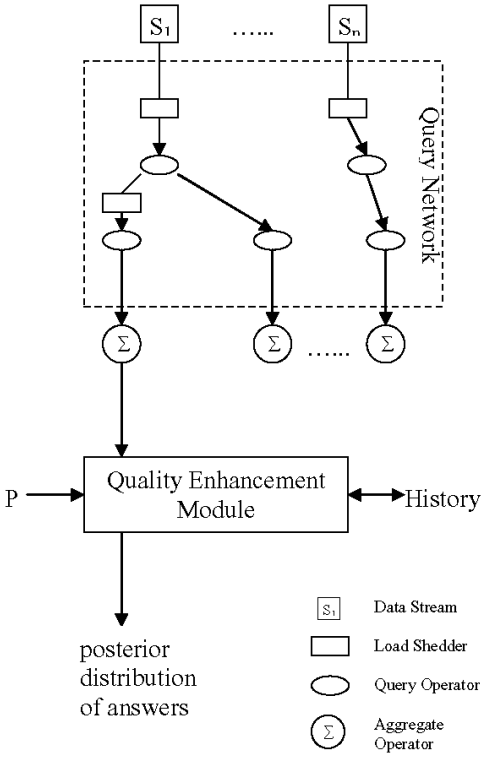
Roadmap. The rest of our paper is organised as follows: We first formally present our model and discuss the problem in Section 2. In Section 3, we propose a quality enhancement technique to combine the information of the answer to improve its accuracy. Section 4 presents some extensions for further applications such as computing ordered statistics and data mining queries. We discuss the minimal overhead of our technique in Section 5. Section 6 shows the experimental results of our technique. Finally, Section 7 presents our conclusions and suggestions for future work.

2 Problem description

Consider two common scenarios in which random samples of data are available for query answering. The first scenario is that of input streams from applications such as sensor networks. Due to the problems of limited bandwidth for transmission or hardware malfunction, it is often the case that only a subset of the sensor data is received. Then, the central station can only compute queries based on the received random samples of the source input. The second scenario is the load shedding situation studied in Babcock et al. (2004). In this scenario, the system copes with overload situations by sampling each input stream and answering the queries from the samples so collected.

The query network model proposed to deal with these two situations is shown in Figure 1. A set of continuous queries $q_1, \ldots, q_m$, over a given set of input streams $S_1, \ldots, S_n$, is implemented by a network of query operators $O_1, \ldots, O_k$. Since we only consider unary operators, our network consists of set of directed trees, whose node represent operators and the arcs represent the flow of the streams between the operators. All leaf nodes represent aggregation operators, while the other nodes contain arbitrary unary operators.

Figure 1 General system model with query network and quality-enhancement module



Following Babcock et al. (2004) we now assume that a load shedder is inserted on each arc except those leading to the leaf aggregation operators. When overload occurs, we can use any load shedding technique that produces uniform random samples of its input data. In particular, we can use the practical load shedding method proposed in Babcock et al. (2004), which provides a systematic approach to load shedding for minimising inaccuracy in query answers, subject to the system constraints. The main idea of said method is to assign a sampling rate to each load shedder in order to minimise the maximum relative error across all queries. Then, the load shedders in the query network are used to produce a uniform random sample for each query q_i with sampling rate P_i.

In this paper however, we want to go beyond the approach proposed in Babcock et al. (2004), and improve the quality of the answer by exploiting the correlation between answers of a given query at different points in time. Therefore, as shown in Figure 1, we add a *quality-enhancement module* to the basic query network described in Babcock et al. (2004).

Therefore, our model consists of two parts. The first part is a query network that can be viewed as a black box which

- delivers a uniform random sample for computing aggregation queries
- reports the sampling rate.

The query network shown in Figure 1 is for load shedding. Indeed, our method works for every situation where uniform random samples are available for answering queries.

The second part of our model is a quality-enhancement module. It is used to reduce the uncertainty of the approximate answers, using:

i the prior knowledge of the answer

ii the error model of the answer.

The quality-enhancement module combines (i) and (ii) with the observed answer and returns the posterior distribution for the true answer. We will discuss each component in the coming sections.

3 Improving answer quality

The main idea of our method is to use a Bayesian approach to combine the information about the answers in order to reduce the uncertainty on the answers caused by random sampling. As a result, the method is beneficial in coping with random losses of input data due to intermittent faults, besides the load shedding situation discussed in Babcock et al. (2004). We now discuss the theory underlying the quality-enhancement technique. This theory is based on:

- an in-depth study of the error models of the answers of queries including aggregates, ordered statistics and data mining problem
- a model for learning the prior knowledge on the current answer from the past answers
- a technique for combining the above two parts together with the observed answers in order to obtain a more accurate estimate of the answer.

3.1 Obtaining answers under random sampling

In this section, we discuss a method to find the precise error distribution of the approximate answer, which can be used to

- derive accuracy estimate for optimising the load shedding policies
- to further improve the answers, on the basis of statistical information of the answers.

SUM: Let us first consider a query network consisting of a continuous window aggregation SUM query q. Let V be the set of input tuples of Σ operator in the sliding window. Thus, they will contribute to the answer for q. When load shedders are present with sampling rate p, every tuple in V will get included independently in the sample with probability p. For the tuples in V, let $v_1, \ldots, v_N$ be the values of the attribute being summed up, and let A be their sum. Based on this random sample, we can set an approximate answer $\hat{A}$ to be the sum of v_i's for the tuples that get included, scaled by $1/p$. The fact stated in Babcock et al. (2004) says that this scenario is equivalent to the following setting: Let $X_1, X_2, \ldots, X_N$ be N random variables, such that each random variable X_j takes the value v_j/p with probability p and the value zero otherwise. These random variables are used to identify if the tuples get included in the final answer. Let $\hat{A}$ be the sum of these random variables and let $A = \sum_{j=1}^{N} v_j$ be the true answer.

Then we can use $\hat{A}$ to be an approximation of A. By using the load shedding algorithm in Babcock et al. (2004), we will get a random sample of V with optimal sampling rate P.

So far, we have a query network model for producing a random sample of V with optimal sampling rate P. However, we now take our study beyond this by developing an error model of the approximate answer, in order to further improve its accuracy of the answer. We first observe the fact that $\hat{A}$ is an unbiased estimator of A, i.e.,

$$\begin{aligned} E(\hat{A}) &= E\left(\sum X_i\right) = \sum E(X_i) \\ &= \sum P(v_i / P) + (1-P) \times 0 = \sum v_i = A. \end{aligned}$$

with variance:

$$\begin{aligned} Var(\hat{A}) &= E(\hat{A}^2) - E(\hat{A})^2 \\ &= \sum E(X_i^2) + \sum_{i \neq j} E(X_i)E(X_j) - A^2 \\ &= \sum P(v_i / P)^2 + \sum_{i \neq j} P(v_i / P) \times P(v_j) / P - A^2 \\ &= \frac{1}{P}\sum v_i^2 + \left(\sum v_i\right)^2 - \left(\sum v_i^2\right) - \left(\sum v_i\right)^2 = \frac{1-P}{P}\sum v_i^2. \end{aligned}$$

Note that the ratio $\sum v_i^2 / \left(\sum v_i\right)^2$ is equal to $(\sigma^2 + \mu^2)/(N\mu^2)$, where $\mu = \sum_{j=1}^{N}(v_j / N)$ and $\sigma^2 = \sum_{j=1}^{N}(v_j - \mu)^2 / N$ are the mean and the variance of all the tuples in V respectively. Thus, the right-hand side of the above expression can be reduce to

$$Var(\hat{A}) = \frac{1-P}{P} \times \frac{\sigma^2 + \mu^2}{N\mu^2} \times A^2. \tag{1}$$

By the Central Limit Theorem, for an infinite population, if the sample size is large, then the mean of the sample follows a normal distribution. Therefore, it is reasonable to assume that the approximate answer $\hat{A}$ is normally distributed with mean A and standard deviation σ_e, i.e., $\hat{A} \sim N(A, \sigma_e^2)$. where

$$\sigma_e = \sqrt{\frac{1-P}{P} \times \frac{\sigma^2 + \mu^2}{N\mu^2}} \times A.$$

Hence, the error of the approximate answer is normal distributed with mean zero and standard deviation σ_e, i.e., $p(\hat{A} \mid A) \sim N(A, \sigma_e^2)$. We will use this error distribution to improve the answer using the technique discussed in Sections 3.2 and 3.3. Note that σ_e tends to zero when P tends to 1, which is consistent to the fact that the uncertainty is reduced when the sampling rate is increased.

COUNT: For computing the windowed counts, we simply set all the v_i to be equal to 1. Moreover, it is easy to see that $\mu = 1$ and $\sigma^2 = 0$.

AVG: The windowed averages can be computed as follows: we assign to each random variable X_j the value $v_j = (NP)$, instead of v_j/P. Then $\hat{A}$ will be an unbiased estimator of A and its variance will be

$$\frac{1-P}{P} \times \frac{\sigma^2 + \mu^2}{N\mu^2}.$$

Use of error model: First we show that how to use our error model to derive an error bound for optimising the load shedding policies. Assume that the probability that the absolute error exceeds $\varepsilon(\delta)$ is at most δ, i.e.,

$$\Pr\{|\hat{A} - A| \geq \varepsilon(\delta) \leq \delta\}.$$

With the normal error model, for each confident level δ, we can obtain the associated critical value $\varepsilon(\delta)$ from the error function of a normal distribution. For instance, 95% (i.e., $\delta = 0.05$) confidence limits for the answer of SUM are $A - 1.96\sigma_e$ and $A + 1.96\sigma_e$ (i.e., $\varepsilon(\delta) = 1.96\sigma_e$). Therefore, the error bound of the relative error of the answer derived by our error model will be

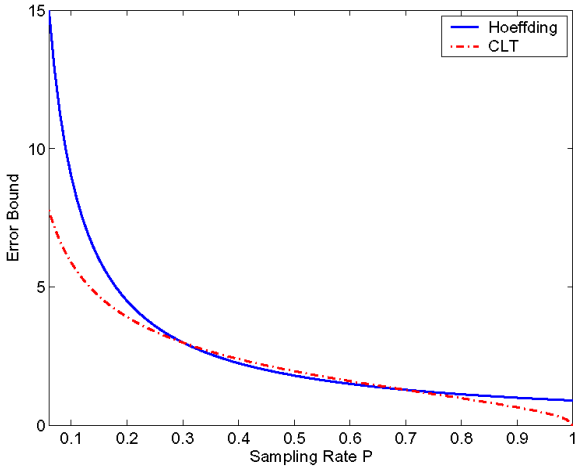
$$\varepsilon_{CLT} = 1.96 \times \sqrt{\frac{1-P}{P}} \times \sqrt{\frac{\sigma^2 + \mu^2}{N\mu^2}}.$$

On the other hand, the error bound with 95% confidence level provided by Babcock et al. (2004) will be

$$\varepsilon_{Hoeffding} = \sqrt{\frac{1}{2}\log\frac{2}{0.05}} \times \frac{1}{P} \times \sqrt{\frac{\sigma^2 + \mu^2}{N\mu^2}}.$$

In Figure 2, the plot depicts the factor $1.96 \times \sqrt{\frac{1-P}{P}}$ and $\sqrt{\frac{1}{2}\log\frac{2}{0.05}} \times \frac{1}{P}$ in the error bounds $\varepsilon_{CLT}(CLT)$ and $\varepsilon_{Hoeffding}(Hoeffding)$ respectively under different sampling rates P.

Figure 2 Error bounds $\left(\times\sqrt{\frac{\sigma^2 + \mu^2}{N\mu^2}}\right)$ vs. sampling rates



From Figure 2, we observe that the bound *CLT* is tighter than *Hoeffding* almost everywhere. This indicates that *CLT* can provide more accurate information for optimising the load shedding policies. In particular, when P approaches to 1, the error bound *CLT* approaches to 0. This is consistent to the fact that we suppose to have the exact answer when all the input are available for answering the query. However, the bound *Hoeffding* does not have this property. On the other hand, when P approaches 0, both error bounds approach to infinity. However, the bound *CLT* (i.e., $O(P^{-1/2})$) has a much slower rate than the bound *Hoeffding* (i.e., $O(P^{-1})$).

In the above discussion, we analysed the accuracy of the current answer. Our aim is not only to give an upper error bound of the query answer as it was done in Babcock et al. (2004), but also to provide the precise error model of the answers, which can be used to further improve its accuracy as follows.

3.2 Learning prior distribution from the past

Next we show how to use the temporal correlation between each tuple in a data stream to learn the prior distribution on the current answer from past answers. Although the past answers are obtained from a random sample of the tuples, we found that they are also some useful information for learning the prior distribution on the current one.

Note that the selection of the past answers for learning prior will affect the quality of the final result. Indeed, we can obtain more accurate information about the answer if we select a set of past answers which are from the same distribution as the current one. In order to select a set of past answers which contains up-to-date information, we can use any of the change detection techniques proposed in Kifer et al. (2004), Yang et al. (2005), Larson (1982) and Fan (2004). In our experiments, we used the two-sample *t*-test (Larson, 1982) which is a simple method widely used in statistics, and proved quite effective in our tests. This method is based on testing whether two sets of samples are generated from the same distribution while its complexity is relatively small. Consider two sets of samples $\mathbf{X} = \{X_1, \ldots, X_m\}$ and $\mathbf{Y} = \{Y_1, \ldots, Y_n\}$. We calculate

$$\begin{cases} s_p^2 = \dfrac{\sum (X_i - \bar{X})^2 + \sum (Y_j - \bar{Y})^2}{m + n - 2} \\ T = (\bar{X} - \bar{Y})^2 \sqrt{\dfrac{mn}{m+n}} / s_p \end{cases} \tag{2}$$

where $\bar{X}$ is the mean of $\{X_1, \ldots, X_n\}$ and $\bar{Y}$ is the mean of $\{Y_1, \ldots, Y_m\}$. Then we check whether the observed T statistics follows T distribution in order to decide whether there is a concept change. For simplicity, here we consider changes in the mean values, which proved effective in our experiments. In general, however, other criteria and techniques are at hand for detecting concept drifts. For instance, the F-test can be used to measure whether two samples have the same variance, and many other detection techniques have been proposed in the literature Kifer et al. (2004), Yang et al. (2005), Larson (1982) and Fan (2004).

To learn the prior knowledge, we use only the past answers obtained after the last detected change. After a new change is detected, we simply reset the set of past answers. While any change-detection method can be used in conjunction with our quality enhancement method to continuously check whether there are changes in the stream of

answers, we used the two-sample *t*-test because of its simplicity and efficiency. The details are as follows: for each time interval (e.g., time-based window size *p* seconds), we employ a *t*-test to determine whether the set of answers in the latest time slot A_{latest} and the set of past answers A_{past} follow the same distribution. If so, we combine the past answers with the answers in the latest time slot, i.e., $A_{past} = A_{latest} \cup A_{past}$. If not, we drop A_{past} and set the answer in the latest time slot to be A_{past}, i.e., $A_{past} = A_{latest}$. Note that this test only need to store the means and the variances of A_{past} and A_{latest}. Comparing with the method without change detection, the method including change detection only needs two more variables to store the extra statistical information to perform a two-sample test. Indeed, our experiments show that this method is accurate enough while being computationally very inexpensive.

Now we show how to learn the prior knowledge from the selected past answers. Note that the distribution of the prior can be any kind of distribution. However, in the following discussion, we focus on using a normal distribution to estimate the prior statistics. By Maximum Likelihood Estimate (MLE), the prior pdf would be $N(\mu_s, \sigma_s^2)$, where $\mu_s = \sum_{i=1}^{n} x_i / n$ and $\sigma_s^2 = \sum_{i=1}^{n} (x_i - \mu_s)^2 / n$ are the sample mean and the sample variance of the past answers respectively. Note that our Bayesian approach for improving the current answer works for different kinds of prior distributions. However, normal is a natural choice which is suitable for a wide range of applications. Moreover, it has many attractive properties which makes it very suitable for our problem, since

- it only needs two variables to keep track of the mean and the variance of the past answers
- it simplifies the answer-improvement computations discussed next.

3.3 Improving the answers

To improve the accuracy of the current answer, we can use the Bayesian approach to combine

- the answer $\hat{A}$ returned by the query network
- the error model $p(\hat{A} \mid A) \sim N(A, \sigma_e^2)$
- the prior knowledge of the true answer *A*.

In fact, the *posterior* distribution of the true answer is

$$p(A \mid \hat{A}) = \frac{p(\hat{A} \mid A)\, p(A)}{p(\hat{A})}.$$

We then take the *Maximum A Posterior (MAP)*, i.e., $\arg\max_A p(A \mid \hat{A})$, to be our improved answer.

In general, we do not restrict the prior distribution of the true answer to a specific class of distribution. Our Bayesian approach works for any kind of prior distribution when the prior captures the up-to-date information about the current answer. To ensure this, we apply the two-sample *t*-test to detect concept changes as discussed in Section 3.2. However, as mentioned in the last section, normal has many attractive properties which

make it a good choice for modelling the prior, especially when the resources (in term of time and space) are limited. In particular, when we use the normal distribution $N(\mu_s, \sigma_s^2)$ to estimate the prior pdf, we can then use the result obtained in Box and Tiao (1973). Indeed, as proven in Box and Tiao (1973), the combined posterior probability $p(A \mid \hat{A})$ will follow a normal distribution $N(\mu_t, \sigma_t^2)$ with

$$\begin{cases} \mu_t = \dfrac{\sigma_e^2}{\sigma_s^2 + \sigma_e^2}\mu_s + \dfrac{\sigma_s^2}{\sigma_s^2 + \sigma_e^2}\hat{A} \\ \sigma_t^2 = \dfrac{\sigma_s^2}{\sigma_s^2 + \sigma_e^2}\sigma_e^2. \end{cases} \tag{3}$$

Since the MAP of a normal distribution is its mean, we can now obtain the MAP (= μ_t) immediately from the closed form in equation (3).

We can make two interesting observations from equation (3). Firstly, since $\sigma_s^2/(\sigma_s^2 + \sigma_e^2) \le 1$, the variance of the posterior σ_t^2 is always smaller than or equal to the variance of the raw answer σ_e^2. Thus the uncertainty of the answer is further reduced. Secondly, the mean of the posterior μ_t is a linear combination of the mean of prior μ_s and the observed answer $\hat{A}$ with a scalar factor $\sigma_e^2/(\sigma_s^2 + \sigma_e^2)$ and $\sigma_s^2/(\sigma_s^2 + \sigma_e^2)$ respectively. When the variance of the error σ_e^2 is larger than the variance of the prior σ_s^2, then μ_s is dominated, and we will trust the prior knowledge rather than the observed value. Otherwise, the observed value $\hat{A}$ will be dominated as its scalar factor is larger. This achieves our objective stated in Section 1. Moreover, the validity of this observation is confirmed in our experiments in Section 6.

In summary, we discussed a method of using normal for modelling the prior statistics: This method is very attractive for data streams because of the minimal amount of time and space required. Moreover, our experiments demonstrate this method can provide a significant improvement of the accuracy of the answers, even when the prior follows different kinds of distributions.

4 Complex queries

Previous research has only discussed the problem of computing basic SQL aggregates such as SUM and COUNT. We next extend our approach to statistical aggregates such as quantiles, and data mining applications, which were not considered in Babcock et al. (2004).

4.1 Medians and quantiles

Let $v_1 < \ldots < v_N$ be the set of input tuples of the Σ operator obtained without sampling and let $v_{i_1} < \ldots < v_{i_n}$ be the samples obtained for computing query answer, where $(i_1, \ldots, i_n)$ is a subsequence of $(1, \ldots, N)$. Then the pth sample quantile is defined to be v_{i_r} where $r = \lfloor np \rfloor + 1$. The pth population quantile is defined to be $F^{-1}(p)$, where $F(x)$ is the cumulative distribution function. We can now use the result obtained in Sen (1961). Indeed, according to Sen (1961), the pth sample quantile is asymptotically normal with mean $F^{-1}(p)$ and variance

$$Var(p) = \frac{p(1-p)}{n(f(F^{-1}(p)))^2} \tag{4}$$

where F is the distribution function and $f = F'$ is the density function.

To obtain the variance of the pth sample quantile, we need

i the pth population quantile

ii the density function f.

For (i), we use the pth sample quantile to estimate $F^{-1}(p)$. For (ii), we could learn from the samples $v_{i_1} < \ldots < v_{i_n}$. If we do not restrict f to a specific class of distribution, we can use the Parzen window approximation to obtain $f(F^{-1}(p))$. Having (i) and (ii), the variance of the error model is obtained directly from equation (4). Then we can use the Bayesian approach discussed in Section 3 to improve the answer of the quantile query.

4.2 Data mining applications

Most of the mining algorithms for streaming data focus on a specific problem. For instance, the algorithm proposed in Guha et al. (2000) is for the k-Median problems in the data stream model of computation and the algorithm (Zhong, 2005) proposed by Zhong is for online k-Means clustering problem. Moreover, Case et al. (2001) focused on learning models for data produced by drifting concepts. However, since many data mining algorithms involve computations of aggregation such as AVG and COUNT, we can generalise our approach to handle these algorithms, in order to improve their performance in the presence of sampling and load shedding.

Clustering. Consider the problem of continuously finding K means over sliding windows of a data stream. Assume that only random samples are available for learning. Then we generalise our method for window averages discussed in Section 3 for computing K means: let S be the set of samples for answering the K means query q. First we find the K means of S. For each mean, we learn its prior knowledge from the past means and find its error model. Next, we combine these two components with the observed result to obtain the posterior of the approximate mean and use the MAP as the improved mean. We then assign each tuple in S to its closest improved mean.

Classification. Consider the problem of continuously building naive Bayesian classifiers using the training tuples over sliding windows of a data stream. The query network for computing the conditional distribution and the prior of each class consists of several COUNT aggregates. Then we can apply our algorithm to the counting results in order to improve the learning result when load shedding occurs.

5 Complexity

The only overhead of our algorithm is introduced by the quality-enhancement module. Indeed, there are two parts of computations involving in the module:

1 The posterior distribution of the answers

2 The detection of changes in the past answers.

For (1), if we use a normal model to fit the past answers, we only need two variables to store the mean and the variance of the past answers to obtain the prior knowledge of the current one. Also, the time for updating them is constant. Moreover, the computation of the posterior mean in equation (3) is also constant. For (2), we apply the two-sample *t*-test which only requires two more variables to store the extra statistical information. The time required for the test is also constant. Therefore, the overhead introduced by our algorithm is negligible. However, we show in the experiments that the improvement produced by our technique is so significant.

6 Experiments

The main purpose of our experiments is to test the effectiveness of our quality-enhancement technique for different kinds of aggregation queries and applications, and their robustness in the presence of non-normal distributions and concept changes. Moreover, we want to measure the computation and storage overhead introduced by this quality-enhancement step.

6.1 Experimental setup

We ran our experiments on a Linux server with 1024 MB of main memory. The dataset used contains an hour's worth of all wide-area traffic between the Lawrence Berkeley Laboratory and the rest of the world. This dataset is available for download from the Internet Traffic Archive (http://www.acm.org/sigs/sigcomm/ita/). Each record consists of six attributes including one *timestamp* column, which represents the timestamp of packet arrival and five integer-valued columns, *srcHost*, *destHost*, *srcPort*, *destPort*, and *packetSize*, which represent the source host, destination host, source TCP port, destination TCP port and number of data bytes in the packet respectively. From this dataset, we produced others by keeping the values of the first five attributes unchanged and generating synthetic values for the sixth attribute(*packetSize*). For testing the ability of adapting to concept changes, we modelled a concept drift scenario by changing the distribution of *packetSize* over time. For *k*-means, we generated a dataset with mixture distribution on *packetSize*.

To test the quality-enhancement ability of our approach, we only need a simple query network which consists of one stream source and a selection operator to filter based on source ports. In the experiments, we select *srcPort* < 1024 which are all the well-known source port numbers. There is one load shedder between the stream and the selection operator. When overload occurs, we use the load shedding technique proposed in Babcock et al. (2004) to find the sampling rate of the load shedder.

We compared the results of three methods *Approximate*, *Posterior*, *Exact*, where:

- *Approximate* uses the random sample of the input stream produced by load shedding policy in Babcock et al. (2004) to obtain an approximate answers
- *Posterior* uses the quality enhancement step described in Section 3.3 to improve the answers obtained by *Approximate*
- *Exact* uses the complete input stream to obtain the exact answers as a control.

Their continuous answers over time are plotted. Also, the average relative errors of *Approximate* and *Posterior* will be reported in each experiment.

6.2 Accuracy estimate

SUM: The aim of this experiment is to test the quality enhancement ability of our method for computing SUM on a sliding window. In this experiment, we use the original LBL dataset. With an average sampling rate for the load shedder of 0.4, we see a small improvement (4.6%) with the posterior method. We increased the arrival rate of the data streams and reduce the sampling rate on the past answers to satisfy maximum load constraint as described in Babcock et al. (2004). Then, we only see a 4.8% improvement when the average sampling rate is 0.2, but the improvement reaches 8.3% when the average sampling rate is reduced to 0.04. Indeed, when the observed answer becomes more uncertain and the variance of the error distribution becomes larger, the prior knowledge weighs more. In the expression of the mean of the posterior μ_t (in Section 3.3), μ_t is dominated by the mean of the prior μ_s and the observed answer become less important. Also, the error of posterior σ_t^2 becomes smaller and the uncertainty is further reduced, as discussed in Section 3.3. Thus the quality-enhancement method makes larger improvements when the sampling rate is reduced.

The Kolmogorov-Smirnov(KS) test (Boes et al., 1974) is widely accepted in statistics for deciding whether the prior is normal with sample mean and variance. For our LBL dataset, the maximum vertical deviation between two cumulative distribution functions (*D*-statistics) is 0.2852 and the *p*-value is $5.6e^{-29}$; these values indicate that the hypothesis that the prior is normal is very likely to be false. Then we used the Lilliefors test (Boes et al., 1974) to test whether the prior is normal with unspecified mean and variance and got the same conclusion: the prior is not normally distributed. This suggests that our method delivers reasonable improvements even for non-normal priors. To further determine the extent to which a deviation from normality of prior will reduce the effectiveness of our approach, we generated two datasets with same mean and variance of their attribute *packetSize*. However, one of them is normally distributed and the other is a mixture of normal distributions. In Table 1, we show the improvements introduced by *Posterior* on the two dataset, for which we also show their respective *p*-values produced by the KS-test. Note that the improvements on both datasets are almost the same, which indicates that our approach is robust with respect to distribution changes of the dataset.

k-Means clustering: In this experiment, we test the ability of finding *k*-means for clustering purpose under a stationary data source. For the dataset, we generated the values for the attribute *packetSize* from the mixture of two Gaussian distributions.

Then we test the ability of different methods for finding two means ($k = 2$) on the attribute *packetSize*. The average sampling rate for the load shedder is 0.49. Figures 3 and 4 show the relative errors of the two methods over time for the first mean and the second mean respectively. Moreover, the average relative errors are reported in Table 2. Thus, the posterior method significantly outperforms the approximate method that does not consider the prior knowledge about the answers.

Naive Bayesian Classifier. We further test the ability of improving the naive Bayesian classifiers. We used a tic-tac-toe data set (http://www.ics.uci.edu/»mlearn/mlrepository.html) and naive Bayesian classifier and try to predict the result of a possible board configuration. We use 90% of examples to be our training set and 10% to be our test set. Each entity has nine symbolic attributes for recording the status of nine blocks. We build

a classifier using data sets with different sampling rates (0.08–0.5). Moreover, we applied our algorithm *Posterior* to the counting results to show the improvement on the classification accuracy. The accuracies obtained by *Posterior* are about 62–66% under different sampling rates and the average accuracy improvement is about 2%. Our rough estimate is that, for the problem at hand, the improvement delivered by our method using only the prior distribution, is about 50% of that obtainable using bagging techniques on a full ensemble of several classifiers.

Table 1 Improvements of SUM for different sampling rates on normal and non-normal prior

Sampling rates	*Improvement* (%)
Normal prior (*p-value* = 0.8226)	
0.30	6.51
0.03	8.96
Non-normal prior (*p-value* = $1.8811e^{-4}$)	
0.30	6.42
0.03	8.60

Figure 3 Relative error for the first mean

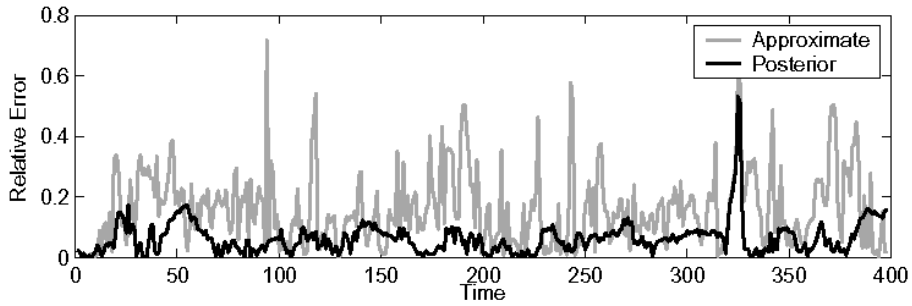


Figure 4 Relative error for the second mean

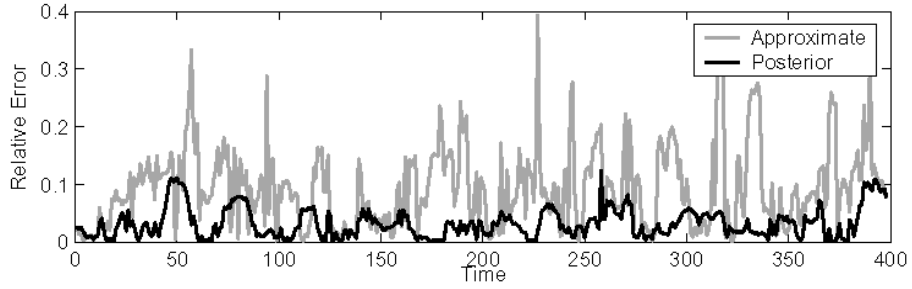


Table 2 Average relative errors for 2-means

Mean	*Approximate*	*Posterior*
1st	0.159499	0.064339
2nd	0.094051	0.032238

6.3 Quantiles and concept changes

The aims of this experiment are to test the ability of improving quantiles on sliding windows and to test whether our method can adapt to concept changes. For the LBL dataset, we generated synthetic values for the attribute *packetSize* with distribution changes over time. The average sampling rate for the load shedder is 0.02. We applied the two-sample *t*-test (Larson, 1982) to our algorithm to detect concept changes. Concept changes occurred at times: $t = 15, 65, 120, 180, 200, 210, 285, 300, 350, 375$. These times can be identified by steep drops or increases in the exact answers.

In Figure 5, we observe that the performance of our method degrades slightly when a concept changes occurs. For instance, there is a abrupt distribution change around $t = 15$. The errors of our technique (*Posterior)* is slightly increased at that time. The error is even larger than the method without prior utilisation (*Approximate*). This is consistent with the fact that prior knowledge often provides wrong information about the current answers when there is concept change. However, our method adapts to changes very quickly. The average relative errors of the approximate answers of both methods reported in Table 3 shows that the overall improvement provided by our method is almost above 19%, which is quite significant.

Figure 5 Continuous answers for windowed quantiles (80%, 60%, 40%, 20%)

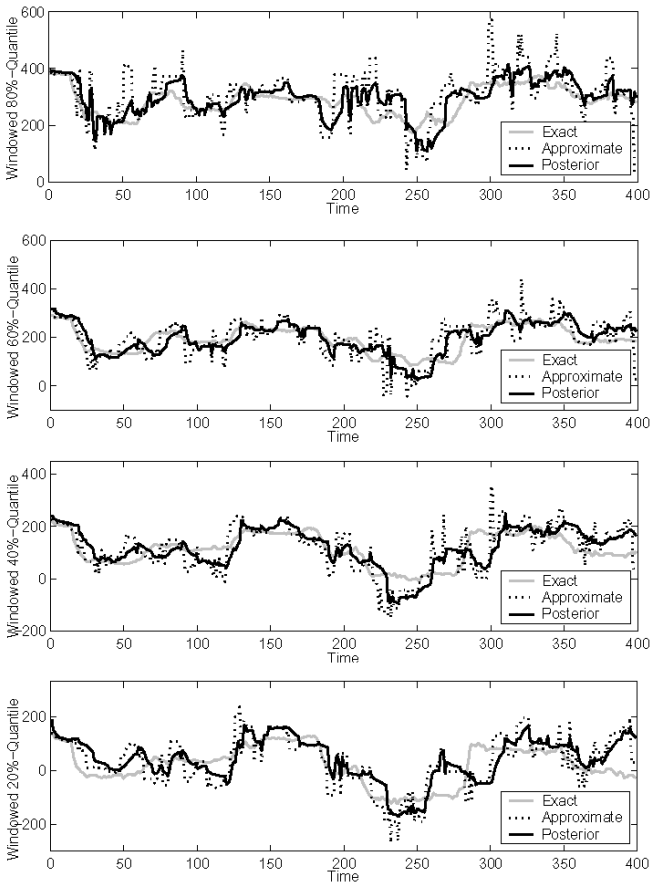


Table 3 Average rel. errors for quantiles

pth quantile (%)	*Approximate*	*Posterior*
20	3.301581	2.667624
40	1.585431	1.556009
60	0.296128	0.202173
80	0.208330	0.133511

6.4 Computational overhead

In Section 5, we showed that the storage used to perform the quality-enhancement step is negligible. We test the time spent for each kind of query including SUM, QUANTILE and K-MEANS in Table 4. Note that there are only minimal over-heads required by our technique: for the first two queries, there are only <1% overhead introduced by the quality-enhancement module. Because the queries of *k*-means requires to reassign the label for each tuple after obtaining the improved means, the overhead is about 3%, which is quite small when we got the significant improvements from this step.

Table 4 Time (in ms) for each query

Query	*Approximate*	*Posterior*
SUM	200	200
QUANTILE	1230	1240
K-MEANS	960	990

7 Conclusion and future work

Because of the high data rates and real-time response requirement characteristics of data streams, it is very important that DSMS be able to optimise the usage of sampling techniques to satisfy resource constraints. To deal with this problem, previous work (Babcock et al., 2004) have provided accuracy estimate for basic SQL aggregates. In this paper, we provide new error models for queries on order statistics and data mining functions – in addition to traditional SQL aggregates. Moreover, we propose a technique to combine our error model with statistical information from the past to further improve the quality of the query answers with minimal time and space overheads. Experiments on synthetic and real-life data sets demonstrate that our algorithm is effective at providing further improvements, under both the scenarios of stationary data sources and data with concept changes.

In terms of load shedding strategies, a natural problem that follow from our work is finding policies that maximise the combined quality of answers by selecting the best sampling rates on the input data streams under maximum load constraints. This will be a subject of further work.

Acknowledgements

The authors would like to thank the reviewers for their useful suggestions. This work was supported in part by the NSF grant IIS 0742267: "SGER: Efficient Support for Mining Queries in Data Stream Management Systems".

References

Arasu, A. and Manku, G.S. (2004) 'Approximate counts and quantiles over sliding windows', *Proc. 23rd ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, Paris, France, June, pp.286–296.

Babcock, B., Babu, S., Datar, M., Motwani, R. and Widom, J. (2002) 'Models and issues in data stream systems', *Proc. 21st ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, Madison, Wisconsin, USA, June, pp.1–16.

Babcock, B., Datar, M. and Motwani, R. (2004) 'Load shedding for aggregation queries over data streams', *Proc. 20th Intl. Conf. on Data Engineering, Boston*, MA, USA, March, pp.350–361.

Box, G.E.P. and Tiao, G.C. (1973) *Bayesian Inference in Statistical Analysis*, Addison-Wesley Publishing Company, Boston, MA, USA.

Case, J., Jain, S., Kaufmann, S., Sharma, A. and Stephan, F. (2001) 'Predictive learning models for concept drift', *Theoretical Computer Science*, Vol. 268, No. 2, pp.323–349.

Chandrasekaran, S., Cooper, O., Deshpande, A., Franklin, M.J., Hellerstein, J.M., Hong, W., Krishnamurthy, S., Madden, S.R., Raman, V., Reiss, F. and Shah, M.A. (2003) 'TelegraphCQ: continuous dataflow processing for an uncertain world', *Proc. 1st. Biennial Conf. on Innovative Data Systems Research*, January, Asilomar, CA, USA, pp.98–110.

Chen, J., DeWitt, D.J., Tian, F. and Wang, Y. (2000) 'NiagaraCQ: a scalable continuous query system for internet databases', *Proc. 2000 ACM SIGMOD Intl. Conf. on Management of Data*, Dallas, Texas, USA, May, pp.379–390.

Cortes, C., Fisher, K., Pregibon, D. and Rogers, A. (2000) 'Hancock: a language for extracting signatures from data streams', *Proc. 6th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining*, Boston, MA, USA, August, pp.9–17.

Das, A., Gehrke, J. and Riedewald, M. (2003) 'Approximate join processing over data streams', *Proc. 2003 ACM SIGMOD Intl. Conf. on Management of Data*, San Diego, California, USA, June, pp.40–51.

Dobra, A., Garofalakis, M., Gehrke, J. and Rastogi, R. (2002) 'Processing complex aggregate queries over data streams', *Proc. 2002 ACM SIGMOD Intl. Conf. on Management of Data*, Madison, Wisconsin, June, pp.61–72.

Fan, W. (2004) 'Systematic data selection to mine concept-drifting data streams', *Proc. 10th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining*, Seattle, Washington, USA, August, pp.128–137.

Guha, S., Mishra, N., Motwani, R. and O'Callaghan, L. (2000) 'Clustering data streams', *Proc. 41st Annual IEEE Symposium on Foundations of Computer Science*, Berkeley, California, USA, October, pp.359–366.

Jain, A., Chang, E. and Wang, Y. (2004) 'Adaptive stream resource management using Kalman filters', *Proc. ACM SIGMOD Intl. Conf. on Management of Data*, Paris, France, June, pp.11–22.

Jeffery, S., Alonso, G., Franklin, M., Hong, W. and Widom, J. (2006) 'A pipelined framework for online cleaning of sensor data streams', *Proc. 22nd Intl. Conf. on Data Engineering*, Atlanta, GA, USA, April, p.140.

Kang, J., Naughton, J. and Viglas, S. (2003) 'Evaluating window joins over unbounded streams', *Proc. 19th Intl. Conf. on Data Engineering*, Bangalore, India, March, pp.341–352.

Kifer, D., Ben-David, S. and Gehrke, J. (2004) 'Detecting change in data streams', *Proc. 13th Intl. Conf. on Very Large Data Bases*, Toronto, Canada, August, pp.180–191.

Larson, H. (1982) *Introduction to Probability Theory and Statistical Inference*, John Wiley & Son, Hoboken, NJ, USA.

Mood, A.M., Graybill, F.A. and Boes, D.C. (1974) *Introduction to the Theory of Statistics*, McGraw-Hill Companies, Columbus, OH, USA.

Sen, P.K. (1961) 'On some properties of the asymptotic variance of the sample quantiles and mid-ranges', *J. Royal Stat. Soc. Series B (Methodological)*, Vol. 23, No. 2, pp.453–459.

Srivastava, U. and Widom, J. (2004) 'Memory-limited execution of windowed stream joins', *Proc. 13th Intl. Conf. on Very Large Data Bases*, Toronto, Canada, August, pp.324–335.

Sullivan, M. (1996) 'Tribeca: a stream database manager for network traffic analysis', *Proc. 22th Intl. Conf. on Very Large Data Bases*, Mumbai (Bombay), India, September, p.594.

Tatbul, N., Setintemel, U., Zdonik, S., Cherniack, M. and Stonebraker, M. (2003) 'Load shedding in a data stream manager', *Proc. 29th Intl. Conf. on Very Large Data Bases*, Berlin, Germany, September, pp.309–320.

Vuran, M., Akan, Ö. and Akyildiz, I. (2004) 'Spatio-temporal correlation: theory and applications for wireless sensor networks', *Computer Networks*, Vol. 45, No. 3, pp.245–259.

Yang, Y., Wu, X. and Zhu, X. (2005) 'Combining proactive and reactive predictions for data streams', *Proc. 11th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining*, Chicago, Illinois, USA, August, pp.710–715.

Zhong, S. (2005) 'Efficient online spherical k-means clustering', *Proc. 2005 IEEE Intl. Joint Conf. on Neural Networks*, Vol. 5, pp.3180–3185.

Websites

Internet traffic archive, Trace LBL-PKT-5, http://www.acm.org/sigs/sigcomm/ita/

UCI machine learning databases, http://www.ics.uci.edu/~mlearn/mlrepository.html